Amazon Simple Storage Service

Entwicklerhandbuch

API-Version 2006-03-01

Amazon Simple Storage Service: Entwicklerhandbuch

Table of Contents

Was ist Amazon S3?

Bei Amazon Simple Storage Service handelt es sich um Speicher für das Internet. Der Service ist darauf ausgelegt, Cloud Computing für Entwickler zu erleichtern.

Amazon S3 besitzt eine einfache Web-Service-Schnittstelle zum Speichern und Abrufen einer beliebigen Datenmenge zu jeder Zeit und von jedem Ort im Internet aus. Mit Amazon S3 haben Entwickler Zugriff auf dieselbe hochgradig skalierbare, zuverlässige, schnelle und kostengünstige Datenspeicherinfrastruktur, die Amazon zum Ausführen seines eigenen globalen Website-Netzwerks verwendet. Somit können auch Entwickler von den Vorteilen einer flexiblen Skalierbarkeit profitieren.

Diese Anleitung erklärt die wichtigsten Konzepte von Amazon S3 wie beispielsweise Buckets, Zugriffspunkte und Objekte, und wie Sie mit diesen Ressourcen unter Verwendung der Amazon S3-API (Application Programming Interface) arbeiten.

Wie gehe ich vor...?

Informationen	Relevante Abschnitte
Allgemeine Produktübersicht und Preise	Amazon S3
Eine schnelle praktische Einführung in Amazon S3.	Amazon Simple Storage Service Handbuch Erste Schritte
Mehr über die wichtigste Terminologie von Amazon S3 und seine Konzepte	Einführung in Amazon S3 (p. 2)
Wie arbeite ich mit Buckets?	Arbeiten mit Amazon S3-Buckets (p. 57)
Wie arbeite ich mit Zugriffspunkten?	Verwalten des Datenzugriffs mit Amazon S3 Access Points (p. 103)
Wie arbeite ich mit Objekten?	Arbeiten mit Amazon S3-Objekten (p. 116)
Wie mache ich Anfragen?	Senden von Anforderungen (p. 11)
Wie verwalte ich den Zugriff auf meine Ressourcen?	Identity and Access Management in Amazon S3 (p. 329)

Einführung in Amazon S3

Diese Einführung in Amazon Simple Storage Service (Amazon S3) enthält einen detaillierten Überblick zu diesem Web-Service. Nach der Lektüre dieses Abschnitts sollten Sie sich ein Bild davon machen können, welche Vorteile diese Lösung bietet und wie Sie sie in Ihren Geschäftsablauf integrieren können.

Themen

Übersicht über Amazon S3 und dieses Handbuch

Amazon S3 besitzt eine einfache Web-Service-Schnittstelle zum Speichern und Abrufen einer beliebigen Datenmenge zu jeder Zeit und von jedem Ort im Internet aus.

Dieses Handbuch beschreibt, wie Sie Anfragen senden, Buckets zu erstellen, Ihre Objekte zu speichern und abzurufen und Berechtigungen für Ihre Ressourcen zu verwalten. Das Handbuch beschreibt außerdem die Zugriffskontrolle und den Authentifizierungsprozess. Die Zugriffskontrolle definiert, wer auf Objekte und Buckets in Amazon S3 zugreifen kann, ebenso wie den Zugriffstyp (z. B. READ und WRITE). Der Authentifizierungsprozess überprüft die Identität eines Benutzers, der versucht, auf die Amazon Web Services (AWS) zuzugreifen.

Vorteile der Verwendung von Amazon S3

Amazon S3 wurde ursprünglich mit einer minimalen Funktionsmenge angelegt, die einfach und robust sein sollte. Im Folgenden sind einige Vorteile der Verwendung von Amazon S3 aufgeführt:

- Erstellen von Buckets: Buckets zum Speichern von Daten erstellen und benennen. Buckets sind die grundlegende Behälter für die Datenspeicherung in Amazon S3.
- Speichern von Daten: unbegrenzte Menge an Daten in einem Bucket speichern. Sie können beliebig viele Objekte in einen Amazon S3-Bucket hochladen. Jedes Objekt kann bis zu 5 TB Daten enthalten. Jedes Objekt wird unter Verwendung eines eindeutigen, vom Entwickler zugewiesenen Schlüssels gespeichert.
- Herunterladen von Daten: Daten herunterladen oder anderen Benutzern dies ermöglichen. Sie können Daten jederzeit herunterladen oder anderen Benutzern dies ermöglichen.
- Berechtigungen – Sie können den Zugriff für andere Benutzer gewähren oder ablehnen, die Daten zu Ihrem Amazon S3-Bucket hochladen oder aus diesem herunterladen möchten. Erteilen von Upload- und Download-Berechtigungen für drei Benutzertypen. Authentifizierungsmechanismen können Ihnen helfen, Daten vor unbefugtem Zugriff zu schützen.

- Standardschnittstellen: Sie können standardbasierte REST- und SOAP-Schnittstellen verwenden, die für beliebige Toolkits für Internet-basierte Entwicklung entworfen wurden.

  Note

  Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Amazon S3-Konzepte

Dieser Abschnitt beschreibt die wichtigsten Konzepte und die Terminologie, die Sie benötigen, um Amazon S3 effektiv nutzen zu können. Sie werden in der Reihenfolge vorgestellt, in der Sie ihnen am wahrscheinlichsten begegnen.

Themen

Buckets

Ein Bucket ist ein Container für Objekte, die in Amazon S3 gespeichert werden. Jedes Objekt ist in einem Bucket enthalten. Wenn beispielsweise ein Objekt mit dem Namen `photos/puppy.jpg` im Bucket `awsexamplebucket1` in der Region USA West (Oregon) gespeichert ist, ist es über die URL `https://awsexamplebucket1.s3.us-west-2.amazonaws.com/photos/puppy.jpg` adressierbar.

Buckets dienen verschiedenen Zwecken:

- Sie strukturieren den Amazon S3-Namespace auf der höchsten Ebene.
- Sie identifizieren das Konto, dem die Gebühren für Datenspeicherung und -übertragung belastet werden.
- Sie werden für die Zugriffskontrolle eingesetzt.
- Sie dienen im Rahmen der Erstellung von Nutzungsberichten als Auswertungseinheit.

Sie können Buckets so konfigurieren, dass sie in einer bestimmten AWS-Region erstellt werden. Weitere Informationen siehe Zugriff auf einen Bucket (p. 60). Sie können einen Bucket auch so konfigurieren, dass Amazon S3 bei jedem Hinzufügen eines Objekts eine eindeutige Versions-ID erzeugt und sie dem Objekt zuweist. Weitere Informationen finden Sie unter Verwendung von Versioning (p. 514).

Weitere Informationen über Buckets finden Sie unter Arbeiten mit Amazon S3-Buckets (p. 57).

Objekte

Objekte sind die Grundeinheiten, die in Amazon S3 gespeichert sind. Objekte bestehen aus Objekt- und Metadaten. Der Datenanteil ist für Amazon S3 nicht einsichtig. Metadaten bestehen aus mehreren Name/Wert-Paaren, die das Objekt beschreiben. Dazu gehören Standardmetadaten wie das Datum der letzten Aktualisierung und HTTP-Standardmetadaten wie `Content-Type`. Sie können bei der Speicherung des Objekts auch benutzerdefinierte Metadaten angeben.

Ein Objekt wird innerhalb eines Buckets eindeutig durch einen Schlüssel (Name) und eine Versions-ID identifiziert. Weitere Informationen siehe Schlüssel (p. 4) und Verwenden von Versioning (p. 514).

Schlüssel

Ein Schlüssel ist der eindeutige Bezeichner für ein Objekt in einem Bucket. Jedes Objekt in einem Bucket besitzt genau einen Schlüssel. Jedes Objekt wird durch die Kombination aus Bucket, Schlüssel und Versions-ID eindeutig identifiziert. Amazon S3 fungiert also als grundlegende Datenzuordnung zwischen "Bucket + Schlüssel + Version" und dem Objekt selbst. Jedes Objekt in Amazon S3 ist über eine Kombination von Webservice-Endpunkt, Bucket-Name, Schlüssel und wahlweise einer Version aufrufbar. So ist "`doc`" in der URL `https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl` der Name des Buckets und "`2006-03-01/AmazonS3.wsdl`" der Schlüssel.

Weitere Informationen zu Objektschlüsseln finden Sie unter Objektschlüssel.

Regionen

Sie können auswählen, in welcher geografischen Region Amazon S3 die erstellten Buckets speichern soll. Sie sollten eine Region im Hinblick auf Latenz, Kosten sowie Einhaltung der relevanten Vorschriften auswählen. In einer Region gespeicherte Objekte verbleiben in der Region, bis sie explizit in eine andere Region verschoben werden. In der Region Europa (Irland) gespeicherte Objekte verlassen diese Region beispielsweise nie.

> Note
>
> Sie können auf Amazon S3 und die zugehörigen Funktionen nur in AWS-Regionen zugreifen, die für das Konto aktiviert sind.

Die Liste der Amazon S3-Regionen und -Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS-Referenz.

Amazon S3 Datenkonsistenzmodell

Amazon S3 bietet Lesen-nach-Schreiben-Konsistenz für PUTS neuer Objekte in den S3-Bucket in allen Regionen mit einer Vorsichtsmaßnahme. Die Einschränkung besteht darin, dass wenn Sie eine HEAD- oder GET-Anforderung an einen Schlüsselnamen stellen, bevor das Objekt erstellt wird, und dann kurz danach das Objekt erstellen, eine nachfolgende GET-Anforderung das Objekt aufgrund letztendlicher Datenkonsistenz (Eventual Consistency) möglicherweise nicht zurückgibt.

Amazon S3 bietet mögliche Konsistenz für das Überschreiben von PUTS und DELETES in allen Regionen.

Aktualisierungen an einem einzelnen Schlüssel sind unteilbar. Wenn Sie z. B. einen vorhandenen Schlüssel eingeben, kann ein anschließendes Lesen die alten Daten oder die aktualisierten Daten zurückgeben, aber es werden niemals beschädigte oder unvollständige Daten zurückgegeben.

Amazon S3 erzielt hohe Verfügbarkeit, indem die Daten innerhalb der AWS-Rechenzentren über mehrere Server repliziert werden. Wenn eine PUT-Anfrage erfolgreich ist, sind die Daten sicher gespeichert. Die Informationen zu den Änderungen werden jedoch möglicherweise nicht sofort überall auf Amazon S3 repliziert und Sie stellen u. U. die folgenden Verhaltensweisen fest.

- Ein Prozess schreibt ein neues Objekt in Amazon S3 und listet sofort Schlüssel innerhalb seines Buckets auf. Bis die Änderung vollständig weitergegeben wurde, wird das Objekt möglicherweise nicht in der Liste angezeigt.
- Ein Prozess ersetzt ein vorhandenes Objekt und versucht sofort, es zu lesen. Bis die Änderung vollständig weitergegeben wurde, gibt Amazon S3 möglicherweise die vorherigen Daten zurück.

- Ein Prozess löscht ein vorhandenes Objekt und versucht sofort, es zu lesen. Bis das Löschen vollständig weitergegeben wurde, gibt Amazon S3 möglicherweise die gelöschten Daten zurück.
- Ein Prozess löscht ein vorhandenes Objekt und listet sofort Schlüssel innerhalb seines Buckets auf. Bis das Löschen vollständig weitergegeben wurde, listet Amazon S3 möglicherweise das gelöschte Objekt auf.

> **Note**
>
> Amazon S3 unterstützt derzeit keine Objektsperre. Wenn gleichzeitig zwei PUT-Anforderungen für denselben Schlüssel eingehen, hat die Anforderung mit dem ältesten Zeitstempel Priorität. Wenn das ein Problem ist, müssen Sie in Ihre Anwendung einen Sperrmechanismus für Objekte einbauen.
> Updates basieren auf Schlüsseln. Es gibt keine Möglichkeit für unteilbare Aktualisierungen über Schlüssel hinweg. Sie können beispielsweise die Aktualisierung eines Schlüssels nicht von der Aktualisierung eines anderen Schlüssels abhängig machen. Dazu müssten Sie diese Funktionalität in Ihrer Anwendung implementieren.

Buckets verfügen über ein ähnliches Konsistenzmodell mit den gleichen Einschränkungen. Wenn Sie beispielsweise einen Bucket löschen und sofort alle Buckets auflisten, wird Amazon S3 möglicherweise weiterhin in der Liste angezeigt.

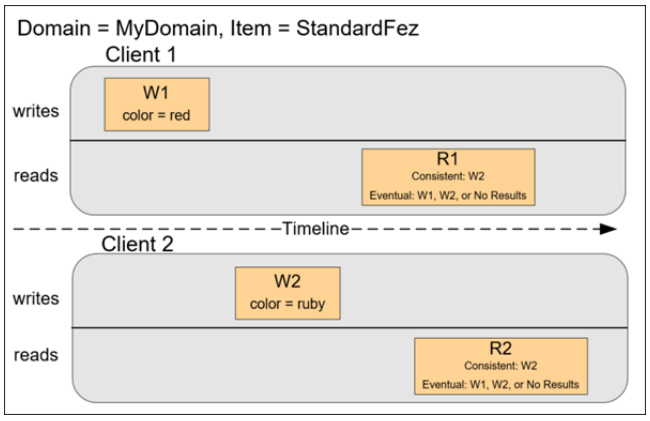
Die folgende Tabelle beschreibt die Merkmale von Eventually Consistent-Lesevorgängen und Consistent-Lesevorgängen.

Eventually Consistent-Lesevorgang	Consistent-Lesevorgang
Das Lesen veralteter Daten ist möglich	Das Lesen veralteter Daten ist nicht möglich
Niedrigste Lese-Latenz	Potenziell höhere Lese-Latenz
Höchster Lesedurchsatz	Potenziell geringerer Lesedurchsatz

Gleichzeitige Anwendungen

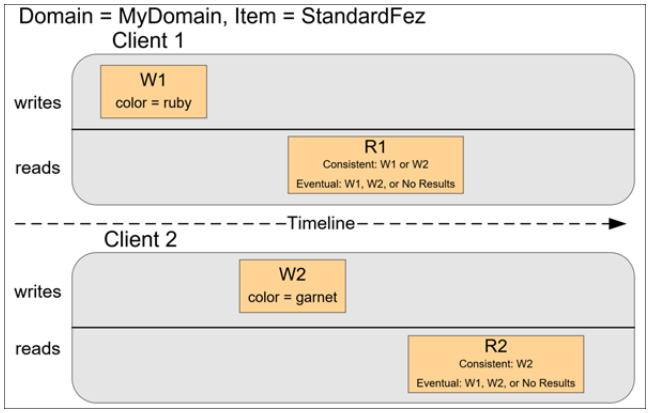
Dieser Abschnitt zeigt Beispiele für Anforderungen von Eventually Consistent- und Consistent-Lesevorgängen, wenn mehrere Clients gleichzeitig in dieselben Elemente schreiben.

In diesem Beispiel werden W1 (Lesen 1) und W2 (Lesen 2) abgeschlossen, bevor R1 (Lesen 1) und R2 (Lesen 2) starten. Für einen Consistent-Lesevorgang geben R1 und R2 beide `color = ruby` zurück. Für einen Eventually Consistent-Lesevorgang können R1 und R2 `color = red` oder `color = ruby` zurückgeben, abhängig von der verstrichenen Zeit.

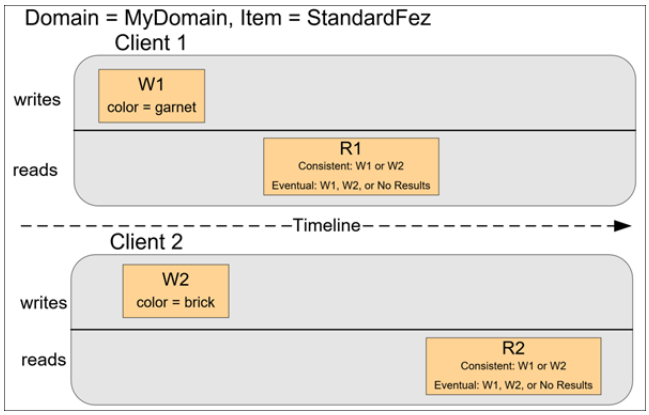


Im nächsten Beispiel ist W2 vor dem Start von R1 nicht abgeschlossen. Aus diesem Grund könnte R1 `color = ruby` oder `color = garnet` für einen Consistent- oder Eventually Consistent-Lesevorgängen zurückgeben. Ein Eventually Consistent-Lesevorgang könnten überhaupt keine Ergebnisse zurückgeben, abhängig von der vergangenen Zeit.

Für einen Consistent-Lesevorgang gibt R2 `color = garnet` zurück. Für einen Eventually Consistent-Lesevorgang kann R2 `color = ruby` oder `color = garnet` zurückgeben, abhängig von der verstrichenen Zeit.



Im letzten Beispiel führt Client 2 W2 aus, bevor Amazon S3 eine Erfolgsmeldung für W1 zurückgibt. Daher ist der endgültige Wert unbekannt (`color = garnet` oder `color = brick`). Alle nachfolgenden Lesevorgänge (Consistent- oder Eventually Consistent-Lesevorgänge) könnten jeden Wert zurückgeben. Ein Eventually Consistent-Lesevorgang könnten überhaupt keine Ergebnisse zurückgeben, abhängig von der vergangenen Zeit.



Amazon S3-Funktionen

Dieser Abschnitt beschreibt wichtige Funktionen von Amazon S3.

Themen

Speicherklassen

Amazon S3 bietet eine Reihe von Speicherklassen, die für verschiedene Anwendungsfälle ausgelegt sind. Dazu gehören Amazon S3 STANDARD für die allgemeine Speicherung häufig abgerufener Daten, Amazon S3 STANDARD_IA für langlebige, aber weniger häufig abgerufene Daten und S3 Glacier für die Langzeitarchivierung.

Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122).

Bucket-Richtlinien

Bucket-Richtlinien bieten eine zentrale Zugriffskontrolle für Buckets und Objekte basierend auf verschiedenen Bedingungen, unter anderem Amazon S3-Operationen, Anforderer, Ressourcen und Aspekte der Anforderung (z. B. IP-Adresse). Die Richtlinien sind in der Sprache der Zugriffsrichtlinie verfasst und unterstützen die zentrale Verwaltung von Berechtigungen. Die einem Bucket zugeordneten Berechtigungen gelten für alle Objekte in diesem Bucket.

Bucket-Richtlinien können von Einzelpersonen und Unternehmen genutzt werden. Wenn Unternehmen sich bei Amazon S3 registrieren, erstellen sie ein Konto. Anschließend ist das Unternehmen synonym mit dem Konto. Konten tragen die Kosten für die AWS-Ressourcen, die sie (und ihre Mitarbeiter) erstellen. Konten besitzen die Möglichkeit, Bucket-Richtlinienberechtigungen zu erteilen und Mitarbeitern basierend auf verschiedensten Bedingungen Berechtigungen zuzuweisen. Beispielsweise könnte ein anderes Konto eine Richtlinie erstellen, die einem Benutzer Schreibzugriff erteilt:

- Für einen bestimmten S3-Bucket
- Aus dem Unternehmensnetzwerk eines Kontos
- Während der Geschäftszeiten

Ein Konto kann einem Benutzer begrenzten Lese- und Schreibzugriff erteilen, einem anderen dagegen zusätzlich erlauben, Buckets zu erstellen und zu löschen. Ein Konto kann verschiedenen Niederlassungen erlauben, die Tagesberichte in einem einzelnen Bucket zu speichern. Es kann jeder Niederlassung gestatten, nur einen bestimmten Satz von Namen (z. B. "Nevada/*" oder "Utah/*") und nur aus dem IP-Adressbereich der betreffenden Niederlassung zu schreiben.

Anders als Zugriffskontrolllisten (siehe unten), die Berechtigungen nur für einzelne Objekte hinzufügen (gewähren), können Richtlinien Berechtigungen für alle Objekte (oder eine Untermenge davon) in einem Bucket hinzufügen oder verweigern. Mit einer Anforderung kann ein Konto Berechtigungen für beliebig viele Objekte in einem Bucket einrichten. Ein Konto kann Platzhalterzeichen (vergleichbar den Operatoren für reguläre Ausdrücke) in Amazon-Ressourcennamen (ARNs) und anderen Werten verwenden. Das Konto kann so den Zugriff auf Gruppen von Objekten steuern, die ein gemeinsames Präfix oder einen gemeinsame Erweiterung wie .html aufweisen.

Nur der Bucket-Eigentümer darf einem Bucket eine Richtlinie zuzuordnen. Richtlinien (geschrieben in der Sprache der Zugriffsrichtlinie) erlauben oder verweigern Anforderungen basierend auf folgenden Kriterien:

- Amazon S3-Bucket-Operationen (wie `PUT ?acl` und Objektoperationen (wie `PUT Object` oder `GET Object`)
- Auftraggeber
- In der Richtlinie angegebene Bedingungen

Ein Konto kann den Zugriff auf der Basis bestimmter Amazon S3-Operationen steuern, beispielsweise `GetObject`, `GetObjectVersion`, `DeleteObject` oder `DeleteBucket`.

Bei den Bedingungen kann es sich um Dinge wie IP-Adressen, IP-Adressbereiche in CIDR-Notation, Datumswerte, Benutzeragenten, HTTP-Referrer und Transportprotokolle (HTTP und HTTPS) handeln.

Weitere Informationen finden Sie unter Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375).

AWS Identity and Access Management

Mit AWS Identity and Access Management (IAM) können Sie den Zugriff auf Amazon S3-Ressourcen verwalten.

Sie können z. B. IAM zusammen mit Amazon S3 verwenden, um die Zugriffsart für einen Benutzer oder eine Benutzergruppe für bestimmte Teile eines Amazon S3-Buckets, den das AWS-Konto besitzt, zu steuern.

Weitere Informationen zu IAM finden Sie unter:

- AWS Identity and Access Management (IAM)
- Erste Schritte
- IAM-Benutzerhandbuch

Zugriffskontrolllisten

Sie können den Zugriff auf alle Buckets und Objekte mit einer Zugriffskontrollrichtlinie (ACL) steuern. Weitere Informationen finden Sie unter Zugriffsverwaltung mit ACLs (p. 479).

Versioning

Mit Versioning ist es möglich, mehrere Versionen eines Objekts in einem Bucket zu verwalten. Weitere Informationen finden Sie unter Objekt-Versioning (p. 127).

Operationen

Nachfolgend finden Sie die gebräuchlichsten Operationen, die über die API ausgeführt werden.

Allgemeine Vorgänge

- Erstellen eines Buckets: Erstellen und benennen Sie einen eigenen Bucket, in dem die Objekte gespeichert werden sollen.
- Schreiben eines Objekts: Speichern Sie Daten durch Erstellen oder Überschreiben eines Objekts. Wenn Sie ein Objekt schreiben, müssen Sie einen im Namespace Ihres Buckets eindeutigen Schlüssel angeben. Dies ist auch ein guter Moment, eine etwaige Zugriffskontrolle für das Objekt festzulegen.
- Lesen eines Objekts: Daten auslesen. Sie können die Daten über HTTP oder BitTorrent herunterladen.
- Löschen eines Objekts: Löschen Sie Daten selektiv.
- Auflisten von Schlüsseln: Listen Sie die in einem Ihrer Buckets enthaltenen Schlüssel auf. Sie können die Schlüsselliste basierend auf einem Präfix filtern.

Diese Operationen und alle anderen Funktionen werden in diesem Handbuch beschrieben.

Amazon S3 Application Programming Interfaces (API)

Die Amazon S3-Architektur ist so ausgelegt, dass sie unabhängig von Programmiersprachen ist und die AWS-unterstützten Schnittstellen verwendet, um Objekte zu speichern und abzurufen.

Amazon S3 unterstützt eine REST- und eine SOAP-Schnittstelle. Diese sind ähnlich, es gibt jedoch einige Unterschiede. Beispielsweise werden über die REST-Schnittstelle Metadaten in HTTP-Headern zurückgegeben. Wir unterstützen nur HTTP-Anfragen von bis zu 4 KB (ohne den Rumpf), deshalb ist die Menge der von Ihnen bereitgestellten Metadaten begrenzt.

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Die REST-Schnittstelle

Die REST API ist eine HTTP-Schnittstelle für Amazon S3. Mit REST verwenden Sie HTTP-Standardanfragen, um Buckets und Objekte zu erstellen, laden oder löschen.

Sie können einen beliebigen Toolkit einsetzen, der HTTP unterstützt, um die REST API verwenden zu können. Sie können sogar einen Browser verwenden, um Objekte zu laden, wenn diese anonym lesbar sind.

Die REST API verwendet die HTTP-Standardheader und -Statuscodes, sodass sich Standard-Browser und -Toolkits wie erwartet verhalten. In einigen Bereichen haben wir HTTP um zusätzliche Funktionen erweitert (wir haben beispielsweise Header hinzugefügt, um die Zugriffskontrolle zu unterstützen). In diesen Fällen haben wir alles dafür getan, die neue Funktion so hinzuzufügen, dass sie der standardmäßigen Nutzung von HTTP entsprechen.

Die SOAP-Schnittstelle

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Die SOAP API bietet eine SOAP 1.1-Schnittstelle mit Dokumentliteral-Codierung. SOAP wird am häufigsten verwendet, um die WSDL herunterzuladen (siehe https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl), dann mittels eines SOAP-Toolkits wie Apache Axis oder Microsoft .NET Bindungen zu erstellen und anschließend Code zu schreiben, der diese Bindungen für den Aufruf von Amazon S3 verwendet.

Bezahlen für Amazon S3

Die Preise für Amazon S3 sind so ausgelegt, dass Sie nicht im Hinblick auf die Speicheranforderungen Ihrer Anwendung planen müssen. Die meisten Speicheranbieter zwingen Sie, eine vorab festgelegte Speichermenge und Netzwerkübertragungskapazität zu kaufen: Wenn Sie diese Kapazität überschreiten, wird Ihr Service abgeschaltet, und Sie zahlen hohe Überziehungsgebühren. Wenn Sie diese Kapazität nicht überschreiten, zahlen Sie genauso viel, als wenn Sie sie verwendet hätten.

Amazon S3 stellt Ihnen nur Gebühren für Kapazitäten in Rechnung, die Sie tatsächlich genutzt haben, ohne verborgene Kosten und Überziehungsgebühren. Auf diese Weise erhalten die Entwickler einen Service mit variablen Kosten, der zusammen mit ihrem Unternehmen wächst, während sie gleichzeitig die Kostenvorteile der AWS-Infrastruktur genießen.

Bevor Sie etwas in Amazon S3 speichern, müssen Sie sich beim Service registrieren und eine Zahlungsweise angeben, damit die Kosten am Monatsende belastet werden können. Es gibt keine

Einrichtungsgebühren für den Service. Die festgelegte Zahlungsweise wird am Ende des Monats automatisch mit dem Betrag für die monatliche Nutzung belastet.

Information über die Bezahlung von Amazon S3-Speicher finden Sie unter Amazon S3-Preise.

Zugehörige Services

Nachdem Sie Daten in Amazon S3 hochgeladen haben, können Sie sie mit anderen AWS-Services nutzen. Häufig genutzte Services:

- Amazon Elastic Compute Cloud (Amazon EC2): Dieser Service stellt virtuelle Datenverarbeitungsressourcen in der Cloud bereit. Weitere Informationen finden Sie auf der Produktdetailseite von Amazon EC2.
- Amazon EMR: Mit diesem Service können Unternehmen, Forschungseinrichtungen, Datenanalysten und Entwickler einfach und kosteneffektiv riesige Datenmengen verarbeiten. Er nutzt ein gehostetes Hadoop-Framework, das auf der Amazon EC2- und Amazon S3-Infrastruktur im Web-Maßstab ausgeführt wird. Weitere Informationen finden Sie auf der Produktdetailseite von Amazon EMR.
- AWS Snowball: Dieser Service beschleunigt die Übertragung großer Datenmengen von und nach AWS mit physischen Speichergeräten durch Umgehung des Internets. Mit allen AWS Snowball-Gerätetypen lassen sich Daten schneller als im Internet transportieren. Die Geräte mit den Daten werden durch einen regionalen Kurierdienst transportiert. Weitere Informationen finden Sie auf der Produktdetailseite von AWS Snowball.

Senden von Anforderungen

Themen

Amazon S3 ist ein REST-Service. Sie können Anfragen an Amazon S3 unter Verwendung der REST-API oder der Wrapper-Bibliotheken des AWS SDK (siehe Beispiel-Code und Bibliotheken) senden, die zur Vereinfachung der Programmierarbeit die zugrunde liegende Amazon S3-REST-API umschließen.

Jede Interaktion mit Amazon S3 erfolgt entweder authentifiziert oder anonym. Die Authentifizierung ist ein Vorgang, bei dem die Identität des Auftraggebers, der auf ein Amazon Web Services (AWS)-Produkt zugreifen möchte, überprüft wird. Authentifizierte Anfragen müssen einen Signaturwert enthalten, der den Sender der Anfrage authentifiziert. Ein Teil des Signaturwerts wird von den AWS-Zugriffsschlüsseln des Auftraggebers generiert (Zugriffsschlüssel-ID und geheimer Zugriffsschlüssel). Weitere Informationen zum Abrufen von Zugriffsschlüsseln finden Sie unter Wie erhalte ich Sicherheitsanmeldeinformationen? in der AWS General Reference.

Wenn Sie das AWS-SDK verwenden, berechnen die Bibliotheken die Signatur anhand der von Ihnen bereitgestellten Schlüssel. Wenn Sie in Ihrer Anwendung jedoch direkte REST-API-Aufrufe senden, müssen Sie den Code für die Berechnung der Signatur schreiben und diesen der Anfrage hinzufügen.

Über Zugriffsschlüssel

In den folgenden Abschnitten werden die Arten der Zugriffsschlüssel vorgestellt, die Sie verwenden können, um authentifizierte Anfragen zu senden.

AWS-Kontozugriffsschlüssel

Die Kontozugriffsschlüssel stellen vollständigen Zugriff auf die AWS-Ressourcen bereit, die das jeweilige Konto besitzt. Im Folgenden finden Sie Beispiele für Zugriffsschlüssel:

- Zugriffsschlüssel-ID (eine alphanumerische Zeichenfolge mit 20 Zeichen). Beispiel: AKIAIOSFODNN7EXAMPLE
- Geheimer Zugriffsschlüssel (eine Zeichenfolge mit 40 Zeichen). Beispiel: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

Die Zugriffsschlüssel-ID identifiziert ein AWS-Konto auf spezifische Weise. Sie können diese Zugriffsschlüssel verwenden, um authentifizierte Anfragen an Amazon S3 zu senden.

IAM-Benutzer-Zugriffsschlüssel

Sie können ein einzelnes AWS-Konto für Ihr Unternehmen erstellen. Möglicherweise gibt es jedoch mehrere Mitarbeiter in der Organisation, die auf die AWS-Ressourcen Ihrer Organisation zugreifen müssen. Wenn Sie die Zugriffsschlüssel für Ihr AWS-Konto freigeben, wird die Sicherheit reduziert. Das Erstellen

einzelner AWS-Konten für jeden Mitarbeiter ist jedoch vielleicht nicht praktikabel. Darüber hinaus können Sie Ressourcen wie Buckets und Objekte nicht einfach freigeben, da sie im Besitz verschiedener Konten sind. Um Ressourcen freizugeben, müssen Sie Berechtigungen gewähren. Dies bedeutet zusätzlichen Aufwand.

In solchen Szenarien können Sie AWS Identity and Access Management (IAM) zum Erstellen von Benutzern mit eigenen Zugriffsschlüsseln in Ihrem AWS-Konto verwenden und IAM-Benutzerrichtlinien anfügen, die diesen Benutzern die entsprechenden Zugriffsberechtigungen für Ressourcen gewähren. Um diese Benutzer besser verwalten zu können, ermöglicht IAM Ihnen das Erstellen von Benutzergruppen und das Gewähren von Berechtigungen auf Gruppenebene, die für alle Benutzer in der betreffenden Gruppe gelten.

Diese Benutzer werden als IAM-Benutzer bezeichnet. Sie erstellen und verwalten diese innerhalb von AWS. Das übergeordnete Konto steuert die Möglichkeiten der Benutzer, auf AWS zuzugreifen. Alle von einem IAM-Benutzer erstellten Ressourcen werden über das übergeordnete AWS-Konto gesteuert und bezahlt. Diese IAM-Benutzer können unter Verwendung ihrer eigenen Sicherheitsanmeldeinformationen authentifizierte Anfragen an Amazon S3 senden. Weitere Informationen zum Erstellen und Verwalten von Benutzern in Ihrem AWS-Konto finden Sie auf der AWS Identity and Access Management-Produktdetailseite.

Temporäre Sicherheitsanmeldeinformationen

Zusätzlich zur Erstellung von IAM-Benutzern mit eigenen Zugriffsschlüsseln ermöglicht IAM Ihnen auch die Gewährung temporärer Sicherheitsanmeldeinformationen (temporärer Zugriffsschlüssel und Sicherheitstoken) für IAM-Benutzer, damit diese auf Ihre AWS-Services und -Ressourcen zugreifen können. Sie können Benutzer auch außerhalb von AWS in Ihrem System verwalten. Diese Benutzer werden als verbundene Benutzer bezeichnet. Zusätzlich kann es sich bei Benutzern um Anwendungen handeln, die Sie erstellen, um auf Ihre AWS-Ressourcen zuzugreifen.

IAM stellt die AWS Security Token Service-API bereit, über die Sie temporäre Sicherheitsanmeldeinformationen anfordern können. Sie können entweder die AWS STS-API oder das AWS SDK verwenden, um diese Anmeldeinformationen anzufordern. Die API gibt temporäre Sicherheitsanmeldeinformationen (Zugriffsschlüssel-ID und geheimen Zugriffsschlüssel) und ein Sicherheitstoken zurück. Diese Anmeldeinformationen sind nur für die Dauer gültig, die Sie bei der Anforderung angeben. Sie verwenden die Zugriffsschlüssel-ID und den geheimen Schlüssel auf dieselbe Weise, wie Sie diese beim Senden von Anfragen über Ihr AWS-Konto oder IAM-Benutzerzugriffsschlüssel verwenden. Zusätzlich muss jede Anfrage, die Sie an Amazon S3 senden, das Token enthalten.

Ein IAM-Benutzer kann diese temporären Sicherheitsanmeldeinformationen für die eigene Verwendung anfordern oder sie an verbundene Benutzer oder Anwendungen vergeben. Wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, in der die Berechtigungen definiert sind, die Sie mit diesen temporären Sicherheitsanmeldeinformationen verknüpfen möchten. Ein verbundener Benutzer kann nicht mehr Berechtigungen als der übergeordnete IAM-Benutzer erhalten, der die temporären Anmeldeinformationen angefordert hat.

Sie können diese temporären Sicherheitsanmeldeinformationen beim Senden von Anfragen an Amazon S3 verwenden. Die API-Bibliotheken berechnen anhand dieser Anmeldeinformationen den notwendigen Signaturwert, um Ihre Anfrage zu authentifizieren. Wenn Sie beim Senden von Anfragen abgelaufene Anmeldeinformationen verwenden, lehnt Amazon S3 die Anfrage ab.

Informationen zum Signieren von Anfragen mittels temporärer Sicherheitsanmeldeinformationen in Ihren REST-API-Anfragen finden Sie unter Signieren und Authentifizieren von REST-Anforderungen (p. 821). Informationen zum Senden von Anfragen mittels AWS-SDKs finden Sie unter Senden von Anfragen unter Verwendung der AWS SDKs (p. 20).

Weitere Informationen zur IAM-Unterstützung für temporäre Sicherheitsanmeldeinformationen finden Sie unter Temporäre Sicherheits-Anmeldeinformationen im IAM-Benutzerhandbuch.

Um die Sicherheit zu verbessern, können Sie für den Zugriff auf Ihre Amazon S3-Ressourcen eine Multi-Factor-Authentication (MFA) anfordern, indem Sie eine Bucket-Richtlinie konfigurieren. Weitere Informationen finden Sie unter Hinzufügen einer Bucket-Richtlinie zur Anforderung einer MFA (p. 451). Nachdem Sie für den Zugriff auf Ihre Amazon S3-Ressourcen eine MFA angefordert haben, können Sie auf diese Ressourcen nur durch die Bereitstellung temporärer Anmeldeinformationen zugreifen, die mittels eines MFA-Schlüssels erstellt wurden. Weitere Informationen finden Sie auf der Detailseite für die AWS-Multifaktor-Authentifizierung und unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Anforderungsendpunkte

Sie senden REST-Anfragen an den vordefinierten Endpunkt des Service. Die Liste aller AWS-Services und ihrer entsprechenden Endpunkte finden Sie unter Regionen und Endpunkte in der AWS General Reference.

Senden von Anforderungen an Amazon S3 über IPv6

Amazon Simple Storage Service (Amazon S3) unterstützt den Zugriff auf S3-Buckets über das Internetprotokollversion 6 (IPv6)-Protokoll, zusätzlich zum IPv4-Protokoll. Amazon S3-Dual-Stack-Endpunkte unterstützen Anfragen an S3-Buckets über IPv6 und IPv4. Für den Zugriff auf Amazon S3 über IPv6fallen keine zusätzlichen Gebühren an. Weitere Informationen zu Preisen finden Sie unter Amazon S3-Preise.

Themen

Erste Schritte für Anforderungen über IPv6

Um eine Anfrage für einen S3-Bucket über IPv6 zu erstellen, brauchen Sie einen Dual-Stack-Endpunkt. Der nächste Abschnitt beschreibt Anfragen über IPv6 unter Verwendung von Dual-Stack-Endpunkten.

Nachfolgend sind einige Dinge beschrieben, die Sie wissen sollten, bevor Sie versuchen, über IPv6 auf einen Bucket zuzugreifen.

- Der Client und das Netzwerk, die auf den Bucket zugreifen, müssen für IPv6 aktiviert sein.
- Für den IPv6-Zugriff werden Anfragen über virtuelle Hostings und über den PathStyle unterstützt. Weitere Informationen finden Sie unter Amazon S3-Dual-Stack-Endpunkte (p. 16).
- Wenn Sie eine IP-Quelladressen-Filterung in Ihren AWS Identity and Access Management (IAM)-Benutzer- oder -Bucket-Richtlinien verwenden, müssen Sie die Richtlinien aktualisieren, um IPv6-Adressbereiche zu berücksichtigen. Weitere Informationen finden Sie unter Verwendung von IPv6-Adressen in IAM-Richtlinien (p. 14).
- Bei Verwendung von IPv6 geben die Serverzugriff-Protokolldateien IP-Adressen in einem IPv6-Format aus. Sie müssen die Tools, Skripts und Softwareanwendungen aktualisieren, mit denen Sie Amazon S3-Protokolldateien analysieren, damit diese die `Remote IP`-Adressen im IPv6-Format analysieren können. Weitere Informationen finden Sie unter Amazon S3-Serverzugriff-Protokollformat (p. 784) und Amazon S3-Serverzugriffsprotokollierung (p. 777).

Note

Bei Problemen mit IPv6-Adressen in Protokolldateien nehmen Sie bitte Kontakt mit dem AWS-Support auf.

Anforderungen über IPv6 unter Verwendung von Dual-Stack-Endpunkten

Sie führen Anfragen mit Amazon S3-API-Aufrufen über IPv6 aus, indem Sie Dual-Stack-Endpunkte verwenden. Die Amazon S3-API-Operationen funktionieren stets gleich, unabhängig davon, ob Sie auf Amazon S3 über IPv6 oder IPv4 zugreifen. Die Leistung sollte ebenfalls dieselbe bleiben.

Wenn Sie die REST API verwenden, greifen Sie direkt auf einen Dual-Stack-Endpunkt zu. Weitere Informationen finden Sie unter Dual-Stack-Endpunkte (p. 16).

Wenn Sie die AWS Command Line Interface (AWS CLI) und AWS SDKs verwenden, können Sie einen Parameter oder ein Flag verwenden, um zu einem Dual-Stack-Endpunkt zu wechseln. Sie können den Dual-Stack-Endpunkt auch direkt als Override des Amazon S3-Endpunkts in der Konfigurationsdatei angeben.

Sie können einen Dual-Stack-Endpunkt verwenden, um über IPv6 auf einen Bucket zuzugreifen. Dazu können Sie Folgendes verwenden:

- Die AWS CLI; siehe Verwenden von Dual-Stack-Endpunkten von der AWS CLI (p. 17).
- Die AWS SDKs, siehe Verwenden von Dual-Stack-Endpunkte von den AWS SDKs (p. 17).
- Die REST API, siehe Senden von Anforderungen an Dual-Stack-Endpunkte unter Verwendung der REST-API (p. 48).

Funktionen, die über IPv6 nicht zur Verfügung stehen

Die folgenden Funktionen werden derzeit bei Zugriff auf einen S3-Bucket über IPv6 nicht unterstützt:

- Hosting statischer Websites in einem S3-Bucket
- BitTorrent

Verwendung von IPv6-Adressen in IAM-Richtlinien

Bevor Sie versuchen, über IPv6auf einen Bucket zuzugreifen, müssen Sie sicherstellen, dass alle IAM-Benutzer- oder S3-Bucket-Richtlinien, die für die IP-Adressfilterung verwendet werden, aktualisiert werden, um den IPv6-Adressbereich zu berücksichtigen. Wenn Richtlinien für die IP-Adressfilterung nicht für die Verarbeitung von IPv6-Adressen aktualisiert werden, verlieren oder erhalten Clients möglicherweise fälschlicherweise Zugriff auf den Bucket, wenn sie mit der Verwendung von IPv6 beginnen. Weitere Informationen zum Verwalten von Zugriffsberechtigungen mit IAM finden Sie unter Identity and Access Management in Amazon S3 (p. 329).

IAM-Richtlinien zum Filtern von IP-Adressen verwenden IP-Adressen-Bedingungsoperatoren. Die folgende Bucket-Richtlinie identifiziert den Bereich 54.240.143.* als Bereich zulässiger IPv4-Adressen durch Verwendung von Bedingungsoperatoren für IP-Adressen. Alle IP-Adressen außerhalb dieses Bereichs erhalten keinen Zugriff auf den Bucket (`examplebucket`). Da alle IPv6-Adressen außerhalb des zulässigen Bereichs liegen, verhindert diese Richtlinie, dass IPv6 auf `examplebucket` zugreifen können.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

Sie können das Element `Condition` der Bucket-Richtlinie ändern, um sowohl IPv4 (`54.240.143.0/24`)- als auch IPv6 (`2001:DB8:1234:5678::/64`)-Adressbereiche zuzulassen, wie im folgenden Beispiel gezeigt. Sie können denselben Typ von `Condition`-Block verwenden wie im Beispiel gezeigt, um Ihre IAM-Benutzer- und -Bucket-Richtlinien zu aktualisieren.

```
       "Condition": {
         "IpAddress": {
            "aws:SourceIp": [
              "54.240.143.0/24",
               "2001:DB8:1234:5678::/64"
             ]
          }
        }
```

Bevor Sie IPv6 verwenden, müssen Sie alle relevanten IAM-Benutzer- und -Bucket-Richtlinien aktualisieren, die eine IP-Adressfilterung verwenden, um IPv6-Adressbereiche zuzulassen. Sie sollten Ihre IAM-Richtlinien mit den IPv6-Adressbereichen Ihres Unternehmens aktualisieren, zusätzlich zu den vorhandenen IPv4-Adressbereichen. Ein Beispiel für eine Bucket-Richtlinie, die den Zugriff über IPv6 und IPv4 zulässt, finden Sie unter Beschränken des Zugriffs auf bestimmte IP-Adressen (p. 448).

Sie können Ihre IAM-Benutzerrichtlinien unter Verwendung der IAM-Konsole unter https://console.aws.amazon.com/iam/ überprüfen. Weitere Informationen zu IAM finden Sie im IAM-Benutzerhandbuch. Informationen zum Bearbeiten von S3-Bucket-Richtlinien finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Testen der IP-Adresskompatibilität

Wenn Sie Linux/Unix oder Mac OS X verwenden, können sie testen, ob Sie über IPv6 auf einen Dual-Stack-Endpunkt zugreifen können, indem Sie den Befehl `curl` wie im folgenden Beispiel gezeigt ausführen:

Example

```
curl -v  http://s3.dualstack.us-west-2.amazonaws.com/
```

Sie erhalten Informationen wie im folgenden Beispiel gezeigt zurück. Wenn Sie über IPv6 verbunden sind, wird es sich bei der IP-Adresse, mit der Sie die Verbindung herstellen, um eine IPv6-Adresse handeln.

```
* About to connect() to s3-us-west-2.amazonaws.com port 80 (#0)
*   Trying IPv6 address... connected
* Connected to s3.dualstack.us-west-2.amazonaws.com (IPv6 address) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1 OpenSSL/1.0.1t
 zlib/1.2.3
```

```
> Host: s3.dualstack.us-west-2.amazonaws.com
```

Wenn Sie Microsoft Windows 7 oder 10 verwenden, können sie testen, ob Sie über IPv6 oder IPv4 auf einen Dual-Stack-Endpunkt zugreifen können, indem Sie den Befehl `ping` wie im folgenden Beispiel gezeigt ausführen:

```
ping ipv6.s3.dualstack.us-west-2.amazonaws.com
```

Verwenden von Amazon S3-Dual-Stack-Endpunkten

Dual-Stack-Endpunkte in Amazon S3 unterstützen Anfragen an S3-Buckets über IPv6 und IPv4. In diesem Abschnitt wird die Verwendung von Dual-Stack-Endpunkten beschrieben.

Themen

Amazon S3-Dual-Stack-Endpunkte

Wenn Sie eine Anfrage an einen Dual-Stack-Endpunkt richten, wird die Bucket-URL in eine IPv6- oder eine IPv4-Adresse aufgelöst. Weitere Informationen zum Zugriff auf einen Bucket über IPv6 finden Sie unter Senden von Anforderungen an Amazon S3 über IPv6 (p. 13).

Wenn Sie die REST-API verwenden, können Sie direkt auf einen Amazon S3-Dual-Stack-Endpunkt zugreifen, indem Sie den Endpunktnamen (URI) verwenden. Sie können auf einen S3-Bucket über einen Dual-Stack-Endpunkt mit einem Endpunktnamen im Virtual-Hosting- oder Pfadformat zugreifen. Amazon S3 unterstützt ausschließlich regionsbasierte Namen von Dual-Stack-Endpunkten. Das bedeutet, dass Sie die Region als Teil des Namens angeben müssen.

Verwenden Sie die folgenden Namenskonventionenfür Endpunktnamen im Virtual-Hosted-Style und im Path-Style:

- Dual-Stack-Endpunkte im Virtual-Hosted–Style:

  bucketname.s3.dualstack.*aws-region*.amazonaws.com

- Dual-Stack-Endpunkt im Path–Style:

  s3.dualstack.*aws-region*.amazonaws.com/*bucketname*

Weitere Informationen zum Stil von Endpunktnamen finden Sie unter Zugriff auf einen Bucket (p. 60). Die Liste der Amazon S3-Endpunkte finden Sie unter Regionen und Endpunkte im AWS General Reference.

> **Important**
>
> Für Dual-Stack-Endpunkte kann eine Transfer Acceleration verwendet werden. Weitere Informationen finden Sie unter Erste Schritte mit Amazon S3 Transfer Acceleration (p. 76).

Wenn Sie die AWS Command Line Interface (AWS CLI) und AWS SDKs verwenden, können Sie einen Parameter oder ein Flag verwenden, um zu einem Dual-Stack-Endpunkt zu wechseln. Sie können den Dual-Stack-Endpunkt auch direkt als Override des Amazon S3-Endpunkts in der Konfigurationsdatei

angeben. In den folgenden Abschnitten wird erläutert, wie Dual-Stack-Endpunkte von der AWS CLI und den AWS SDKs verwendet werden.

Verwenden von Dual-Stack-Endpunkten von der AWS CLI

Dieser Abschnitt enthält Beispiele für AWS CLI-Befehle für Anfragen an einen Dual-Stack-Endpunkt. Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Einrichten der AWS-CLI (p. 808).

Sie legen den Konfigurationswert `use_dualstack_endpoint` auf `true` in einem Profil in Ihrer AWS Config-Datei fest, um alle Amazon S3-Anfragen über die Befehle `s3` und `s3api` der AWS CLI an den Dual-Stack-Endpunkt für die angegebene Region zu leiten. Sie geben die Region in der Konfigurationsdatei oder in einem Befehl mit der Option `--region` an.

Bei Verwendung von Dual-Stack-Endpunkten mit der AWS CLI werden sowohl das Adressenformat `path` als auch das Adressenformat `virtual` unterstützt. Der Adressierungsstil, der in der Konfigurationsdatei festgelegt wird, steuert, ob der Bucketname im Hostnamen enthalten oder Teil der URL ist. Standardmäßig versucht die CLI, den virtuellen Stil zu verwenden, wann immer das möglich ist, verwendet aber auch den Pfadstil, wenn das notwendig ist. Weitere Informationen finden Sie unter AWS CLI Amazon S3-Konfiguration.

Sie können auch Konfigurationsänderungen über einen Befehl vornehmen, wie im folgenden Beispiel gezeigt, das im Standardprofil `use_dualstack_endpoint` auf `true` und `addressing_style` auf `virtual` setzt.

```
$ aws configure set default.s3.use_dualstack_endpoint true
$ aws configure set default.s3.addressing_style virtual
```

Wenn Sie einen Dual-Stack-Endpunkt nur für bestimmte AWS CLI-Befehle (nicht für alle) verwenden wollen, können Sie eine der folgenden Methoden anwenden:

- Sie können den Dual-Stack-Endpunkt pro Befehl verwenden, indem Sie den Parameter `--endpoint-url` auf `https://s3.dualstack.`*`aws-region`*`.amazonaws.com` oder `http://s3.dualstack.`*`aws-region`*`.amazonaws.com` für jeden `s3`- oder `s3api`-Befehl setzen.

  ```
  $ aws s3api list-objects --bucket bucketname --endpoint-url https://s3.dualstack.aws-region.amazonaws.com
  ```

- Sie können separate Profile in Ihrer AWS Config-Datei einrichten. Legen Sie beispielsweise ein Profil an, das `use_dualstack_endpoint` auf `true` setzt, und ein Profil, das `use_dualstack_endpoint` nicht setzt. Wenn Sie einen Befehl ausführen, geben Sie an, welches Profil Sie verwenden wollen, abhängig davon, ob Sie den Dual-Stack-Endpunkt verwenden wollen oder nicht.

> **Note**
>
> Wenn Sie die AWS CLI verwenden, können Sie derzeit für Dual-Stack-Endpunkte keine Transfer Acceleration verwenden. Die AWS CLI wird jedoch demnächst unterstützt. Weitere Informationen finden Sie unter Verwendung von Transfer Acceleration mit AWS Command Line Interface (AWS CLI) (p. 78).

Verwenden von Dual-Stack-Endpunkte von den AWS SDKs

Dieser Abschnitt enthält Beispiele für den Zugriff auf einen Dual-Stack-Endpunkt unter Verwendung der AWS SDKs.

Beispiel für einen AWS SDK for Java-Dual-Stack-Endpunkt

Das folgende Beispiel zeigt die Aktivierung von Dual-Stack-Endpunkten beim Erstellen eines Amazon S3-Clients mittels AWS SDK for Java.

Anweisungen zum Erstellen und Testen eines funktionierenden Java-Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;

public class DualStackEndpoints {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            // Create an Amazon S3 client with dual-stack endpoints enabled.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .withDualstackEnabled(true)
                    .build();

            s3Client.listObjects(bucketName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Wenn Sie das AWS SDK for Java unter Windows einsetzen, müssen Sie möglicherweise die folgende JVM (Java Virtual Machine)-Eigenschaft festlegen:

```
java.net.preferIPv6Addresses=true
```

Beispiel für einen AWS .NET SDK Dual-Stack-Endpunkt

Wenn Sie das AWS SDK for .NET verwenden, verwenden die Klasse `AmazonS3Config`, um die Verwendung eines Dual-Stack-Endpunkts zu erlauben, wie im folgenden Beispiel gezeigt.

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DualStackEndpointTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
```

```
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            var config = new AmazonS3Config
            {
                UseDualstackEndpoint = true,
                RegionEndpoint = bucketRegion
            };
            client = new AmazonS3Client(config);
            Console.WriteLine("Listing objects stored in a bucket");
            ListingObjectsAsync().Wait();
        }

        private static async Task ListingObjectsAsync()
        {
            try
            {
                var request = new ListObjectsV2Request
                {
                    BucketName = bucketName,
                    MaxKeys = 10
                };
                ListObjectsV2Response response;
                do
                {
                    response = await client.ListObjectsV2Async(request);

                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }
                    Console.WriteLine("Next Continuation Token: {0}",
 response.NextContinuationToken);
                    request.ContinuationToken = response.NextContinuationToken;
                } while (response.IsTruncated == true);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Ein vollständiges .NET-Beispiel für die Auflistung von Objekten finden Sie unter Auflisten von Schlüsseln mit AWS SDK für .NET (p. 247).

Weitere Informationen zum Erstellen und Testen eines funktionierenden .NET-Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Verwenden von Dual-Stack-Endpunkte von der REST API

Weitere Informationen über Anfragen an Dual-Stack-Endpunkte über die REST API finden Sie unter Senden von Anforderungen an Dual-Stack-Endpunkte unter Verwendung der REST-API (p. 48).

Senden von Anfragen unter Verwendung der AWS SDKs

Themen

Sie können mit dem AWS SDK oder durch direkte REST API-Aufrufe innerhalb Ihrer Anwendung authentifizierte Anfragen an Amazon S3 senden. Die AWS SDK API verwendet die Anmeldeinformationen, die Sie für die Berechnung der Signatur für die Authentifizierung bereitstellen. Bei direkter Verwendung der REST-API in Ihren Anwendungen müssen Sie den benötigten Code zur Berechnung der Signatur für die Authentifizierung Ihrer Anfrage schreiben. Die Liste der verfügbaren AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Senden von Anfragen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen

Sie können Ihre AWS-Konto- oder IAM-Benutzeranmeldeinformationen verwenden, um authentifizierte Anfragen an Amazon S3 zu senden. In diesem Abschnitt werden Beispiele aufgeführt, wie Sie über AWS SDK for Java, AWS SDK für .NET und AWS SDK für PHP authentifizierte Anfragen senden können. Die Liste der verfügbaren AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Themen

Jedes dieser AWS SDKs verwendet eine SDK-spezifische Anmeldeinformationen-Anbieterkette, um die Anmeldeinformationen zu finden und zu verwenden und Aktionen für den Eigentümer der Anmeldeinformationen auszuführen. Was alle diese Anmeldeinformationen-Anbieterketten gemeinsam haben, ist, dass sie alle nach Ihrer lokalen Datei mit AWS-Anmeldeinformationen suchen.

Die einfachste Methode zum Konfigurieren von Anmeldeinformationen für Ihre AWS-SDKs ist die Verwendung einer AWS-Anmeldeinformationsdatei. Wenn Sie die AWS Command Line Interface (AWS CLI) verwenden, wurde möglicherweise bereits eine lokale Datei mit AWS-Anmeldeinformationen konfiguriert. Andernfalls gehen Sie wie folgt vor, um eine Anmeldeinformationsdatei einzurichten:

Erstellen einer lokalen AWS-Anmeldeinformationsdatei

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

2. Erstellen Sie einen neuen Benutzer mit Berechtigungen, die auf die Services und Aktionen beschränkt sind, auf die Ihr Code Zugriff hat. Weitere Informationen zum Erstellen eines neuen IAM-Benutzers finden Sie unter Erstellen von IAM-Benutzern (Konsole). Folgen Sie der Anleitung bis Schritt 8.
3. Wählen Sie Download .csv (CSV herunterladen) aus, um eine lokale Kopie Ihrer AWS-Anmeldeinformationen zu speichern.
4. Gehen Sie auf Ihrem Computer zu Ihrem Stammverzeichnis und erstellen Sie ein `.aws` Verzeichnis. Auf Unix-basierten Systemen (wie Linux oder OS X) befindet es sich an der folgenden Position:

   ```
   ~/.aws
   ```

   Unter Windows befindet es sich an der folgenden Position:

   ```
   %HOMEPATH%\.aws
   ```
5. Erstellen Sie in dem Verzeichnis `.aws` eine neue Datei namens `credentials`.
6. Öffnen Sie die .csv-Datei mit den Anmeldeinformationen aus, die Sie von der IAM-Konsole heruntergeladen haben, und kopieren Sie ihren Inhalt in die `credentials` Datei mit dem folgenden Format:

   ```
   [default]
   aws_access_key_id = your_access_key_id
   aws_secret_access_key = your_secret_access_key
   ```
7. Speichern Sie die `credentials` Datei, und löschen Sie die .csv-Datei, die Sie in Schritt 3 heruntergeladen haben.

Die gemeinsame Anmeldeinformationsdatei ist jetzt auf Ihrem lokalen Computer konfiguriert und kann für die AWS SDKs verwendet werden.

Senden von Anforderungen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen – AWS SDK for Java

Um unter Verwendung Ihrer AWS-Konto-Anmeldeinformationen oder IAM-Benutzeranmeldeinformationen authentifizierte Anfragen an Amazon S3 zu senden, führen Sie Folgendes aus:

- Erstellen Sie mit der `AmazonS3ClientBuilder`-Klasse eine `AmazonS3Client`-Instance.
- Führen Sie eine der `AmazonS3Client`-Methoden aus, um Anfragen an Amazon S3 zu senden. Der Client erstellt aus den von Ihnen angegebenen Anmeldeinformationen die erforderliche Signatur und nimmt sie in die Anfrage auf.

Das folgende Beispiel führt die vorhergehenden Aufgaben aus. Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsRequest;
import com.amazonaws.services.s3.model.ObjectListing;
```

```
import com.amazonaws.services.s3.model.S3ObjectSummary;

import java.io.IOException;
import java.util.List;

public class MakingRequests {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Get a list of objects in the bucket, two at a time, and
            // print the name and size of each object.
            ListObjectsRequest listRequest = new
 ListObjectsRequest().withBucketName(bucketName).withMaxKeys(2);
            ObjectListing objects = s3Client.listObjects(listRequest);
            while (true) {
                List<S3ObjectSummary> summaries = objects.getObjectSummaries();
                for (S3ObjectSummary summary : summaries) {
                    System.out.printf("Object \"%s\" retrieved with size %d\n",
 summary.getKey(), summary.getSize());
                }
                if (objects.isTruncated()) {
                    objects = s3Client.listNextBatchOfObjects(objects);
                } else {
                    break;
                }
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Senden von Anforderungen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen – AWS SDK für .NET

So senden Sie authentifizierte Anfragen unter Verwendung der Anmeldeinformationen Ihres AWS-Kontos oder eines IAM-Benutzers:

- Erstellen Sie eine Instance der `AmazonS3Client`-Klasse.
- Führen Sie eine der `AmazonS3Client`-Methoden aus, um Anfragen an Amazon S3 zu senden. Der Client erstellt aus den von Ihnen angegebenen Anmeldeinformationen die erforderliche Signatur und nimmt sie in die Anfrage auf, die er an Amazon S3 sendet.

Das folgende C#-Beispiel veranschaulicht, wie sie die vorhergehenden Aufgaben ausführen. Weitere Informationen zur Ausführung der .NET-Beispiele in diesem Handbuch sowie Anweisungen, wie Sie Ihre Anmeldeinformationen in einer Konfigurationsdatei speichern, finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class MakeS3RequestTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            using (client = new AmazonS3Client(bucketRegion))
            {
                Console.WriteLine("Listing objects stored in a bucket");
                ListingObjectsAsync().Wait();
            }
        }

        static async Task ListingObjectsAsync()
        {
            try
            {
                ListObjectsRequest request = new ListObjectsRequest
                {
                    BucketName = bucketName,
                    MaxKeys = 2
                };
                do
                {
                    ListObjectsResponse response = await client.ListObjectsAsync(request);
                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }

                    // If the response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.Marker = response.NextMarker;
                    }
                    else
                    {
                        request = null;
                    }
                } while (request != null);
            }
            catch (AmazonS3Exception e)
            {
```

```
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Note

Sie können den `AmazonS3Client`-Client erstellen, ohne Sicherheitsanmeldeinformationen anzugeben. Über diesen Client gesendete Anfragen sind anonym und enthalten keine Signatur. Amazon S3 gibt einen Fehler zurück, wenn Sie anonyme Anfragen für eine Ressource senden, die nicht öffentlich verfügbar ist.

Funktionierende Beispiele finden Sie unter Arbeiten mit Amazon S3-Objekten (p. 116) und Arbeiten mit Amazon S3-Buckets (p. 57). Sie können diese Beispiele unter Verwendung der Anmeldeinformationen Ihres AWS-Kontos oder eines IAM-Benutzers testen.

Um beispielsweise alle Objektschlüssel in Ihrem Bucket aufzulisten, lesen Sie unter Auflisten von Schlüsseln mit AWS SDK für .NET (p. 247) nach.

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Senden von Anforderungen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen – AWS SDK für PHP

In diesem Abschnitt wird die Verwendung einer Klasse aus Version 3 des AWS SDK für PHP zum Senden authentifizierter Anfragen unter Verwendung Ihrer AWS-Konto- oder IAM-Benutzeranmeldeinformationen beschrieben. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel veranschaulicht, wie der Client eine Anfrage unter Verwendung Ihrer Sicherheitsanmeldeinformationen durchführt, um alle Buckets für Ihr Konto aufzulisten.

Example

```
require 'vendor/autoload.php';

use Aws\Sts\StsClient;
use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
]);

// Retrieve the list of buckets.
```

```
$result = $s3->listBuckets();

try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // Print the list of objects to the page.
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

> **Note**
>
> Sie können den `S3Client`-Client erstellen, ohne Sicherheitsanmeldeinformationen anzugeben. Anfrage, die unter Verwendung dieses Clients gesendet werden, sind anonym und haben keine Signatur. Amazon S3 gibt einen Fehler zurück, wenn Sie anonyme Anfragen für eine Ressource stellen, die nicht öffentlich verfügbar ist.

Funktionierende Beispiele finden Sie unter Operationen für Objekte (p. 184). Sie können diese Beispiele unter Verwendung der Anmeldeinformationen Ihres AWS-Kontos oder eines IAM-Benutzers testen.

Ein Beispiel für die Auflistung der Objektschlüssel in einem Bucket finden Sie unter Auflisten von Schlüsseln mit AWS SDK für PHP (p. 249).

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Senden von Anfragen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen – AWS SDK für Ruby

Bevor Sie Version 3 des AWS SDK für Ruby verwenden können, um Aufrufe an Amazon S3 zu senden, müssen Sie die AWS-Zugriffsinformationen festlegen, die das SDK zur Verifizierung Ihres Zugriffs auf Ihre Buckets und Objekte verwendet. Wenn Sie freigegebene Anmeldeinformationen im AWS-Anmeldeinformationenprofil auf Ihrem lokalen System bereitgestellt haben, kann Version 3 von SDK für Ruby diese Anmeldeinformationen verwenden, ohne dass Sie in Ihrem Code deklariert haben müssen. Weitere Informationen zum Einrichten freigegebener Anmeldeinformationen finden Sie unter Senden von Anfragen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen (p. 20).

Der folgende Ruby-Codeausschnitt verwendet die Anmeldeinformationen in einer freigegebenen AWS-Anmeldeinformationsdatei auf einem lokalen Computer zum Authentifizieren einer Anfrage, um alle Objektschlüsselnamen in einem bestimmten Bucket anzufordern. Es führt die folgenden Aktionen aus:

1. Erstellt eine Instance der `Aws::S3::Resource`-Klasse.
2. Sendet eine Anfrage an Amazon S3 durch Auflisten der Objekte in einem Bucket mittels der Methode `bucket` von `Aws::S3::Resource`. Der Client erstellt aus den Anmeldeinformationen der AWS-Anmeldeinformationsdatei auf Ihrem Computer den erforderlichen Signaturwert und nimmt ihn in die Anfrage auf, die er an Amazon S3 sendet.

3. Gibt das Array der Objektschlüsselnamen an das Terminal aus.

Example

```
# This snippet example does the following:
# Creates an instance of the Aws::S3::Resource class.
# Makes a request to Amazon S3 by enumerating objects in a bucket using the bucket method
 of Aws::S3::Resource.
# The client generates the necessary signature value from the credentials in the AWS
 credentials file on your computer,
# and includes it in the request it sends to Amazon S3.
# Prints the array of object key names to the terminal.
# The credentials that are used for this example come from a local AWS credentials file on
 the computer that is running this application.
# The credentials are for an IAM user who can list objects in the bucket that the user
 specifies when they run the application.

# Use the Amazon S3 modularized gem for version 3 of the AWS Ruby SDK.
require 'aws-sdk-s3'

# Get an Amazon S3 resource.
s3 = Aws::S3::Resource.new(region: 'us-west-2')

# Create an array of up to the first 100 object keynames in the bucket.
bucket = s3.bucket('example_bucket').objects.collect(&:key)

# Print the array to the terminal.
puts bucket
```

Wenn Sie keine lokale Datei mit AWS-Anmeldeinformationen besitzen, können Sie dennoch die `Aws::S3::Resource`-Ressource erstellen und Code für Amazon S3-Buckets und -Objekte ausführen. Über Version 3 des SDK für Ruby gesendete Anfragen sind anonym und enthalten standardmäßig keine Signatur. Amazon S3 gibt einen Fehler zurück, wenn Sie anonyme Anfragen für eine Ressource senden, die nicht öffentlich verfügbar ist.

Sie können den obigen Codeausschnitt für SDK für Ruby-Anwendungen verwenden und erweitern, wie im folgenden, robusteren Beispiel gezeigt. Die für dieses Beispiel verwendeten Anmeldeinformationen stammen aus einer lokalen AWS-Anmeldeinformationsdatei auf dem Computer, auf dem diese Anwendung ausgeführt wird. Die Anmeldeinformationen sind für einen IAM-Benutzer vorgesehen, der Objekte im Bucket auflisten kann, die der Benutzer bei der Ausführung der Anwendung angibt.

```
# This snippet example does the following:
# Creates an instance of the Aws::S3::Resource class.
# Makes a request to Amazon S3 by enumerating objects in a bucket using the bucket method
 of Aws::S3::Resource.
# The client generates the necessary signature value from the credentials in the AWS
 credentials file on your computer,
# and includes it in the request it sends to Amazon S3.
# Prints the array of object key names to the terminal.
# The credentials that are used for this example come from a local AWS credentials file on
 the computer that is running this application.
# The credentials are for an IAM user who can list objects in the bucket that the user
 specifies when they run the application.

# Use the Amazon S3 modularized gem for version 3 of the AWS Ruby SDK.
require 'aws-sdk-s3'

# Usage: ruby auth_request_test.rb OPERATION BUCKET
# Currently only the list operation is supported

# The operation to perform on the bucket.
operation = 'list' # default
```

```
operation = ARGV[0] if (ARGV.length > 0)

if ARGV.length > 1
  bucket_name = ARGV[1]
else
  exit 1
end

# Get an Amazon S3 resource.
s3 = Aws::S3::Resource.new(region: 'us-west-2')

# Get the bucket by name.
bucket = s3.bucket(bucket_name)

case operation

when 'list'
  if bucket.exists?
    # Enumerate the bucket contents and object etags.
    puts "Contents of '%s':" % bucket_name
    puts '  Name => GUID'

    bucket.objects.limit(50).each do |obj|
      puts "  #{obj.key} => #{obj.etag}"
    end
  else
    puts "The bucket '%s' does not exist!" % bucket_name
  end

else
  puts "Unknown operation: '%s'! Only list is supported." % operation
end
```

Senden von Anforderungen unter Verwendung temporärer IAM-Benutzeranmeldeinformationen

Themen

AWS-Konten oder IAM-Benutzer können temporäre Sicherheitsanmeldeinformationen anfordern und diese zum Senden authentifizierter Anfragen an Amazon S3 verwenden. Dieser Abschnitt enthält Beispiele für die Verwendung von AWS SDK for Java, .NET, und PHP, um temporäre Sicherheitsanmeldeinformationen für die Authentifizierung Ihrer Anfragen an Amazon S3 zu erhalten.

Senden von Anforderungen unter Verwendung temporärer IAM-Benutzeranmeldeinformationen – AWS SDK for Java

IAM-Benutzer oder AWS-Konten können über das AWS SDK for Java temporäre Sicherheitsanmeldeinformationen anfordern (siehe Senden von Anforderungen (p. 11)) und diese für den Zugriff auf Amazon S3 verwenden. Diese Anmeldeinformationen laufen nach der angegebenen

Sitzungsdauer ab. Geben Sie zum Verwenden der temporären IAM-Sicherheitsanmeldeinformationen wie folgt vor:

1. Erstellen Sie eine Instance der `AWSSecurityTokenService`-Klasse. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).
2. Rufen Sie die temporären Sicherheitsanmeldeinformationen für die gewünschte Rolle ab, indem Sie die `assumeRole()`-Methode des Security Token Service (STS) -Clients aufrufen.
3. Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einem `BasicSessionCredentials`-Objekt. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon S3-Client bereitzustellen.
4. Erstellen Sie mit den temporären Sicherheitsanmeldeinformationen eine Instance der `AmazonS3Client`-Klasse. Mit diesem Client senden Sie Anfragen an Amazon S3. Wenn Sie beim Senden von Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

   Note

   Wenn Sie unter Verwendung der Sicherheitsanmeldeinformationen Ihres AWS-Kontos temporäre Sicherheitsanmeldeinformationen erhalten haben, sind sie nur eine Stunde lang gültig. Sie können eine Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzeranmeldeinformationen verwenden, um eine Sitzung anzufordern.

Das folgende Beispiel listet einen Satz von Objektschlüsseln im angegebenen Bucket auf. Das Beispiel fordert temporäre Sicherheitsanmeldeinformationen für eine Sitzung an, die dann zum Senden einer authentifizierten Anfrage an Amazon S3 verwendet werden.

Wenn Sie das Beispiel für die Verwendung von IAM-Benutzeranmeldeinformationen testen möchten, müssen Sie in Ihrem AWS-Konto einen IAM-Benutzer erstellen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.AssumeRoleRequest;
import com.amazonaws.services.securitytoken.model.AssumeRoleResult;
import com.amazonaws.services.securitytoken.model.Credentials;

public class MakingRequestsWithIAMTempCredentials {
    public static void main(String[] args) {
        String clientRegion = "*** Client region ***";
        String roleARN = "*** ARN for role to be assumed ***";
        String roleSessionName = "*** Role session name ***";
        String bucketName = "*** Bucket name ***";

        try {
            // Creating the STS client is part of your trusted code. It has
```

```
            // the security credentials you use to obtain temporary security credentials.
            AWSSecurityTokenService stsClient =
AWSSecurityTokenServiceClientBuilder.standard()
                                                    .withCredentials(new
ProfileCredentialsProvider())
                                                    .withRegion(clientRegion)
                                                    .build();

            // Obtain credentials for the IAM role. Note that you cannot assume the role of
an AWS root account;
            // Amazon S3 will deny access. You must use credentials for an IAM user or an
IAM role.
            AssumeRoleRequest roleRequest = new AssumeRoleRequest()
                                                    .withRoleArn(roleARN)
                                                    .withRoleSessionName(roleSessionName);
            AssumeRoleResult roleResponse = stsClient.assumeRole(roleRequest);
            Credentials sessionCredentials = roleResponse.getCredentials();

            // Create a BasicSessionCredentials object that contains the credentials you
just retrieved.
            BasicSessionCredentials awsCredentials = new BasicSessionCredentials(
                    sessionCredentials.getAccessKeyId(),
                    sessionCredentials.getSecretAccessKey(),
                    sessionCredentials.getSessionToken());

            // Provide temporary security credentials so that the Amazon S3 client
    // can send authenticated requests to Amazon S3. You create the client
    // using the sessionCredentials object.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                                    .withCredentials(new
AWSStaticCredentialsProvider(awsCredentials))
                                    .withRegion(clientRegion)
                                    .build();

            // Verify that assuming the role worked and the permissions are set correctly
            // by getting a set of object keys from the bucket.
            ObjectListing objects = s3Client.listObjects(bucketName);
            System.out.println("No. of Objects: " + objects.getObjectSummaries().size());
        }
        catch(AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch(SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Senden von Anforderungen unter Verwendung temporärer IAM-Benutzeranmeldeinformationen – AWS SDK für .NET

IAM-Benutzer oder AWS-Konten können über das AWS SDK für .NET temporäre Sicherheitsanmeldeinformationen anfordern und diese für den Zugriff auf Amazon S3 verwenden.

Diese Anmeldeinformationen laufen nach Ende der Sitzung ab. Gehen Sie zum Abruf temporärer Sicherheitsanmeldeinformationen und zum Zugriff auf Amazon S3 wie folgt vor:

1. Erstellen Sie eine Instance des AWS Security Token Service-Clients, `AmazonSecurityTokenServiceClient`. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).
2. Starten Sie eine Sitzung durch Aufruf der Methode `GetSessionToken` des STS-Clients, den Sie im vorigen Schritt erstellt haben. Sie stellen für diese Methode Sitzungsdaten mithilfe eines `GetSessionTokenRequest`-Objekts bereit.

   Die Methode gibt Ihre temporären Sicherheitsanmeldeinformationen zurück.
3. Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einer Instance des Objekts `SessionAWSCredentials`. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon S3-Client bereitzustellen.
4. Erstellen Sie eine Instance der Klasse `AmazonS3Client`, indem Sie die temporären Sicherheitsanmeldeinformationen übergeben. Mit diesem Client senden Sie Anfragen an Amazon S3. Wenn Sie beim Senden der Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

   Note

   Wenn Sie unter Verwendung der Sicherheitsanmeldeinformationen Ihres AWS-Kontos temporäre Sicherheitsanmeldeinformationen erhalten haben, sind diese Anmeldeinformationen nur eine Stunde lang gültig. Sie können die Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzeranmeldeinformationen verwenden, um eine Sitzung anzufordern.

Das folgende C#-Beispiel listet Objektschlüssel im angegebenen Bucket auf. Zur Veranschaulichung fordert das Beispiel temporäre Sicherheitsanmeldeinformationen für die standardmäßige, einstündige Sitzung an, die dann zum Senden einer authentifizierten Anfrage an Amazon S3 verwendet werden.

Wenn Sie das Beispiel für die Verwendung von IAM-Benutzeranmeldeinformationen testen möchten, müssen Sie in Ihrem AWS-Konto einen IAM-Benutzer erstellen. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch. Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen (p. 11).

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempCredExplicitSessionStartTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
```

```
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
            {
                // Credentials use the default AWS SDK for .NET credential search chain.
                // On local development machines, this is your default profile.
                Console.WriteLine("Listing objects stored in a bucket");
                SessionAWSCredentials tempCredentials = await
GetTemporaryCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (s3Client = new AmazonS3Client(tempCredentials, bucketRegion))
                {
                    var listObjectRequest = new ListObjectsRequest
                    {
                        BucketName = bucketName
                    };
                    // Send request to Amazon S3.
                    ListObjectsResponse response = await
s3Client.ListObjectsAsync(listObjectRequest);
                    List<S3Object> objects = response.S3Objects;
                    Console.WriteLine("Object count = {0}", objects.Count);
                }
            }
            catch (AmazonS3Exception s3Exception)
            {
                Console.WriteLine(s3Exception.Message, s3Exception.InnerException);
            }
            catch (AmazonSecurityTokenServiceException stsException)
            {
                Console.WriteLine(stsException.Message, stsException.InnerException);
            }
        }

        private static async Task<SessionAWSCredentials> GetTemporaryCredentialsAsync()
        {
            using (var stsClient = new AmazonSecurityTokenServiceClient())
            {
                var getSessionTokenRequest = new GetSessionTokenRequest
                {
                    DurationSeconds = 7200 // seconds
                };

                GetSessionTokenResponse sessionTokenResponse =
                              await stsClient.GetSessionTokenAsync(getSessionTokenRequest);

                Credentials credentials = sessionTokenResponse.Credentials;

                var sessionCredentials =
                    new SessionAWSCredentials(credentials.AccessKeyId,
                                              credentials.SecretAccessKey,
                                              credentials.SessionToken);
                return sessionCredentials;
            }
        }
    }
}
```

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Senden von Anforderungen unter Verwendung temporärer AWS-Konto- oder IAM-Benutzeranmeldeinformationen – AWS SDK für PHP

In diesem Thema wird die Verwendung von Klassen aus Version 3 des AWS SDK für PHP beschrieben, um temporäre Sicherheitsanmeldeinformationen anzufordern und sie für den Zugriff auf Amazon S3 zu verwenden. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

IAM-Benutzer oder AWS-Konten können unter Verwendung von Version 3 des AWS SDK für PHP temporäre Sicherheitsanmeldeinformationen anfordern. Er kann mit den temporären Benutzeranforderungen dann auf Amazon S3 zugreifen. Die Anmeldeinformationen laufen mit dem Ende der Sitzung ab. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie IAM-Anmeldeinformationen verwenden, können Sie beim Anfordern der temporären Sicherheitsanmeldeinformationen die Dauer (von 1 bis 36 Stunden) angeben. Weitere Informationen zu temporären Sicherheitsanmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im . Weitere Informationen zum Senden von Anfragen finden Sie unter Senden von Anforderungen (p. 11).

> Note
>
> Wenn Sie unter Verwendung Ihrer AWS-Kontoanmeldeinformationen temporäre Sicherheitsanmeldeinformationen erhalten haben, sind sie lediglich eine Stunde lang gültig. Sie können eine Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzeranmeldeinformationen verwenden, um eine Sitzung anzufordern.

Example

Das folgende PHP-Beispiel listet die Objektschlüssel im angegebenen Bucket bei Verwendung temporärer Sicherheitsanmeldeinformationen auf. Das Beispiel fordert temporäre Sicherheitsanmeldeinformationen für eine standardmäßige, einstündige Sitzung an, die dann zum Senden einer authentifizierten Anfrage an Amazon S3 verwendet werden. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

Wenn Sie das Beispiel für die Verwendung von IAM-Benutzeranmeldeinformationen testen möchten, müssen Sie in Ihrem AWS-Konto einen IAM-Benutzer erstellen. Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch. Ein Beispiel für das Festlegen der Sitzungsdauer, wenn Sie IAM-Benutzeranmeldeinformationen zum Anfordern einer Sitzung verwenden, finden Sie unter Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer – AWS SDK für PHP (p. 42).

```
 require 'vendor/autoload.php';

use Aws\Sts\StsClient;
use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';

$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
]);

$sessionToken = $sts->getSessionToken();

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
```

```
    'credentials' => [
        'key'    => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token'  => $sessionToken['Credentials']['SessionToken']
    ]
]);

$result = $s3->listBuckets();


try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // List objects
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Senden von Anfragen unter Verwendung temporärer IAM-Benutzeranmeldeinformationen – AWS SDK für Ruby

IAM-Benutzer oder AWS-Konten können über das AWS SDK für Ruby temporäre Sicherheitsanmeldeinformationen anfordern und diese für den Zugriff auf Amazon S3 verwenden. Diese Anmeldeinformationen laufen nach Ende der Sitzung ab. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie IAM-Anmeldeinformationen verwenden, können Sie beim Anfordern der temporären Sicherheitsanmeldeinformationen die Dauer (von 1 bis 36 Stunden) angeben. Weitere Informationen zum Anfordern von temporären Sicherheitsanmeldeinformationen finden Sie unter Senden von Anforderungen (p. 11).

> **Note**
>
> Wenn Sie unter Verwendung Ihrer AWS-Kontoanmeldeinformationen temporäre Sicherheitsanmeldeinformationen erhalten haben, sind sie lediglich eine Stunde lang gültig. Sie können die Sitzungsdauer nur dann festlegen, wenn Sie IAM-Benutzeranmeldeinformationen verwenden, um eine Sitzung anzufordern.

Das folgende Ruby-Beispiel erstellt einen temporären Benutzer, damit dieser für eine Stunde die Elemente in einem bestimmten Bucket auflisten kann. Um dieses Beispiel verwenden zu können, müssen Sie über AWS-Anmeldeinformationen mit den notwendigen Berechtigungen zum Erstellen neuer AWS Security Token Service (AWS STS)-Clients und zum Auflisten von Amazon S3-Buckets verfügen.

```
# This snippet example does the following:
# The following Ruby example creates a temporary user to list the items in a specified
 bucket
# for one hour. To use this example, you must have AWS credentials that have the necessary
# permissions to create new AWS Security Token Service (AWS STS) clients, and list Amazon
 S3 buckets using temporary security credentials
```

```
# using your AWS account security credentials, the temporary security credentials are valid
 for only one hour. You can
# specify session duration only if you use &IAM; user credentials to request a session.

require 'aws-sdk-core'
require 'aws-sdk-s3'
require 'aws-sdk-iam'

USAGE = <<DOC

Usage: assumerole_create_bucket_policy.rb -b BUCKET -u USER [-r REGION] [-d] [-h]

  Assumes a role for USER to list items in BUCKET for one hour.

  BUCKET is required and must already exist.

  USER is required and if not found, is created.

  If REGION is not supplied, defaults to us-west-2.

  -d gives you extra (debugging) information.

  -h displays this message and quits.

DOC

def print_debug(debug, s)
  if debug
    puts s
  end
end

# Get the user if they exist, otherwise create them
def get_user(region, user_name, debug)
  iam = Aws::IAM::Resource.new(region: region)

  # See if user exists
  user = iam.user(user_name)

  # If user does not exist, create them
  if user == nil
    user = iam.create_user(user_name: user_name)
    iam.wait_until(:user_exists, user_name: user_name)
    print_debug(debug, "Created new user #{user_name}")
  else
    print_debug(debug, "Found user #{user_name} in region #{region}")
  end

  user
end

# main
region = 'us-west-2'
user_name = ''
bucket_name = ''

i = 0

while i &lt; ARGV.length
  case ARGV[i]

    when '-b'
      i += 1
      bucket_name = ARGV[i]

    when '-u'
```

```
      i += 1
      user_name = ARGV[i]

    when '-r'
      i += 1

      region = ARGV[i]

    when '-h'
      puts USAGE
      exit 0

    else
      puts 'Unrecognized option: ' + ARGV[i]
      puts USAGE
      exit 1

  end

  i += 1
end

if bucket_name == ''
  puts 'You must supply a bucket name'
  puts USAGE
  exit 1
end

if user_name == ''
  puts 'You must supply a user name'
  puts USAGE
  exit 1
end

# Create a new Amazon STS client and get temporary credentials. This uses a role that was
 already created.
begin
  creds = Aws::AssumeRoleCredentials.new(
    client: Aws::STS::Client.new(region: region),
    role_arn: "arn:aws:iam::111122223333:role/assumedrolelist",
    role_session_name: "assumerole-s3-list"
  )

  # Create an Amazon S3 resource with temporary credentials.
  s3 = Aws::S3::Resource.new(region: region, credentials: creds)

  puts "Contents of '%s':" % bucket_name
  puts '  Name => GUID'

  s3.bucket(bucket_name).objects.limit(50).each do |obj|
    puts "  #{obj.key} => #{obj.etag}"
  end
rescue StandardError => ex
  puts 'Caught exception accessing bucket ' + bucket_name + ':'
  puts ex.message
end
```

Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer

Sie können temporäre Sicherheitsanmeldeinformationen anfordern und Sie für Ihre verbundenen Benutzer oder Anwendungen bereitstellen, die Zugriff auf Ihrer AWS-Ressourcen benötigen.

Dieser Abschnitt gibt Beispiele dafür, wie Sie das AWS-SDK dazu verwenden können, temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer oder Anwendungen zu erlangen und mithilfe dieser Anmeldeinformationen authentifizierte Anfragen an Amazon S3 zu senden. Die Liste der verfügbaren AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

> Note
>
> Sowohl das AWS-Konto als auch ein IAM-Benutzer können temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer anfordern. Um die Sicherheit zu verbessern, sollten ausschließlich IAM-Benutzer mit den notwendigen Berechtigungen diese temporären Anmeldeinformationen anfordern, um sicherzustellen, dass der verbundene Benutzer höchstens die Berechtigungen des jeweils anfragenden IAM-Benutzers erhält. In einigen Anwendungen kann es nützlich sein, einen IAM-Benutzer mit den spezifischen Berechtigungen nur zu dem Zweck zu erstellen, temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen zu erteilen.

Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer – AWS SDK for Java

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben wird, die Sie erteilen möchten. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Sie können explizit einen anderen Wert für die Dauer angeben, wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern.

> Note
>
> Zur Gewährleistung zusätzlicher Sicherheit beim Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen empfehlen wir die Verwendung eines dedizierten IAM-Benutzers mit nur den dazu benötigten Zugriffsberechtigungen. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen finden Sie unter AWS Identity and Access Management – Häufig gestellte Fragen.

Gehen Sie zum Bereitstellen von Sicherheitsanmeldeinformationen und zum Senden einer authentifizierten Anfrage für den Zugriff auf Ressourcen wie folgt vor:

- Erstellen Sie eine Instance der `AWSSecurityTokenServiceClient`-Klasse. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Verwendung von AWS SDK for Java (p. 809).
- Starten Sie mit dem Aufruf der Methode `getFederationToken()` des Security Token Service (STS)-Clients eine Sitzung. Stellen Sie Sitzungsinformationen, wie beispielsweise den Benutzernamen und eine IAM-Richtlinie, bereit, die Sie den temporären Anmeldeinformationen zuordnen möchten. Optional können Sie eine Sitzungsdauer angeben. Diese Methode gibt Ihre temporären Sicherheitsanmeldeinformationen zurück.
- Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einer Instance des Objekts `BasicSessionCredentials`. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon S3-Client bereitzustellen.
- Erstellen Sie mit den temporären Sicherheitsanmeldeinformationen eine Instance der `AmazonS3Client`-Klasse. Mit diesem Client senden Sie Anfragen an Amazon S3. Wenn Sie beim Senden der Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

Example

Das folgende Beispiel listet die Schlüssel im angegebenen S3-Bucket auf. In dem Beispiel erhalten Sie temporäre Sicherheitsanmeldeinformationen für eine zweistündige Sitzung für Ihren verbundenen Benutzer und verwenden sie, um authentifizierte Anfragen an Amazon S3 zu senden. Zum Ausführen des Beispiels müssen Sie einen IAM-Benutzer mit einer zugeordneten Richtlinie erstellen, die es dem Benutzer ermöglicht, temporäre Sicherheitsanmeldeinformationen anzufordern und Ihre AWS-Ressources aufzulisten. Dies wird mit der folgenden Richtlinie erreicht:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
}
```

Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Nachdem ein IAM-Benutzer erstellt und die vorhergehende Richtlinie angefügt wurde, können Sie das folgende Beispiel ausführen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.S3Actions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;

import java.io.IOException;

public class MakingRequestsWithFederatedTempCredentials {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Specify bucket name ***";
        String federatedUser = "*** Federated user name ***";
        String resourceARN = "arn:aws:s3:::" + bucketName;

        try {
            AWSSecurityTokenService stsClient = AWSSecurityTokenServiceClientBuilder
                    .standard()
                    .withCredentials(new ProfileCredentialsProvider())
```

```
                    .withRegion(clientRegion)
                    .build();

            GetFederationTokenRequest getFederationTokenRequest = new
 GetFederationTokenRequest();
            getFederationTokenRequest.setDurationSeconds(7200);
            getFederationTokenRequest.setName(federatedUser);

            // Define the policy and add it to the request.
            Policy policy = new Policy();
            policy.withStatements(new Statement(Effect.Allow)
                    .withActions(S3Actions.ListObjects)
                    .withResources(new Resource(resourceARN)));
            getFederationTokenRequest.setPolicy(policy.toJson());

            // Get the temporary security credentials.
            GetFederationTokenResult federationTokenResult =
 stsClient.getFederationToken(getFederationTokenRequest);
            Credentials sessionCredentials = federationTokenResult.getCredentials();

            // Package the session credentials as a BasicSessionCredentials
            // object for an Amazon S3 client object to use.
            BasicSessionCredentials basicSessionCredentials = new BasicSessionCredentials(
                    sessionCredentials.getAccessKeyId(),
                    sessionCredentials.getSecretAccessKey(),
                    sessionCredentials.getSessionToken());
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new
 AWSStaticCredentialsProvider(basicSessionCredentials))
                    .withRegion(clientRegion)
                    .build();

            // To verify that the client works, send a listObjects request using
            // the temporary security credentials.
            ObjectListing objects = s3Client.listObjects(bucketName);
            System.out.println("No. of Objects = " + objects.getObjectSummaries().size());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer – AWS SDK für .NET

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben wird, die Sie erteilen

möchten. Standardmäßig beträgt die Dauer einer Sitzung eine Stunde. Sie können explizit einen anderen Wert für die Dauer angeben, wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern. Informationen zum Senden authentifizierter Anfragen finden Sie unter Senden von Anforderungen (p. 11).

> **Note**
>
> Wenn temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen angefordert werden, empfehlen wir, für zusätzliche Sicherheit einen dedizierten IAM-Benutzer mit nur den dazu benötigten Zugriffsberechtigungen zu verwenden. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen finden Sie unter AWS Identity and Access Management – Häufig gestellte Fragen.

Dazu führen Sie die folgenden Schritte aus:

- Sie erstellen eine Instance des AWS Security Token Service-Clients, Klasse `AmazonSecurityTokenServiceClient`. Weitere Informationen zum Bereitstellen von Anmeldeinformationen finden Sie unter Verwendung von AWS SDK für .NET (p. 810).
- Starten Sie eine Sitzung durch Aufruf der Methode `GetFederationToken` des STS-Clients. Sie müssen Sitzungsinformationen bereitstellen, wie beispielsweise den Benutzernamen und eine IAM-Richtlinie, die Sie den temporären Anmeldeinformationen zuordnen wollen. Optional können Sie eine Sitzungsdauer angeben. Diese Methode gibt Ihre temporären Sicherheitsanmeldeinformationen zurück.
- Bündeln Sie die temporären Sicherheitsanmeldeinformationen in einer Instance des Objekts `SessionAWSCredentials`. Sie verwenden dieses Objekt, um die temporären Sicherheitsanmeldeinformationen für Ihren Amazon S3-Client bereitzustellen.
- Erstellen Sie eine Instance der `AmazonS3Client`-Klasse, indem Sie die temporären Sicherheitsanmeldeinformationen übergeben. Sie verwenden diesen Client zum Senden von Anfragen an Amazon S3. Wenn Sie beim Senden der Anfragen abgelaufene Anmeldeinformationen verwenden, gibt Amazon S3 einen Fehler zurück.

Example

Das folgende C#-Beispiel listet die Schlüssel im angegebenen Bucket auf. In dem Beispiel erhalten Sie temporäre Sicherheitsanmeldeinformationen für eine zweistündige Sitzung für Ihren verbundenen Benutzer (User1) und verwenden sie, um authentifizierte Anfragen an Amazon S3 zu senden.

- In dieser Übung erstellen Sie einen IAM-Benutzer mit minimalen Berechtigungen. Unter Verwendung der Anmeldeinformationen dieses IAM-Benutzers fordern Sie temporäre Anmeldeinformationen für andere an. Dieses Beispiel listet nur die Objekte in einem bestimmgen Bucket auf. Erstellen Sie einen IAM-Benutzer, dem die folgenden Richtlinie zugeordnet ist:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
}
```

  Die Richtlinie gestattet dem IAM-Benutzer, temporäre Sicherheitsanmeldeinformationen anzufordern, und erteilt ihm nur Zugriffsberechtigungen, um Ihre AWS-Ressourcen aufzulisten. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

- Verwenden Sie die Sicherheitsanmeldeinformationen des IAM-Benutzers, um das folgende Beispiel zu testen. Das Beispiel sendet unter Verwendung temporärer Sicherheitsanmeldeinformationen eine authentifizierte Anfrage an Amazon S3. Das Beispiel gibt die folgende Richtlinie an, wenn temporäre Sicherheitsanmeldeinformationen für den verbundenen Benutzer (User1) angefordert werden, womit der Zugriff auf eine Auflistung der Objekte in einem spezifischen Bucket eingeschränkt wird (`YourBucketName`). Sie müssen die Richtlinie aktualisieren, indem Sie Ihren eigenen bestehenden Bucket-Namen angeben.

```
{
  "Statement":[
    {
      "Sid":"1",
      "Action":["s3:ListBucket"],
      "Effect":"Allow",
      "Resource":"arn:aws:s3:::YourBucketName"
    }
  ]
}
```

- Example

  Aktualisieren Sie das folgende Beispiel und geben Sie den Bucket-Namen an, den Sie in der vorherigen Zugriffsrichtlinie für verbundene Benutzer angegeben haben. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempFederatedCredentialsTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
            {
                Console.WriteLine("Listing objects stored in a bucket");
                // Credentials use the default AWS SDK for .NET credential search chain.
                // On local development machines, this is your default profile.
                SessionAWSCredentials tempCredentials =
                    await GetTemporaryFederatedCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (client = new AmazonS3Client(bucketRegion))
                {
```

```
                    ListObjectsRequest listObjectRequest = new ListObjectsRequest();
                    listObjectRequest.BucketName = bucketName;

                    ListObjectsResponse response = await
client.ListObjectsAsync(listObjectRequest);
                    List<S3Object> objects = response.S3Objects;
                    Console.WriteLine("Object count = {0}", objects.Count);

                    Console.WriteLine("Press any key to continue...");
                    Console.ReadKey();
                }
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing an
object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        private static async Task<SessionAWSCredentials>
GetTemporaryFederatedCredentialsAsync()
        {
            AmazonSecurityTokenServiceConfig config = new
AmazonSecurityTokenServiceConfig();
            AmazonSecurityTokenServiceClient stsClient =
                new AmazonSecurityTokenServiceClient(
                                            config);

            GetFederationTokenRequest federationTokenRequest =
                                    new GetFederationTokenRequest();
            federationTokenRequest.DurationSeconds = 7200;
            federationTokenRequest.Name = "User1";
            federationTokenRequest.Policy = @"{
               ""Statement"":
               [
                 {
                   ""Sid"":""Stmt1311212314284"",
                   ""Action"":[""s3:ListBucket""],
                   ""Effect"":""Allow"",
                   ""Resource"":""arn:aws:s3:::" + bucketName + @"""
                  }
               ]
             }
            ";

            GetFederationTokenResponse federationTokenResponse =
                        await stsClient.GetFederationTokenAsync(federationTokenRequest);
            Credentials credentials = federationTokenResponse.Credentials;

            SessionAWSCredentials sessionCredentials =
                new SessionAWSCredentials(credentials.AccessKeyId,
                                          credentials.SecretAccessKey,
                                          credentials.SessionToken);
            return sessionCredentials;
        }
    }
}
```

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer – AWS SDK für PHP

In diesem Thema wird die Verwendung von Klassen aus Version 3 des AWS SDK für PHP zum Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen und ihre Verwendung für den Zugriff auf in Amazon S3 gespeicherte Ressourcen beschrieben. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie diese temporären Anmeldeinformationen anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, mit der die Ressourcenberechtigungen beschrieben wird, die Sie erteilen möchten. Diese Anmeldeinformationen laufen mit dem Ende der Sitzung ab. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Sie können explizit einen anderen Wert für die Dauer angeben, wenn Sie temporäre Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern. Weitere Informationen zu temporären Sicherheitsanmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch. Informationen zum Bereitstellen temporärer Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen finden Sie unter Senden von Anforderungen (p. 11).

Für zusätzliche Sicherheit beim Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen empfehlen wir die Verwendung eines dedizierten IAM-Benutzers mit nur den dazu benötigten Zugriffsberechtigungen. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen zum Identitätsverbund finden Sie unter AWS Identity and Access Management – Häufig gestellte Fragen.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

Example

Das folgende PHP-Beispiel listet Schlüssel im angegebenen Bucket auf. In dem Beispiel erhalten Sie temporäre Sicherheitsanmeldeinformationen für eine einstündige Sitzung für Ihren verbundenen Benutzer (User1). Sie verwenden die temporären Sicherheitsanmeldeinformationen dann zum Senden authentifizierter Anfragen an Amazon S3.

Wenn Sie als zusätzliche Sicherheitsmaßnahme für andere temporäre Sicherheitsanmeldeinformationen anfordern, verwenden Sie die Sicherheitsanmeldeinformationen eines IAM-Benutzers, der zum Anfordern temporärer Sicherheitsanmeldeinformationen berechtigt ist. Um sicherzustellen, dass der IAM-Benutzer dem verbundenen Benutzer ausschließlich die anwendungsspezifischen Mindestberechtigungen gewährt, können Sie auch die Zugriffsberechtigungen des betreffenden IAM-Benutzers einschränken. Dieses Beispiel listet nur Objekte in einem bestimmten Bucket auf. Erstellen Sie einen IAM-Benutzer, dem die folgenden Richtlinie zugeordnet ist:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
```

```
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
}
```

Die Richtlinie gestattet dem IAM-Benutzer, temporäre Sicherheitsanmeldeinformationen anzufordern, und erteilt ihm nur Zugriffsberechtigungen, um Ihre AWS-Ressourcen aufzulisten. Weitere Informationen zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen Ihrer ersten IAM-Benutzer- und Administratorengruppe im IAM-Benutzerhandbuch.

Jetzt können Sie die Sicherheitsanmeldeinformationen des IAM-Benutzers verwenden, um das folgende Beispiel zu testen. Das Beispiel sendet eine unter Verwendung temporärer Sicherheitsanmeldeinformationen eine authentifizierte Anfrage an Amazon S3. Wenn temporäre Sicherheitsanmeldeinformationen für den verbundenen Benutzer (User1) angefordert werden, gibt das Beispiel die folgenden Richtlinie an, womit der Zugriff auf eine Auflistung der Objekte in einem bestimmten Bucket eingeschränkt wird. Aktualisieren Sie die Richtlinie mit Ihrem Bucket-Namen.

```
{
  "Statement":[
    {
      "Sid":"1",
      "Action":["s3:ListBucket"],
      "Effect":"Allow",
      "Resource":"arn:aws:s3:::YourBucketName"
    }
  ]
}
```

Ersetzen Sie im folgenden Beispiel bei der Angabe der Richtlinien-Ressource `YourBucketName` durch den Namen Ihres Buckets:

```
 require 'vendor/autoload.php';

use Aws\Sts\StsClient;
use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';

// In real applications, the following code is part of your trusted code. It has
// the security credentials that you use to obtain temporary security credentials.
$sts = new StsClient(
    [
    'version' => 'latest',
    'region' => 'us-east-1']
);

// Fetch the federated credentials.
$sessionToken = $sts->getFederationToken([
    'Name'              => 'User1',
    'DurationSeconds'    => '3600',
    'Policy'            => json_encode([
        'Statement' => [
            'Sid'               => 'randomstatementid' . time(),
            'Action'            => ['s3:ListBucket'],
            'Effect'            => 'Allow',
            'Resource'          => 'arn:aws:s3:::' . $bucket
        ]
    ])
]);
```

```
// The following will be part of your less trusted code. You provide temporary
// security credentials so the code can send authenticated requests to Amazon S3.

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key'    => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token'  => $sessionToken['Credentials']['SessionToken']
    ]
]);

try {
    $result = $s3->listObjects([
        'Bucket' => $bucket
    ]);
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Anforderungen unter Verwendung temporärer Anmeldeinformationen verbundener Benutzer – AWS SDK für Ruby

Sie können temporäre Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen bereitstellen, sodass sie für den Zugriff auf Ihre AWS-Ressourcen authentifizierte Anfragen senden können. Wenn Sie temporäre Anmeldeinformationen aus dem IAM-Service anfordern, müssen Sie einen Benutzernamen und eine IAM-Richtlinie bereitstellen, in der die Ressourcenberechtigungen beschrieben werden, die Sie erteilen möchten. Die Sitzungsdauer beträgt standardmäßig eine Stunde. Wenn Sie temporäre Sicherheitsanmeldeinformationen unter Verwendung von IAM-Benutzeranmeldeinformationen anfordern, können Sie explizit einen anderen Wert für die Gültigkeitsdauer festlegen, wenn Sie die temporären Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen anfordern. Informationen zu temporären Sicherheitsanmeldeinformationen für Ihre verbundenen Benutzer und Anwendungen finden Sie unter Senden von Anforderungen (p. 11).

> **Note**
>
> Für zusätzliche Sicherheit beim Anfordern temporärer Sicherheitsanmeldeinformationen für verbundene Benutzer und Anwendungen könnten Sie einen dedizierten IAM-Benutzer mit nur den dazu benötigten Zugriffsberechtigungen verwenden. Der temporäre Benutzer, den Sie erstellen, darf nie mehr Berechtigungen erhalten als der IAM-Benutzer, der die temporären Sicherheitsanmeldeinformationen angefordert hat. Weitere Informationen finden Sie unter AWS Identity and Access Management – Häufig gestellte Fragen.

Example

Das folgende Ruby-Codebeispiel gestattet es einem verbundenen Benutzer mit einem limitierten Satz von Berechtigungen Schlüssel im angegebenen Bucket aufzulisten.

```
require 'aws-sdk-s3'
require 'aws-sdk-iam'
```

```
USAGE = <<DOC

Usage: ruby auth_federation_token_request_test.rb -b BUCKET -u USER [-r REGION] [-d] [-h]

  Creates a federated policy for USER to list items in BUCKET for one hour.

  BUCKET is required and must already exist.

  USER is required and if not found, is created.

  If REGION is not supplied, defaults to us-west-2.

  -d gives you extra (debugging) information.

  -h displays this message and quits.

DOC

def print_debug(debug, s)
  if debug
    puts s
  end
end

# Get the user if they exist, otherwise create them
def get_user(region, user_name, debug)
  iam = Aws::IAM::Client.new(region: 'us-west-2')

  # See if user exists
  user = iam.user(user_name)

  # If user does not exist, create them
  if user == nil
    user = iam.create_user(user_name: user_name)
    iam.wait_until(:user_exists, user_name: user_name)
    print_debug(debug, "Created new user #{user_name}")
  else
    print_debug(debug, "Found user #{user_name} in region #{region}")
  end

 user
end

# main
region = 'us-west-2'
user_name = ''
bucket_name = ''

i = 0

while i < ARGV.length
  case ARGV[i]

    when '-b'
      i += 1
      bucket_name = ARGV[i]

    when '-u'
      i += 1
      user_name = ARGV[i]

    when '-r'
      i += 1
      region = ARGV[i]
```

```
    when '-h'
      puts USAGE
      exit 0

    else
      puts 'Unrecognized option: ' + ARGV[i]
      puts USAGE
      exit 1

  end

  i += 1
end

if bucket_name == ''
  puts 'You must supply a bucket name'
  puts USAGE
  exit 1
end

if user_name == ''
  puts 'You must supply a user name'
  puts USAGE
  exit 1
end

# Create a new STS client and get temporary credentials.
sts = Aws::STS::Client.new(region: region)

creds = sts.get_federation_token({
  duration_seconds: 3600,
  name: user_name,
  policy: "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Stmt1\",\"Effect\":\"Allow
\",\"Action\":\"s3:ListBucket\",\"Resource\":\"arn:aws:s3:::#{bucket_name}\"}]}",
})

# Create an Amazon S3 resource with temporary credentials.
s3 = Aws::S3::Resource.new(region: region, credentials: creds)

puts "Contents of '%s':" % bucket_name
puts '  Name => GUID'

begin
  s3.bucket(bucket_name).objects.limit(50).each do |obj|
    puts "  #{obj.key} => #{obj.etag}"
  end
rescue StandardError => ex
  puts 'Caught exception accessing bucket ' + bucket_name + ':'
  puts ex.message
end
```

Senden von Anforderungen unter Verwendung der REST-API

In diesem Abschnitt finden Sie Informationen zum Senden von Anfragen an Amazon S3-Endpunkte über die REST-API. Die Liste der Amazon S3-Endpunkte finden Sie unter Regionen und Endpunkte im AWS General Reference.

Themen

Erstellen von S3-Hostnamen für REST-API-Anforderungen

Amazon S3-Endpunkte folgen der folgenden Struktur:

```
s3.Region.amazonaws.com
```

Amazon S3-Access-Points-Endpunkte und Dual-Stack-Endpunkte folgen ebenfalls der Standardstruktur:

- Amazon S3-Zugriffspunkte -`s3-accesspoint.Region.amazonaws.com`
- Dual-Stack - `s3.dualstack.Region.amazonaws.com`

Eine vollständige Liste der Amazon S3-Regionen und -Endpunkte finden Sie unter Amazon S3-Regionen und -Endpunkte in der Allgemeinen AWS-Referenz.

Anforderungen mit Format mit virtuellem Hosting und Anforderungen im Pfadformat

Beim Senden von Anfragen über die REST-API können Sie für die Amazon S3-Endpunkte Virtual-Hosted-Style- oder Path-Style-URIs verwenden. Weitere Informationen finden Sie unter Virtuelles Hosting bei Buckets (p. 48).

Example Methode im Stil des virtuellen Hostings

Im Folgenden finden Sie ein Beispiel für eine virtuell gehostete Anforderung zum Löschen der Datei `puppy.jpg` aus dem Bucket `examplebucket` in der Region USA West (Oregon). Weitere Informationen zum Wiederholen von Anforderungen finden Sie unter Virtuell gehostete Anforderungen (p. 49).

```
DELETE /puppy.jpg HTTP/1.1
Host: examplebucket.s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example Pfadanforderung

Im Folgenden finden Sie ein Beispiel für eine Path-Style-Version derselben Anfrage.

```
DELETE /examplebucket/puppy.jpg HTTP/1.1
Host: s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Derzeit unterstützt Amazon S3 Virtual Hosted-Style- und Path-Style-Zugriff in allen Regionen, dies wird jedoch geändert (siehe folgenden Hinweis: Wichtig).

Weitere Informationen zu Anforderungen im Pfadformat finden Sie unter Anforderungen im Pfadformat (p. 49).

> **Important**
>
> Buckets, die nach dem 30. September 2020 erstellt werden, unterstützen nur virtuelle Hosted-Style-Anfragen. Path-Style-Anfragen werden weiterhin für Buckets unterstützt, die an oder vor diesem Datum erstellt wurden. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan – The Rest of the Story.

Senden von Anforderungen an Dual-Stack-Endpunkte unter Verwendung der REST-API

Wenn Sie die REST-API verwenden, können Sie direkt auf einen Dual-Stack-Endpunkt zugreifen, indem Sie einen Virtual-Hosted-Style- oder Path-Style-Endpunktnamen (URI) verwenden. Alle Amazon S3-Dual-Stack-Endpunktnamen enthalten die Region. Anders als im Fall von Standardendpunkten, die ausschließlich IPv4 verwenden, verwenden sowohl Virtual-Hosted-Style- als auch Path-Style-Endpunkte regionsspezifische Endpunktnamen.

Example Anforderungen für virtuell gehostete Dual-Stack-Endpunkte

Sie können in Ihrer REST-Anforderung einen virtuell gehosteten Endpunkt wie im folgenden Beispiel gezeigt verwenden, der das Objekt `puppy.jpg` aus dem Bucket `examplebucket` in der Region USA West (Oregon) abruft.

```
GET /puppy.jpg HTTP/1.1
Host: examplebucket.s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example Pfadanforderungen für Dual-Stack-Endpunkte

Sie können auch wie im folgenden Beispiel gezeigt einen Path-Style-Endpunkt in Ihrer Anfrage verwenden.

```
GET /examplebucket/puppy.jpg HTTP/1.1
Host: s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Weitere Informationen zu Dual-Stack-Endpunkten finden Sie unter Verwenden von Amazon S3-Dual-Stack-Endpunkten (p. 16).

Virtuelles Hosting bei Buckets

Das virtuelle Hosting ist das beste Verfahren, um mehrere Websites mit einem Webserver zu unterstützen. Eine Möglichkeit, zwischen den Websites zu unterscheiden, ist die Verwendung des Hostnamens in der Anforderung anstelle des Pfadnamens in der URI. Eine gewöhnliche Amazon S3 REST-Anforderung gibt einen Bucket an, indem sie die erste durch Schrägstrich abgetrennte Komponente des Pfads der Anforderungs-URI verwendet. Alternativ können Sie das virtuelle Hosting von Amazon S3 verwenden, um mit dem HTTP-Header `Host` einen Bucket in einem REST-API-Aufruf anzusprechen. In der Praxis interpretiert Amazon S3 `Host` so, dass die meisten Buckets automatisch für begrenzte Anforderungstypen unter `https://bucketname.s3.Region.amazonaws.com` zur Verfügung stehen. Eine vollständige Liste der Amazon S3-Regionen und -Endpunkte finden Sie unter Amazon S3-Regionen und -Endpunkte in der Allgemeinen AWS-Referenz.

Virtuelles Hosting hat auch andere Vorteile. Wenn Sie Ihrem Bucket denselben Namen wie Ihrer registrierten Domäne geben und diesen Namen dann zu einem DNS-Alias für Amazon S3 machen, können Sie die URL Ihrer Amazon S3-Ressourcen vollständig anpassen, z. B. `https://my.bucketname.com/`. Sie können auch im „Stammverzeichnis“ des virtuellen Servers Ihres Buckets veröffentlichen. Diese Fähigkeit kann wichtig sein, da viele vorhandene Anwendungen nach Dateien an diesem Standardspeicherort suchen. Beispielsweise wird erwartet, dass `favicon.ico`, `robots.txt`, `crossdomain.xml` im Stamm gefunden werden können.

Themen

Anforderungen im Pfadformat

Derzeit unterstützt Amazon S3 Virtual Hosted-Style- und Path-Style-Zugriff in allen Regionen, dies wird jedoch geändert (siehe folgenden Hinweis: Wichtig).

Path Style-Amazon S3-URLs haben das unten gezeigte Format.

```
https://s3.Region.amazonaws.com/bucket-name/key name
```

Wenn Sie beispielsweise einen Bucket namens `mybucket` in der Region USA West (Oregon) erstellen und in dem Bucket auf das Objekt `puppy.jpg` zugreifen möchten, können Sie die folgende Path Style-URL verwenden:

```
https://s3.us-west-2.amazonaws.com/mybucket/puppy.jpg
```

> **Important**
>
> Buckets, die nach dem 30. September 2020 erstellt werden, unterstützen nur virtuelle Hosted-Style-Anfragen. Path-Style-Anfragen werden weiterhin für Buckets unterstützt, die an oder vor diesem Datum erstellt wurden. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan – The Rest of the Story.

Virtuell gehostete Anforderungen

In einem URI mit virtuellem Hosting ist der Bucket-Name Teil des Domänennamens in der URL.

Virtual Hosted-Amazon S3-URLs haben das unten gezeigte Format.

```
https://bucket-name.s3.Region.amazonaws.com/key name
```

In diesem Beispiel ist `my-bucket` der Bucket-Name, "US West (Oregon) (USA West (Oregon))" die Region und `puppy.png` der Schlüsselname:

```
https://my-bucket.s3.us-west-2.amazonaws.com/puppy.png
```

Bucket-Spezifikation für HTTP-Host-Header

So lange Ihre `GET`-Anforderung nicht den SSL-Endpunkt verwendet, können Sie mit dem HTTP-`Host`-Header den Bucket für die Anforderung festlegen. Der `Host`-Header in einer REST-Anforderung wird folgendermaßen interpretiert:

- Wenn der Header `Host` ausgelassen wird oder der Wert `s3.`*`Region`*`.amazonaws.com` lautet, ist der Bucket für die Anforderung die erste durch Schrägstrich abgetrennte Komponente des Anforderungs-URI und der Schlüssel für die Anforderung bildet den Rest des Anforderungs-URI. Dies ist die übliche Methode wie im ersten und zweiten Beispiel in diesem Abschnitt dargestellt. Das Auslassen des Host-Headers ist nur bei HTTP 1.0-Anforderungen möglich.
- Wenn andernfalls der Wert des `Host`-Headers mit `.s3.`*`Region`*`.amazonaws.com` endet, ist der Bucket-Name die führende Komponente im Wert des `Host`-Headers bis zu `.s3.`*`Region`*`.amazonaws.com`. Der Schlüssel für die Anforderung ist die Anforderungs-URI. Diese Interpretation stellt Buckets als Unterdomänen von `.s3.`*`Region`*`.amazonaws.com` bereit (siehe das dritte und vierte Beispiel in diesem Abschnitt).
- Ansonsten ist der Bucket für die Anforderung der klein geschriebene Wert des `Host`-Headers und die Schlüssel für die Anforderung ist die Anforderungs-URI. Diese Interpretation ist nützlich, wenn Sie als DNS-Namen Ihren Bucket-Namen registriert und diesen Namen so konfiguriert haben, dass er ein CNAME-Alias für Amazon S3 ist. Das Verfahren zum Registrieren von Domänennamen und Konfigurieren von DNS ist nicht Gegenstand dieses Leitfadens. Das Ergebnis wird jedoch im letzten Beispiel in diesem Abschnitt dargestellt.

Beispiele

In diesem Abschnitt finden Sie Beispiel-URLs und -anforderungen.

Example Pfadformat

In diesem Beispiel wird Folgendes verwendet:

- Bucket-Name - `awsexamplebucket1.net`
- Region - USA Ost (Nord-Virginia)
- Schlüsselname - `homepage.html`

Die URL lautet wie folgt:

```
https://s3.us-east-1.amazonaws.com/awsexamplebucket1.net/homepage.html
```

die Anforderung lautet wie folgt:

```
GET /awsexamplebucket1.net/homepage.html HTTP/1.1
Host: s3.us-east-1.amazonaws.com
```

die Anforderung mit HTTP 1.0 unter Auslassen des `host`-Headers lautet wie folgt:

```
GET /awsexamplebucket1.net/homepage.html HTTP/1.0
```

Informationen zu mit DNS kompatiblen Namen finden Sie unter Einschränkungen (p. 52). Weitere Informationen zu Schlüsseln finden Sie unter Schlüssel (p. 4).

Example Format mit virtuellem Hosting

In diesem Beispiel wird Folgendes verwendet:

- Bucket-Name - `awsexamplebucket1.eu`
- Region - Europa (Irland)
- Schlüsselname - `homepage.html`

Die URL lautet wie folgt:

```
https://awsexamplebucket1.eu.s3.eu-west-1.amazonaws.com/homepage.html
```

die Anforderung lautet wie folgt:

```
GET /homepage.html HTTP/1.1
Host: awsexamplebucket1.eu.s3.eu-west-1.amazonaws.com
```

Example CNAME-Methode

Um diese Methode zu verwenden, müssen Sie den DNS-Namen als CNAME-Alias für *`bucketname`*`.s3.us-east-1.amazonaws.com` konfigurieren. Weitere Informationen finden Sie unter Anpassen von Amazon S3-URLs mit CNAMEs (p. 51). In diesem Beispiel wird Folgendes verwendet:

- Bucket-Name - `awsexamplebucket1.net`
- Schlüsselname - `homepage.html`

Die URL lautet wie folgt:

```
https://www.awsexamplebucket1.net/homepage.html
```

Das Beispiel ist wie folgt:

```
GET /homepage.html HTTP/1.1
Host: www.awsexamplebucket1.net
```

Anpassen von Amazon S3-URLs mit CNAMEs

Abhängig von den Anforderungen können Sie die Anzeige von `s3.`*`Region`*`.amazonaws.com` auf der Website oder im Service gegebenenfalls unterbinden. Wenn Sie die Websitebilder beispielsweise auf Amazon S3 hosten, bevorzugen Sie möglicherweise `https://images.awsexamplebucket1.net/` anstelle von `https://awsexamplebucket1-images.s3.us-east-1.amazonaws.com/`.
Auf Buckets mit einem mit DNS kompatiblen Namen kann wie folgt verwiesen werden: `https://`*`BucketName`*`.s3.`*`Region`*`.amazonaws.com/[`*`Filename`*`]`, z. B. `https://images.awsexamplebucket1.net.s3.us-east-1.amazonaws.com/mydog.jpg`. Durch die Verwendung von CNAME können Sie `images.awsexamplebucket1.net` einem Amazon S3-Host-Namen zuordnen, sodass die vorherige URL als `https://images.awsexamplebucket1.net/mydog.jpg` angezeigt wird.

Der Bucket-Name muss dem CNAME entsprechen. Wenn Sie z. B. einen CNAME für die Zuordnung von `images.awsexamplebucket1.net` zu `images.awsexamplebucket1.net.s3.amazonaws.com` erstellen, sind `https://images.awsexamplebucket1.net/filename` und `https://images.awsexamplebucket1.net.s3.us-east-1.amazonaws.com/filename` identisch.

Der CNAME-DNS-Datensatz muss einen Alias des Domänennamens für den entsprechenden Host-Namen im Stil des virtuellen Hostings erstellen. Wenn der Bucket-Name und der Domänenname z. B. `images.awsexamplebucket1.net` lauten und sich Ihr Bucket in der Region USA Ost (Nord-Virginia)

befindet, sollte der CNAME-Datensatz einen Alias für `images.awsexamplebucket1.net.s3.us-east-1.amazonaws.com` erstellen.

```
images.awsexamplebucket1.net CNAME    images.awsexamplebucket1.net.s3.us-
east-1.amazonaws.com.
```

Amazon S3 verwendet den Host-Namen, um den Bucket-Namen zu ermitteln. CNAME und Bucket-Name müssen also identisch sein. Nehmen wir beispielsweise an, dass Sie `www.example.com` als CNAME für `www.example.com.s3.us-east-1.amazonaws.com` konfiguriert haben. Wenn Sie auf `https://www.example.com` zugreifen, empfängt Amazon S3 eine Anforderung, die in etwa wie folgt aussieht:

Example

```
GET / HTTP/1.1
Host: www.example.com
Date: date
Authorization: signatureValue
```

Amazon S3 sieht nur den ursprünglichen Host-Namen `www.example.com` und kennt die zum Auflösen der Anforderung verwendete CNAME-Zuordnung nicht.

In einem CNAME können alle Amazon S3-Endpunkte verwendet werden. Beispielsweise kann `s3.ap-southeast-1.amazonaws.com` in CNAMEs verwendet werden. Weitere Informationen zu Endpunkten finden Sie unter Anforderungsendpunkte (p. 13).

So verknüpfen Sie einen Host-Namen unter Verwendung von CNAMEs mit einem Amazon S3-Bucket

1. Wählen Sie einen Host-Namen aus, der zu einer von Ihnen kontrollierten Domäne gehört.

   Dieses Beispiel verwendet die Unterdomäne `images` der Domäne `awsexamplebucket1.net`.
2. Erstellen Sie einen Bucket, der dem Host-Namen entspricht.

   In diesem Beispiel lauten der Host- und der Bucket-Name `images.awsexamplebucket1.net`. Der Bucket-Name muss exakt mit dem Host-Namen übereinstimmen.
3. Erstellen Sie einen CNAME-Datensatz, der den Host-Namen als Alias für den Amazon S3-Bucket definiert.

   Beispiel:

   `images.awsexamplebucket1.net CNAME images.awsexamplebucket1.net.s3.us-west-2.amazonaws.com`

   > **Important**
   >
   > Aus Gründen des Anforderungs-Routings muss der CNAME-Datensatz genau wie im vorherigen Beispiel dargestellt definiert werden. Andernfalls kann sich trotz richtig erscheinender Funktion unvorhersehbares Verhalten ergeben.

   Das Verfahren für die Konfiguration von DNS ist abhängig von Ihrem DNS-Server oder DNS-Anbieter. Spezifische Informationen finden Sie in Ihrer Serverdokumentation oder erhalten Sie von Ihrem Anbieter.

Einschränkungen

SSL

Virtuelle gehostete URLs werden nur für Nicht-SSL-(HTTP)-Anforderungen unterstützt. Bei Verwendung von virtuell gehosteten Buckets mit SSL stimmt das SSL-Platzhalterzertifikat nur mit Buckets überein, die keine Punkte enthalten. Um dies zu umgehen, verwenden Sie HTTP oder schreiben Sie Ihre eigene Logik zur Verifizierung von Zertifikaten.

SOAP

Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Abwärtskompatibilität

Legacy-Endpunkte

Einige Regionen unterstützen Legacy-Endpunkte. Möglicherweise werden diese Endpunkte in Ihren Serverzugriffsprotokollen oder CloudTrail-Protokollen angezeigt. Weitere Informationen finden Sie in den nachstehenden Informationen. Eine vollständige Liste der Amazon S3-Regionen und -Endpunkte finden Sie unter Amazon S3-Regionen und -Endpunkte in der Allgemeinen AWS-Referenz.

> **Important**
>
> Obwohl Sie möglicherweise Legacy-Endpunkte in Ihren Protokollen sehen, empfehlen wir Ihnen, immer die Standardendpunktsyntax für den Zugriff auf Ihre Buckets zu verwenden.
> Virtual Hosted-Amazon S3-URLs haben das unten gezeigte Format.

```
https://bucket-name.s3.Region.amazonaws.com/key name
```

Path Style-Amazon S3-URLs haben das unten gezeigte Format.

```
https://s3.Region.amazonaws.com/bucket-name/key name
```

s3-Region

Einige ältere Amazon S3-Regionen unterstützen Endpunkte, die einen Bindestrich zwischen S3 und der Region (z. B. `S3#us-west-2`) anstelle eines Punkts (z. B. `S3.us-west-2`) enthalten. Wenn sich Ihr Bucket in einer dieser Regionen befindet, wird möglicherweise das folgende Endpunktformat in Ihren Serverzugriffsprotokollen oder CloudTrail-Protokollen angezeigt:

```
https://bucket-name.s3-Region.amazonaws.com
```

In diesem Beispiel lautet der Bucket-Name `my-bucket` und die Region USA West (Oregon):

```
https://my-bucket.s3-us-west-2.amazonaws.com
```

Globaler Legacy-Endpunkt

Für einige Regionen kann der globale Legacy-Endpunkt verwendet werden, um Anforderungen zu erstellen, die keinen regionsspezifischen Endpunkt angeben. Der globale Legacy-Endpunkt lautet wie folgt:

```
bucket-name.s3.amazonaws.com
```

In Ihren Serverzugriffsprotokollen oder CloudTrail-Logs sehen Sie möglicherweise Anforderungen, die den globalen Legacy-Endpunkt verwenden. In diesem Beispiel lautet der Bucket-Name `my-bucket` und wird der globale Legacy-Endpunkt angezeigt:

```
https://my-bucket.amazonaws.com
```

Virtuell gehostete Anforderungen für USA Ost (Nord-Virginia)

Anforderungen, die mit dem globalen Legacy-Endpunkt gemacht werden, werden standardmäßig an USA Ost (Nord-Virginia) weitergeleitet. Daher wird manchmal der globale Legacy-Endpunkt anstelle des regionalen Endpunkts für USA Ost (Nord-Virginia) verwendet. Wenn Sie einen Bucket in USA Ost (Nord-Virginia) erstellen und den globalen Endpunkt verwenden, leitet Amazon S3 Ihre Anforderung standardmäßig an diese Region weiter.

Anforderungen mit virtuellem Hosting für andere Regionen

Der globale Legacy-Endpunkt wird auch für virtuell gehostete Anforderungen in anderen unterstützten Regionen verwendet. Wenn Sie einen Bucket in einer Region erstellen, die vor dem 20. März 2019 gestartet wurde, und den globalen Legacy-Endpunkt verwenden, aktualisiert Amazon S3 das DNS so, dass die Anforderung an den richtigen Speicherort umgeleitet wird. Dies kann eine Weile dauern. In der Zwischenzeit gilt die Standardregel und Ihre virtuell gehostete Anforderung wird an die Region USA Ost (Nord-Virginia) gesendet. Amazon S3 leitet diese dann mit einer HTTP 307-Umleitung zur richtigen Region. Für S3-Buckets in nach dem 20. März 2019 gestarteten Regionen leitet der DNS die Anforderung nicht direkt an die AWS-Region weiter, in der sich der Bucket befindet. Stattdessen wird ein "HTTP 400 Bad Request"-Fehler zurückgegeben. Weitere Informationen finden Sie unter Anforderungsumleitung und die REST API (p. 726).

Anforderungen im Pfadformat

Für die Region USA Ost (Nord-Virginia) kann der globale Legacy-Endpunkt für Pfadanforderungen verwendet werden.

Für alle anderen Regionen erfordert die Syntax im Pfadformat, dass beim Zugriff auf einen Bucket der regionsspezifische Endpunkt verwendet werden muss. Wenn Sie versuchen, auf einen Bucket mit dem globalen Legacy-Endpunkt oder einem anderen Endpunkt zuzugreifen, der sich von dem für die Region unterscheidet, in der sich der Bucket befindet, erhalten Sie als Antwortcode einen „HTTP 307 Temporary Redirect“-Fehler und eine Meldung, die den richtigen URI für Ihre Ressource angibt. Wenn Sie beispielsweise `https://s3.amazonaws.com/`*`bucket-name`* für einen Bucket verwenden, der in der Region USA West (Oregon) erstellt wurde, wird ein „HTTP 307 Temporary Redirect“-Fehler angezeigt.

Anforderungsumleitung und die REST API

Themen

In diesem Abschnitt wird beschrieben, wie HTTP-Umleitungen unter Verwendung der Amazon S3 REST API verarbeitet werden. Allgemeine Informationen zu Amazon S3-Umleitungen finden Sie unter Anforderungsumleitung und die REST API (p. 726) im Amazon Simple Storage Service API Reference.

Umleitungen und HTTP-Benutzeragenten

Programme, die die Amazon S3 REST API verwenden, sollten Umleitungen auf der Anwendungsschicht oder auf der HTTP-Schicht verarbeiten. Es können viele HTTP-Client-Bibliotheken und Benutzeragenten konfiguriert werden, um Umleitungen automatisch korrekt zu verarbeiten. Viele andere beinhalten jedoch fehlerhafte oder unvollständige Implementierungen für Umleitungen.

Bevor Sie sich darauf verlassen, dass eine Bibliothek die Umleitungsanfrage erfüllt, testen Sie die folgenden Fälle:

- Überprüfen Sie, ob alle HTTP-Anfrageheader korrekt in die Umleitungsanfrage aufgenommen wurden (die zweite Anfrage nach Empfang einer Umleitung), einschließlich der HTTP-Standards, wie beispielsweise Autorisierung und Datum.
- Überprüfen Sie, ob auch andere als GET-Umleitungen ordnungsgemäß funktionieren, wie beispielsweise PUT und DELETE.
- Überprüfen Sie, ob PUT-Anfragen den Umleitungen ordnungsgemäß folgen.
- Überprüfen Sie, ob PUT-anfragenden Umleitungen ordnungsgemäß folgen, wenn die Antwort auf 100 continue lang dauert.

HTTP-Benutzeragenten, die streng konform zu RFC 2616 arbeiten, fordern möglicherweise eine explizite Bestätigung, bevor sie einer Umleitung folgen, wenn die HTTP-Anfragemethode nicht GET oder HEAD ist. Im Allgemeinen ist es sicher, von Amazon S3 erstellten Umleitungen automatisch zu folgen, da das System ausschließlich Umleitungen zu Hosts innerhalb der Domäne amazonaws.com ausgibt und der Effekt der umgeleiteten Anfrage mit dem der ursprünglichen Anfrage identisch ist.

Umleitungen und 100 Continue

Um die Verarbeitung von Umleitungen zu vereinfachen, die Effizienzen zu verbessern und die Kosten zu vermeiden, die beim doppelten Versenden eines umgeleiteten Anfragerumpfs entstehen, konfigurieren Sie Ihre Anwendung so, dass sie 100 continues für PUT-Operationen verwendet. Wenn Ihre Anwendung 100 continue verwendet, sendet sie den Anfragerumpf erst dann, wenn sie eine Bestätigung erhält. Wird die Nachricht basierend auf den Header abgewiesen, wird der Rumpf der Meldung nicht gesendet. Weitere Informationen zu 100-continue finden Sie unter RFC 2616 Abschnitt 8.2.3.

> Note
>
> Laut RFC 2616 sollten Sie nicht unendlich lange warten, bevor Sie den Anfragerumpf senden, wenn Sie `Expect: Continue` für einen unbekannten HTTP-Server verwenden. Der Grund dafür ist, dass einige HTTP-Server 100 continue nicht erkennen. Amazon S3 erkennt jedoch nicht, ob Ihre Anforderung `Expect: Continue` enthält und antwortet mit einem vorläufigen 100-continue-Status oder einem endgültigen Statuscode. Darüber hinaus tritt kein Umleitungscode auf, nachdem die provisorische 100-continue-Genehmigung empfangen wurde. Dies hilft Ihnen, den Empfang einer Umleitungsantwort zu vermeiden, wenn Sie noch den Anfragerumpf schreiben.

Beispiel für eine Umleitung

Dieser Abschnitt bietet ein Beispiel für eine Client/Server-Interaktion mit HTTP-Umleitungen und 100 continue.

Nachfolgend finden Sie ein Beispiel für ein PUT in den `quotes.s3.amazonaws.com`-Bucket.

```
PUT /nelson.txt HTTP/1.1
Host: quotes.s3.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000

Content-Length: 6
Expect: 100-continue
```

Amazon S3 gibt Folgendes zurück:

```
HTTP/1.1 307 Temporary Redirect
Location: http://quotes.s3-4c25d83b.amazonaws.com/nelson.txt?rk=8d47490b
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Mon, 15 Oct 2007 22:18:46 GMT
```

```
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the
  specified temporary endpoint. Continue to use the
  original request endpoint for future requests.
  </Message>
  <Endpoint>quotes.s3-4c25d83b.amazonaws.com</Endpoint>
  <Bucket>quotes</Bucket>
</Error>
```

Der Client folgt der Umleitungsantwort und gibt eine neue Anfrage an den temporären Endpunkt `quotes.s3-4c25d83b.amazonaws.com` aus.

```
PUT /nelson.txt?rk=8d47490b HTTP/1.1
Host: quotes.s3-4c25d83b.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000

Content-Length: 6
Expect: 100-continue
```

Amazon S3 gibt ein 100 continue zurück, das darauf hinweist, dass der Client damit fortfahren soll, den Anfragerumpf zu senden.

```
HTTP/1.1 100 Continue
```

Der Client sendet den Anfragerumpf.

```
ha ha\n
```

Amazon S3 gibt die endgültige Antwort zurück:

```
HTTP/1.1 200 OK
Date: Mon, 15 Oct 2007 22:18:48 GMT

ETag: "a2c8d6b872054293afd41061e93bc289"
Content-Length: 0
Server: AmazonS3
```

Arbeiten mit Amazon S3-Buckets

Zum Hochladen Ihrer Daten (Fotos, Videos, Dokumente usw.) auf Amazon S3 erstellen Sie zunächst einen S3-Bucket in einer AWS-Region. Anschließend können Sie beliebig viele Objekte in den Bucket hochladen.

Im Hinblick auf die Implementierung sind Buckets und Objekte Ressourcen, und Amazon S3 stellt die APIs für ihre Verwaltung bereit. Beispielsweise können Sie mit dem Amazon S3 API einen Bucket erstellen und Objekte hochladen. Sie können diese Operationen auch in der Amazon S3-Konsole durchführen. Die Konsole verwendet die Amazon S3-APIs, um Anforderungen an Amazon S3 zu senden.

In diesem Abschnitt wird beschrieben, wie Sie mit Buckets arbeiten. Weitere Informationen zur Arbeit mit Objekten finden Sie unter Arbeiten mit Amazon S3-Objekten (p. 116).

Der Name eines Amazon S3-Buckets ist global eindeutig, und der Namespace wird von allen AWS-Konten verwendet. Dies bedeutet, dass der Name eines Buckets nach seiner Erstellung nicht von einem anderen AWS-Konto in irgendeiner AWS-Region verwendet werden kann, bis der Bucket gelöscht wird. Verlassen Sie sich nicht auf eine spezifische Benennungskonvention für Buckets für Verfügbarkeits- oder Sicherheitsprüfungszwecke. Namensrichtlinien für Buckets finden Sie unter Beschränkungen und Einschränkungen von Buckets (p. 64).

Amazon S3 erstellt Buckets in der von Ihnen angegebenen Region. Um die Latenz zu optimieren, Kosten zu minimieren und gesetzliche Anforderungen zu berücksichtigen, können Sie eine beliebige AWS-Region in der Nähe wählen. Wenn Sie beispielsweise in Europa ansässig sind, könnte es vorteilhaft sein, Buckets in der Region Europa (Irland) oder Europa (Frankfurt) zu erstellen. Eine Liste der Amazon S3-Regionen finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS-Referenz.

> Note
>
> Objekte, die zu einem Bucket gehören, den Sie in einer spezifischen AWS-Region erstellt haben, verbleiben so lange in der Region, bis sie explizit in eine andere Region verschoben werden. Beispielsweise verlassen in der Region Europa (Irland) gespeicherte Objekte diese Region nie.

Themen

Erstellen eines Buckets

Amazon S3 stellt APIs bereit, mit denen Sie Buckets erstellen und verwalten können. In der Standardeinstellung können Sie bis zu 100 Buckets in jedem Ihrer AWS-Konten erstellen. Wenn Sie weitere Buckets benötigen, können Sie das Konto-Bucket-Limit auf maximal 1 000 Buckets erhöhen, indem Sie eine Service Limit-Erhöhung senden. Weitere Informationen über das Senden einer Bucket-Limit-

Erhöhung finden Sie unter Limits für AWS-Services in der Allgemeinen AWS-Referenz. Sie können in einem Bucket beliebig viele Objekte speichern.

Wenn Sie ein Bucket erstellen, geben Sie einen Namen und die AWS-Region an, in der der Bucket erstellt werden soll. Weitere Informationen zur Benennung von Buckets finden Sie unter Regeln für die Bucket-Benennung (p. 64).

Zum Erstellen eines Buckets können Sie eine der unten aufgeführten Methoden verwenden. Beispiele finden Sie unter Beispiel zum Erstellen eines Buckets (p. 65).

Amazon S3-Konsole

Sie können einen Bucket in der Amazon S3-Konsole erstellen. Weitere Informationen finden Sie unter Erstellen eines Buckets im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

REST-API

Das Erstellen eines Buckets mit der REST-API kann umständlich sein, da Sie Code schreiben müssen, um Ihre Anforderungen zu authentifizieren. Weitere Informationen finden Sie unter PUT Bucket in der Amazon Simple Storage Service API Reference. Es wird empfohlen, stattdessen die AWS Management Console oder AWS SDKs zu verwenden.

AWS SDK

Wenn Sie die AWS-SDKs zum Erstellen eines Buckets verwenden, erstellen Sie zuerst einen Client und senden mit dem Client dann eine Anforderung zum Erstellen eines Buckets. Wenn Sie beim Erstellen eines Clients oder eines Buckets keine Region angeben, verwendet Amazon S3 USA Ost (Nord-Virginia), die Standardregion. Sie können auch eine bestimmte Region angeben. Eine Liste der verfügbaren AWS-Regionen finden Sie unter Regionen und Endpunkte im AWS General Reference. Weitere Informationen zum Aktivieren oder Deaktivieren einer AWS-Region finden Sie unter Verwalten von AWS-Regionen im AWS General Reference.

Globaler Legacy-Endpunkt

Als bewährte Methode sollten Sie Ihren Client und Ihren Bucket in derselben Region erstellen. Wenn Sie Ihren Client standardmäßig oder manuell in der Region USA Ost (Nord-Virginia) erstellen, kann Amazon S3 jedoch den globalen Legacy-Endpunkt verwenden, um mit dem Client zu kommunizieren:

```
s3.amazonaws.com
```

Daher können Sie einen Client in USA Ost (Nord-Virginia) verwenden, um einen Bucket in jeder Region zu erstellen, die vor dem 20. März 2019 gestartet wurde. Wenn die Region, in der Sie Ihren Bucket erstellen möchten, nach dem 20. März 2019 gestartet wurde, müssen sich Ihr Client und Bucket jedoch in derselben Region befinden. Weitere Informationen finden Sie unter Legacy-Endpunkte (p. 53).

Erstellen eines Clients

Wenn Sie den Client erstellen, können Sie eine AWS-Region angeben, in der der Client erstellt werden soll. Wenn Sie keine Region angeben, erstellt Amazon S3 den Bucket in USA Ost (Nord-Virginia), der Standard-Region. Um einen Client zu erstellen, um auf einen Dual-Stack-Endpunkt zuzugreifen, müssen Sie eine AWS-Region angeben. Weitere Informationen finden Sie unter Dual-Stack-Endpunkte (p. 16).

Wenn Sie einen Client erstellen, wird die Region dem regionsspezifischen Endpunkt zugeordnet. Der Client verwendet diesen Endpunkt für die Kommunikation mit Amazon S3:

```
s3.<region>.amazonaws.com
```

Wenn Sie beispielsweise einen Client unter Angabe der Region eu-west-1 erstellen, wird sie dem folgenden regionsspezifischen Endpunkt zugeordnet:

```
s3.eu-west-1.amazonaws.com
```

Erstellen eines Buckets

Wenn Sie beim Erstellen eines Buckets keine Region angeben, erstellt Amazon S3 den Bucket in der Region USA Ost (Nord-Virginia). Wenn Sie also einen Bucket in einer bestimmten Region erstellen müssen, müssen Sie beim Erstellen des Buckets die Region angeben.

Buckets, die nach dem 30. September 2020 erstellt werden, unterstützen nur virtuelle Hosted-Style-Anfragen. Path-Style-Anfragen werden weiterhin für Buckets unterstützt, die an oder vor diesem Datum erstellt wurden. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan – The Rest of the Story.

Berechtigungen

Sie können die Root-Anmeldeinformationen Ihres AWS-Kontos verwenden, um einen Bucket zu erstellen und andere Amazon S3-Operationen auszuführen. AWS rät davon ab, die Root-Anmeldeinformationen Ihres AWS-Kontos für Anforderungen zu nutzen, wie etwa zum Erstellen eines Buckets. Erstellen Sie stattdessen einen IAM-Benutzer, dem Sie vollständigen Zugriff gewähren (Benutzer haben standardmäßig keine Berechtigung). Wir bezeichnen diese Benutzer als Administratorbenutzer. Anstelle der Root-Anmeldeinformationen Ihres Kontos können Sie die des Administratorbenutzers für Aufgaben in AWS verwenden, z. B. um einen Bucket und Benutzer zu erstellen sowie Benutzern Berechtigungen zu gewähren.

Weitere Informationen finden Sie unter Root-Kontoanmeldeinformationen vs. IAM-Benutzerberechtigungen in der Allgemeinen AWS-Referenz sowie unter Bewährte Methoden für IAM im IAM-Benutzerhandbuch.

Bei dem AWS-Konto, das die Ressource erstellt, handelt es sich um den Ressourceneigentümer. Wenn Sie beispielsweise einen IAM-Benutzer in Ihrem AWS-Konto erstellen und diesem Berechtigungen zum Erstellen eines Buckets erteilen, kann der Benutzer einen Bucket erstellen. Dem Benutzer gehört jedoch der Bucket nicht. Der Bucket gehört dem AWS-Konto, zu dem der Benutzer gehört. Der Benutzer braucht eine zusätzliche Berechtigung vom Bucket-Eigentümer, um andere Bucket-Operation ausführen zu dürfen. Weitere Informationen zum Verwalten von Berechtigungen für Ihre Amazon S3-Ressourcen finden Sie unter Identity and Access Management in Amazon S3 (p. 329).

Verwalten des öffentlichen Zugriffs auf Buckets

Öffentlicher Zugriff auf Buckets und Objekte wird über Zugriffskontrolllisten (ACLs), Bucket-Richtlinien oder beides gewährt. Um Sie bei der Verwaltung des öffentlichen Zugriffs auf Amazon S3-Ressourcen zu unterstützen, stellt Amazon S3 Einstellungen zum Blockieren des öffentlichen Zugriffs zur Verfügung. Amazon S3-Einstellungen zum Blockieren des öffentlichen Zugriffs können ACLs und Bucket-Richtlinien außer Kraft setzen, sodass Sie den öffentlichen Zugriff auf diese Ressourcen einheitlich beschränken können. Sie können Einstellungen zum Blockieren des öffentlichen Zugriffs auf einzelne Buckets oder auf alle Buckets in Ihrem Konto anwenden.

Um zu gewährleisten, dass bei allen Ihren Amazon S3-Buckets und -Objekten der öffentliche Zugriff blockiert ist, empfehlen wird, alle vier Einstellungen zur Blockierung des öffentlichen Zugriffs für Ihr Konto zu aktivieren. Diese Einstellungen blockieren den öffentlichen Zugriff für alle aktuellen und zukünftigen Buckets.

Bevor Sie diese Einstellungen anwenden, verifizieren Sie, dass Ihre Anwendungen ohne öffentlichen Zugriff korrekt funktionieren. Wenn ein bestimmter Umfang an öffentlichem Zugriff auf Ihre Buckets oder Objekte nötig ist, z. B. zum Hosten einer statischen Website, wie unter Hosten einer statischen Website auf Amazon S3 (p. 602) beschrieben, können Sie die einzelnen Einstellungen an Ihre

Speicheranwendungsfälle anpassen. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493).

Zugriff auf einen Bucket

Sie können auf Ihren Bucket mithilfe der Amazon S3-Konsole zugreifen. Mit der Benutzeroberfläche der Konsole können Sie fast alle Bucket-Operationen ausführen, ohne Code schreiben zu müssen.

Wenn Sie programmgesteuert auf einen Bucket zugreifen, achten Sie darauf, dass Amazon S3 die RESTful-Architektur unterstützt, wobei Ihre Buckets und Objekte Ressourcen sind, die alle eine Ressourcen-URI besitzen, die die Ressource eindeutig identifizieren.

Amazon S3 unterstützt URLs im Stil von virtuellem Hosting und im Path-Stil für den Zugriff auf einen Bucket. Da der Zugriff auf Buckets über URLs im Pfadstil und im Stil des virtuellen Hostings möglich ist, empfehlen wir Ihnen, Buckets mit DNS-konformen Bucket-Namen zu erstellen. Weitere Informationen finden Sie unter Beschränkungen und Einschränkungen von Buckets (p. 64).

> Note
>
> Virtuelle gehostete Style- und Pfadanfragen verwenden beispielsweise die S3-Regionen-Endpunktstruktur mit Punkt (`s3.Region`), z. B. `https://my-bucket.s3.us-west-2.amazonaws.com`. Einige ältere Amazon S3-Regionen unterstützen jedoch auch S3-Regionsendpunkte mit Bindestrich (`s3-Region`), z. B. `https://my-bucket.s3-us-west-2.amazonaws.com`. Wenn sich Ihr Bucket in einer dieser Regionen befindet, werden möglicherweise `s3-Region`-Endpunkte in Ihren Serverzugriffsprotokollen oder CloudTrail-Protokollen angezeigt. Es wird empfohlen, diese Endpunktstruktur nicht in Ihren Anforderungen zu verwenden.

Zugriff mit virtuellem Hosting

In einer virtuell gehosteten Anforderung ist der Bucket-Name Teil des Domänennamens in der URL.

Virtual Hosted-Amazon S3-URLs haben das unten gezeigte Format.

```
https://bucket-name.s3.Region.amazonaws.com/key name
```

In diesem Beispiel ist `my-bucket` der Bucket-Name, "US West (Oregon) (USA West (Oregon))" die Region und `puppy.png` der Schlüsselname:

```
https://my-bucket.s3.us-west-2.amazonaws.com/puppy.png
```

Weitere Informationen zum Zugriff mit virtuellem Hosting finden Sie unter Virtuell gehostete Anforderungen (p. 49).

Zugriff im Pfadformat

Path Style-Amazon S3-URLs haben das unten gezeigte Format.

```
https://s3.Region.amazonaws.com/bucket-name/key name
```

Wenn Sie beispielsweise einen Bucket namens `mybucket` in der Region USA West (Oregon) erstellen und in dem Bucket auf das Objekt `puppy.jpg` zugreifen möchten, können Sie die folgende Path Style-URL verwenden:

```
https://s3.us-west-2.amazonaws.com/mybucket/puppy.jpg
```

Weitere Informationen finden Sie unter Anforderungen im Pfadformat (p. 49).

Important

Buckets, die nach dem 30. September 2020 erstellt werden, unterstützen nur virtuelle Hosted-Style-Anfragen. Path-Style-Anfragen werden weiterhin für Buckets unterstützt, die an oder vor diesem Datum erstellt wurden. Weitere Informationen finden Sie unter Amazon S3 Path Deprecation Plan – The Rest of the Story.

Zugriff auf einen S3-Bucket über IPv6

Amazon S3 verfügt über mehrere Dual-Stack-Endpunkte, die Anforderungen an S3-Buckets über Internetprotokollversion 6 (IPv6) und IPv4 unterstützen. Weitere Informationen finden Sie unter Senden von Anfragen über IPv6 (p. 13).

Zugreifen auf einen Bucket über einen S3 Access Point

Zusätzlich zum direkten Zugriff auf einen Bucket können Sie über einen S3-Zugriffspunkt auf einen Bucket zugreifen. Weitere Hinweise zu S3 Access Points finden Sie unter Verwalten des Datenzugriffs mit Amazon S3 Access Points (p. 103).

S3 Access Points unterstützen nur die Adressierung im Virtual-Host-Stil. Um einen Bucket über einen Zugriffspunkt zu adressieren, verwenden Sie dieses Format:

```
https://AccessPointName-AccountId.s3-accesspoint.region.amazonaws.com.
```

Note

- Wenn der Name Ihres Zugriffspunkts Bindestriche (-) enthält, fügen Sie die Bindestriche in die URL ein und fügen einen weiteren Bindestrich vor der Konto-ID ein. Wenn Sie beispielsweise einen Zugriffspunkt mit dem Namen `finance-docs` verwenden möchten, der im Besitz des Kontos `123456789012` in der Region `us-west-2` ist, würde die entsprechende URL `https://finance-docs-123456789012.s3-accesspoint.us-west-2.amazonaws.com` lauten.
- S3-Zugriffspunkte unterstützen keinen Zugriff über HTTP, nur den sicheren Zugriff über HTTPS.

Zugriff auf einen Bucket mit S3: //

Einige AWS-Services erfordern die Angabe eines Amazon S3-Buckets mit `S3://bucket`. Das richtige Format wird nachstehend dargestellt. Beachten Sie bei der Verwendung dieses Formats, dass der Bucket-Name die Region nicht enthält.

```
S3://bucket-name/key-name
```

Zum Beispiel bei der im früheren Abschnitt „Pfadformat“ beschriebenen Verwendung des Beispiel-Buckets:

```
S3://mybucket/puppy.jpg
```

Optionen für die Bucket-Konfiguration

Amazon S3 unterstützt verschiedene Optionen für die Konfiguration Ihres Buckets. Sie können beispielsweise Ihren Bucket für ein Website-Hosting konfigurieren, eine Konfiguration zur Verwaltung

des Lebenszyklus von Objekten im Bucket hinzufügen und den Bucket so konfigurieren, dass er den gesamten Zugriff auf den Bucket protokolliert. Amazon S3 unterstützt Subressourcen, um die Bucket-Konfigurationsinformationen zu speichern und zu verwalten. Sie können diese Subressourcen mit der Amazon S3-API erstellen und verwalten. Sie können auch die Konsole oder die AWS SDKs verwenden.

> **Note**
>
> Außerdem gibt es Konfigurationen auf Objektebene. Beispielsweise können Sie Berechtigungen auf Objektebene konfigurieren, indem Sie eine für dieses Objekt spezifische Zugriffskontrollliste (ACL) konfigurieren.

Man bezeichnet sie als Subressourcen, weil sie im Kontext eines spezifischen Buckets oder Objekts existieren. Die folgende Tabelle listet Subressourcen auf, die ihnen ermöglichen, für den Bucket spezifische Konfigurationen zu verwalten.

Subressource	Beschreibung
cors (ursprungsübergreifende gemeinsame Nutzung von Ressourcen)	Sie können Ihren Bucket so konfigurieren, dass er ursprungsübergreifende Anforderungen zulässt. Weitere Informationen finden Sie unter Aktivieren von Cross-Origin Resource Sharing (CORS).
event notification	Sie können Ihrem Bucket gestatten, Benachrichtigungen über bestimmte Bucket-Ereignisse zu senden. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).
Lebenszyklus	Sie können Lebenszyklusregeln für Objekte in Ihrem Bucket mit definiertem Lebenszyklus definieren. Sie können z. B. eine Regel so festlegen, dass Objekte ein Jahr nach dem Erstellungsdatum archiviert werden, oder dass ein Objekt 10 Jahre nach dem Erstellen gelöscht wird. Weitere Informationen hierzu finden Sie im Abschnitt Objektlebenszyklusverwaltung.
location	Wenn Sie einen Bucket erstellen, geben Sie die AWS-Region an, in der Amazon S3 den Bucket erstellen soll. Amazon S3 speichert diese Informationen am Ort der Subressource und stellt Ihnen eine API bereit, mit der Sie diese Informationen abrufen können.
logging	Protokollierung ermöglicht Ihnen, Zugriffsanforderungen für Ihren Bucket nachzuverfolgen. Jeder Zugriffsprotokolldatensatz enthält Details über eine Zugriffsanforderung, z. B. Auftraggeber, Bucket-Name, Anforderungszeit, Anforderungsaktion, Antwortstatus und Fehlercode, falls vorhanden. Die Zugriffsprotokollinformationen können für Sicherheits- und Zugriffsüberprüfungen nützlich sein. Außerdem erfahren Sie damit mehr über Ihren Kundenstamm und verstehen Ihre Amazon S3-Rechnung. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).
Objektsperre	Um S3 Objektsperre verwenden zu können, muss die Funktion für einen Bucket aktiviert werden. Sie können optional auch einen Standardaufbewahrungsmodus und -zeitraum konfigurieren, der für neue Objekte gilt, die in den Bucket platziert werden. Weitere Informationen finden Sie unter Bucket-Konfiguration (p. 540).

Subressource	Beschreibung
policy und ACL (Access Control List, Zugriffskontrollliste)	Standardmäßig sind alle Ihre Ressourcen (wie Buckets und Objekte) privat. Amazon S3 unterstützt die Optionen Bucket-Richtlinie und Zugriffskontrollliste (ACL) für Sie, um Berechtigungen auf Bucket-Ebene zu erteilen und zu verwalten. Amazon S3 speichert die Berechtigungsinformationen in den Subressourcen policy und acl. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3 (p. 329).
Replikation	Die Replikation ist ein automatisches, asynchrones Kopieren von Objekten über Buckets hinweg in derselben oder in verschiedenen AWS-Regionen. Weitere Informationen finden Sie unter Replikation (p. 669).
requestPayment	Standardmäßig zahlt das AWS-Konto, das den Bucket erstellt (der Bucket-Eigentümer) für die Downloads aus dem Bucket. Mit dieser Subressource kann der Bucket-Eigentümer angeben, dass die Person, die den Download anfordert, die Gebühren für den Download trägt. Amazon S3 stellt eine API für Sie bereit, mit der Sie diese Subressource verwalten können. Weitere Informationen finden Sie unter Buckets mit Zahlung durch Auftraggeber (p. 82).
Markieren	Sie können Ihrem Bucket Kostenzuordnungs-Tags zur Kategorisierung und Nachverfolgung der AWS-Kosten hinzufügen. Amazon S3 stellt die Subressource tagging zum Speichern und Verwalten von Tags auf einem Bucket bereit. Mit Hilfe von Tags, die Sie auf Ihren Bucket anwenden, erzeugt AWS einen Kostenzuordnungsbericht mit Nutzungs- und Kostendaten, aggregiert nach Ihren Tags. Weitere Informationen finden Sie unter Fakturierungs- und Nutzungsberichte für S3-Buckets (p. 87).
transfer acceleration	Transfer Acceleration unterstützt schnelle, einfache und sichere Übertragungen von Dateien zwischen Ihrem Client und einem S3-Bucket über große Entfernungen. Transfer Acceleration nutzt die global verteilten Edge-Standorte von Amazon CloudFront. Weitere Informationen finden Sie unter Amazon S3 Transfer Acceleration (p. 75).
Versioning	Versioning hilft Ihnen bei einer Wiederherstellung nach einem versehentlichen Überschreiben und Löschen. Wir empfehlen das Versioning als bewährte Methode, um zu verhindern, dass Objekte versehentlich gelöscht oder überschrieben werden. Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514).
Website	Sie können Ihren Bucket für ein Hosting statischer Websites konfigurieren. Amazon S3 speichert diese Konfiguration, indem es eine Website-Subressource erstellt. Weitere Informationen finden Sie unter Hosting einer statischen Website auf Amazon S3.

Beschränkungen und Einschränkungen von Buckets

Ein Bucket gehört dem AWS-Konto, das ihn erstellt hat. Der Bucket-Besitz ist nicht übertragbar.

Wenn Sie einen Bucket erstellen, wählen Sie seinen Namen und die Region aus, in der er erstellt werden soll. Name oder Region einmal erstellter Buckets können nicht nachträglich geändert werden.

In der Standardeinstellung können Sie bis zu 100 Buckets in jedem Ihrer AWS-Konten erstellen. Wenn Sie weitere Buckets benötigen, können Sie Ihr Konto-Bucket-Limit auf maximal 1.000 Buckets erhöhen, indem Sie eine Service Limit-Erhöhung senden. Es gibt keinen Leistungsunterschied, ungeachtet dessen, ob Sie viele Buckets oder nur wenige verwenden. Weitere Informationen über die Vorgehensweise zum Erhöhen des Bucket-Kontingents finden Sie unter AWS-Servicekontingente in Allgemeine Referenz für AWS.

Wiederverwenden von Bucket-Namen

Wenn ein Bucket leer ist, können Sie ihn löschen. Nachdem ein Bucket gelöscht wurde, wird der Name zur Wiederverwendung verfügbar. Nachdem Sie den Bucket gelöscht haben, können Sie den Namen aus verschiedenen Gründen möglicherweise jedoch nicht wiederverwenden. Wenn Sie beispielsweise den Bucket löschen und der Name zur Wiederverwendung verfügbar wird, kann ein anderes Konto einen Bucket mit dem Namen erstellen. Außerdem kann einige Zeit vergehen, bis Sie den Namen eines gelöschten Buckets wiederverwenden können. Wenn Sie denselben Bucket-Namen verwenden möchten, empfehlen wir, den Bucket nicht zu löschen.

Objekte und Buckets

Die Anzahl der Objekte, die Sie in einem Bucket speichern können, ist nicht begrenzt. Sie können alle Ihre Objekte in einem einzigen Bucket speichern, oder sie über mehrere Buckets verteilen. Sie können jedoch keinen Bucket über einen anderen Bucket erstellen.

Bucket-Operationen

Das Hochverfügbarkeits-Engineering von Amazon S3 konzentriert sich auf get-, put-, list- und delete-Operationen. Bucket-Operationen arbeiten in einem zentralen, globalen Ressourcenraum, deshalb ist es nicht sinnvoll, Buckets über den Hochverfügbarkeits-Codepfad Ihrer Anwendung zu erstellen oder zu löschen. Sinnvoller ist es, Buckets in einer separaten Initialisierungs- oder Einrichtungsroutine zu erstellen oder zu löschen, die seltener ausgeführt wird.

Bucket-Benennung und automatisch erstellte Buckets

Wenn Ihre Anwendung automatisch Buckets erstellt, wählen Sie ein Bucket-Namensschema, das wahrscheinlich keine Namenskonflikte verursacht. Stellen Sie sicher, dass Ihre Anwendungslogik einen anderen Bucket-Namen auswählt, wenn ein Bucket-Name bereits vergeben ist.

Regeln für die Bucket-Benennung

Die folgenden Regeln gelten für die Benennung von S3-Buckets:

- Bucket-Namen müssen zwischen 3 und 63 Zeichen lang sein.
- Bucket-Namen können nur aus Kleinbuchstaben, Zahlen, Punkten (.) und Bindestrichen (-) bestehen.
- Bucket-Namen müssen mit einem Buchstaben oder einer Zahl beginnen und enden.
- Bucket-Namen dürfen nicht als IP-Adresse formatiert sein (zum Beispiel 192.168.5.4).
- Bucket-Namen müssen innerhalb einer Partition eindeutig sein. Eine Partition ist eine Gruppierung von Regionen. AWS verfügt derzeit über drei Partitionen: `aws` (Standardregionen), `aws-cn` (China-Regionen) und `aws-us-gov` (AWS GovCloud [USA] Regionen).

- Buckets, die mit Amazon S3 Transfer Acceleration verwendet werden, können keine Punkte (.) in ihren Namen enthalten. Weitere Informationen zur Transfer Acceleration finden Sie unter Amazon S3 Transfer Acceleration (p. 75).

Aus Gründen der besten Kompatibilität empfehlen wir, Punkte (.) in Bucket-Namen zu vermeiden, mit Ausnahme von Buckets, die nur für statisches Website-Hosting verwendet werden. Wenn Sie Punkte in den Namen eines Buckets einfügen, können Sie keine Adressierung im Stil virtueller Hosts über HTTPS verwenden, es sei denn, Sie führen eine eigene Zertifikatvalidierung durch. Dies liegt daran, dass die Sicherheitszertifikate, die für das virtuelle Hosten von Buckets verwendet werden, nicht für Buckets mit Punkten in ihren Namen funktionieren.

Diese Einschränkung wirkt sich nicht auf Buckets aus, die für das Hosten statischer Websites verwendet werden, da das Hosten von statischen Websites nur über HTTP verfügbar ist. Weitere Informationen zur Adressierung im Stil virtueller Hosts finden Sie unter Virtuelles Hosting bei Buckets (p. 48). Weitere Hinweise zum Hosten statischer Websites finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).

> **Note**
>
> Vor dem 1. März 2018 konnten Buckets, die in der Region USA Ost (Nord-Virginia) erstellt wurden, Namen mit bis zu 255 Zeichen und mit Großbuchstaben und Unterstrichen haben. Ab dem 1. März 2018 müssen neue Buckets in USA Ost (Nord-Virginia) den gleichen Regeln entsprechen, die in allen anderen Regionen angewendet werden.

Example Bucket-Namen

Die folgenden Beispielnamen für Buckets sind gültig und folgen den empfohlenen Benennungsrichtlinien:

- `awsexamplebucket1`
- `log-delivery-march-2020`
- `my-hosted-content`

Die folgenden Beispiel-Bucket-Namen sind gültig, aber nicht für andere Verwendungszwecke als statisches Website-Hosting empfohlen:

- `awsexamplewebsite.com`
- `www.awsexamplewebsite.com`
- `my.example.s3.bucket`

Die folgenden Beispiel-Bucket-Namen sind ungültig:

- `aws_example_bucket` (enthält Unterstriche)
- `AwsExampleBucket` (enthält Großbuchstaben)
- `aws-example-bucket-` (endet mit einem Bindestrich)

Beispiel zum Erstellen eines Buckets

Themen

- Verwenden anderer AWS SDKs (p. 68)

Die folgenden Codebeispiele erstellen einen Bucket programmgesteuert mit den AWS SDKs for Java, .NET und Ruby. Das Codebeispiel führt die folgenden Aufgaben durch:

- Erstellen eines Buckets, sofern noch keiner vorhanden ist – Das Beispiel erstellt einen Bucket durch Ausführen der folgenden Aufgaben:
  - Erstellen eines Clients durch explizite Angabe einer AWS-Region (das Beispiel verwendet die Region `s3.eu-west-1`). Dementsprechend kommuniziert der Client unter Verwendung des `s3.eu-west-1.amazonaws.com`-Endpunkts mit Amazon S3. Sie können eine beliebige andere AWS-Region angeben. Eine Liste der AWS-Regionen finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS-Referenz.
  - Senden einer Bucket-Anforderung, indem nur ein Bucket-Name angegeben wird. die Anforderung zum Erstellen eines Buckets gibt keine andere AWS-Region an. Der Client sendet eine Anforderung an Amazon S3, um den Bucket in der Region zu erstellen, die Sie beim Erstellen des Client angegeben haben. Sobald Sie einen Bucket erstellt haben, können Sie seine Region nicht mehr ändern.

    Note

    Wenn Sie in der Anforderung zum Erstellen eines Buckets explizit eine AWS-Region angeben, die von der beim Erstellen des Clients angegebenen Region abweicht, wird möglicherweise ein Fehler zurückgegeben. Weitere Informationen finden Sie unter Erstellen eines Buckets (p. 57).

  Die SDK-Bibliotheken senden die PUT Bucket-Anforderung an Amazon S3, um den Bucket zu erstellen. Weitere Informationen finden Sie unter PUT Bucket.
- Abrufen von Informationen zum Standort des Buckets – Amazon S3 speichert Informationen zum Standort des Buckets in der Subressource location, die dem Bucket zugeordnet ist. Die SDK-Bibliotheken senden die GET Bucket location-Anforderung (vgl. GET Bucket location), um diese Informationen abzurufen.

Verwenden der Amazon S3-Konsole

Informationen zum Erstellen eines Buckets mit der Amazon S3-Konsole finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwendung von AWS SDK for Java

Example

Dieses Beispiel veranschaulicht, wie Sie einen Amazon S3-Bucket mit AWS SDK for Java erstellen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CreateBucketRequest;
import com.amazonaws.services.s3.model.GetBucketLocationRequest;

import java.io.IOException;
```

```
public class CreateBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            if (!s3Client.doesBucketExistV2(bucketName)) {
                // Because the CreateBucketRequest object doesn't specify a region, the
                // bucket is created in the region specified in the client.
                s3Client.createBucket(new CreateBucketRequest(bucketName));

                // Verify that the bucket was created by retrieving it and checking its
 location.
                String bucketLocation = s3Client.getBucketLocation(new
 GetBucketLocationRequest(bucketName));
                System.out.println("Bucket location: " + bucketLocation);
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwendung von AWS SDK für .NET

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.S3.Util;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CreateBucketTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CreateBucketAsync().Wait();
        }
```

```
        static async Task CreateBucketAsync()
        {
            try
            {
                if (!(await AmazonS3Util.DoesS3BucketExistAsync(s3Client, bucketName)))
                {
                    var putBucketRequest = new PutBucketRequest
                    {
                        BucketName = bucketName,
                        UseClientRegion = true
                    };

                    PutBucketResponse putBucketResponse = await
 s3Client.PutBucketAsync(putBucketRequest);
                }
                // Retrieve the bucket location.
                string bucketLocation = await FindBucketLocationAsync(s3Client);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
        static async Task<string> FindBucketLocationAsync(IAmazonS3 client)
        {
            string bucketLocation;
            var request = new GetBucketLocationRequest()
            {
                BucketName = bucketName
            };
            GetBucketLocationResponse response = await
 client.GetBucketLocationAsync(request);
            bucketLocation = response.Location.ToString();
            return bucketLocation;
        }
    }
}
```

Verwenden des AWS SDK für Ruby Version 3

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwenden von AWS SDK für Ruby – Version 3 (p. 813).

Example

```
require 'aws-sdk-s3'

s3 = Aws::S3::Client.new(region: 'us-west-2')
s3.create_bucket(bucket: 'bucket-name')
```

Verwenden anderer AWS SDKs

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Löschen oder Leeren von Buckets

In einigen Situationen müssen Sie vielleicht einen leeren Bucket mit Objekten löschen oder leeren. In diesem Abschnitt erklären wir, wie Sie Objekte in einem nicht-versionierten Bucket und wie Sie Objektversionen und Löschmarkierungen in einem Bucket mit aktiviertem Versioning löschen. Weitere Informationen über Versioning finden Sie unter Verwenden von Versioning (p. 514). In einigen Situationen kann es sinnvoll sein, einen Bucket zu leeren, statt ihn zu löschen. Dieser Abschnitt erklärt verschiedene Optionen, die Sie nutzen können, um einen leeren Bucket mit Objekten zu löschen oder zu leeren.

Themen

Löschen eines Buckets

Sie können einen Bucket und seinen Inhalt programmgesteuert mithilfe der AWS-SDKs löschen. Sie können auch eine Lebenszykluskonfiguration für einen Bucket verwenden, um seinen Inhalt zu leeren und ihn dann zu löschen. Es gibt noch weitere Optionen, wie beispielsweise die Verwendung der Amazon S3-Konsole und der AWS-CLI, aber für diese Methode gelten abhängig von der Anzahl der Objekte in Ihrem Bucket und dem Versioning-Status des Buckets Einschränkungen.

Themen

Löschen eines Buckets: Verwenden der Amazon S3-Konsole

Die Amazon S3-Konsole unterstützt das Löschen eines Buckets, das leer sein kann, aber nicht muss. Weitere Informationen über die Verwendung der Amazon S3-Konsole zum Löschen eines Buckets finden Sie unter Wie lösche ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Löschen eines Buckets: Verwenden der AWS CLI

Einen Bucket, der Objekte enthält, können Sie mit der AWS CLI nur dann löschen, wenn für den Bucket das Versioning nicht aktiviert ist. Wenn für Ihren Bucket das Versioning nicht aktiviert ist, können Sie den AWS CLI-Befehl `rb` (remove bucket) mit dem Parameter `--force` verwenden, um einen nicht leeren Bucket zu löschen. Dieser Befehl löscht zuerst alle Objekte und dann den Bucket.

```
$ aws s3 rb s3://bucket-name --force
```

Weitere Informationen finden Sie unter Verwenden von S3-High-Level-Befehlen mit der AWS Command Line Interface im Benutzerhandbuch für AWS Command Line Interface.

Löschen eines Buckets: Verwenden der AWS SDKs

Sie können einen Bucket mit den AWS SDKs löschen. Die folgenden Abschnitte zeigen Beispiele für das Löschen eines Buckets mit AWS SDK for Java und .NET. Zuerst löscht der Code Objekte im Bucket, und dann löscht er das Bucket. Weitere Informationen über andere AWS-SDKs finden Sie unter Tools for Amazon Web Services.

Löschen eines Buckets mit dem AWS SDK for Java

Das folgende Java-Beispiel löscht einen Bucket mit Objekten. Zuerst löscht das Beispiel alle Objekte und dann den Bucket. Das Beispiel funktioniert für Buckets mit oder ohne aktiviertem Versioning.

> **Note**
>
> Bei Buckets ohne aktiviertes Versioning können Sie alle Objekte direkt löschen und danach den Bucket löschen. Bei Buckets mit aktiviertem Versioning müssen Sie zuerst alle Objektversionen löschen, bevor Sie den Bucket löschen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.Iterator;

public class DeleteBucket {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Delete all objects from the bucket. This is sufficient
            // for unversioned buckets. For versioned buckets, when you attempt to delete
 objects, Amazon S3 inserts
            // delete markers for all objects, but doesn't delete the object versions.
            // To delete objects from versioned buckets, delete all of the object versions
 before deleting
            // the bucket (see below for an example).
            ObjectListing objectListing = s3Client.listObjects(bucketName);
            while (true) {
                Iterator<S3ObjectSummary> objIter =
 objectListing.getObjectSummaries().iterator();
                while (objIter.hasNext()) {
                    s3Client.deleteObject(bucketName, objIter.next().getKey());
                }

                // If the bucket contains many objects, the listObjects() call
                // might not return all of the objects in the first listing. Check to
                // see whether the listing was truncated. If so, retrieve the next page of
 objects
                // and delete them.
                if (objectListing.isTruncated()) {
                    objectListing = s3Client.listNextBatchOfObjects(objectListing);
                } else {
                    break;
                }
            }

            // Delete all object versions (required for versioned buckets).
```

```
            VersionListing versionList = s3Client.listVersions(new
 ListVersionsRequest().withBucketName(bucketName));
            while (true) {
                Iterator<S3VersionSummary> versionIter =
 versionList.getVersionSummaries().iterator();
                while (versionIter.hasNext()) {
                    S3VersionSummary vs = versionIter.next();
                    s3Client.deleteVersion(bucketName, vs.getKey(), vs.getVersionId());
                }

                if (versionList.isTruncated()) {
                    versionList = s3Client.listNextBatchOfVersions(versionList);
                } else {
                    break;
                }
            }

            // After all objects and object versions are deleted, delete the bucket.
            s3Client.deleteBucket(bucketName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client couldn't
            // parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Leeren eines Buckets

Sie können einen Bucket und seinen Inhalt programmgesteuert unter Verwendung des AWS SDK leeren (d. h. Sie löschen den gesamten Inhalt, behalten aber den Bucket bei). Sie können auch eine Lebenszykluskonfiguration für einen Bucket angeben, sodass Objekte ablaufen, damit Amazon S3 sie löschen kann. Es gibt noch weitere Optionen, wie beispielsweise die Verwendung der Amazon S3-Konsole und der AWS CLI, aber diese Methode hat abhängig von der Anzahl der Objekte in Ihrem Bucket und dem Versioning-Status des Buckets Einschränkungen.

Themen

Leeren eines Buckets: Verwenden der Amazon S3-Konsole

Weitere Informationen über die Verwendung der Amazon S3-Konsole zum Leeren eines Buckets finden Sie unter Wie leere ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Leeren eines Buckets: Verwenden der AWS CLI

Einen Bucket nur dann mit der AWS CLI leeren, wenn für den Bucket das Versioning nicht aktiviert ist. Wenn für Ihren Bucket das Versioning nicht aktiviert ist, können Sie den AWS CLI-Befehl `rm` (remove bucket) mit dem Parameter `--recursive` verwenden, um einen Bucket zu leeren (oder eine Untermenge von Objekten mit einem spezifischen Schlüsselnamenpräfix zu entfernen).

Der folgende `rm`-Befehl entfernt Objekte mit dem Schlüsselnamenpräfix `doc`, z. B. `doc/doc1` und `doc/doc2`.

```
$ aws s3 rm s3://bucket-name/doc --recursive
```

Verwenden Sie den folgenden Befehl, um alle Objekte zu entfernen, ohne ein Präfix anzugeben.

```
$ aws s3 rm s3://bucket-name --recursive
```

Weitere Informationen finden Sie unter Verwenden von S3-High-Level-Befehlen mit der AWS Command Line Interface im Benutzerhandbuch für AWS Command Line Interface.

> Note
>
> Aus einem Bucket mit aktiviertem Versioning können Sie keine Objekte entfernen. Amazon S3 fügt eine Löschmarkierung hinzu, wenn Sie ein Objekt löschen. Das macht dieser Befehl. Weitere Informationen über Versioning finden Sie unter Verwenden von Versioning (p. 514).

Leeren eines Buckets: Verwenden der Lebenszykluskonfiguration

Sie können den Lebenszyklus für Ihren Bucket konfigurieren, sodass Objekte ablaufen. Amazon S3 kann dann aufgefordert werden, abgelaufene Objekte zu löschen. Sie können Lebenszykluskonfigurationsregeln hinzufügen, sodass alle Objekte oder eine Untermenge davon mit einem spezifischen Schlüsselnamenpräfix ablaufen. Sie können z. B. eine Lebenszyklusregel so festlegen, dass Objekte einen Tag nach dem Erstellungsdatum ablaufen, um alle Objekte in einem Bucket zu entfernen.

Wenn für Ihren Bucket das Versioning aktiviert ist, können Sie die Regel auch so konfigurieren, dass nicht aktuelle Objekte ablaufen. Wenn Sie den Inhalt eines Versioning-fähigen Buckets vollständig leeren möchten, müssen Sie eine Ablaufrichtlinie für aktuelle und nicht aktuelle Objekte im Bucket erstellen.

Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139) und Grundlegendes zum Objektablauf (p. 146).

Leeren eines Buckets: Verwenden der AWS SDKs

Sie können die AWS SDKs verwenden, um einen Bucket zu leeren oder eine Untermenge von Objekten mit einem spezifischen Schlüsselnamenpräfix zu entfernen.

Ein Beispiel zum Leeren eines Buckets mit dem AWS SDK for Java finden Sie unter Löschen eines Buckets mit dem AWS SDK for Java (p. 70). Der Code löscht alle Objekte, unabhängig davon, ob für den Bucket das Versioning aktiviert ist. Anschließend löscht er den Bucket. Um den Bucket nur zu leeren, stellen Sie sicher, dass Sie die Anweisung entfernen, die den Bucket löscht.

Weitere Informationen über die Verwendung anderer AWS-SDKs finden Sie unter Tools für Amazon Web Services.

Amazon S3-Standardverschlüsselung für S3-Buckets

Die Amazon S3-Standardverschlüsselung bietet eine Methode zum Festlegen des Verhaltens der Standardverschlüsselung für einen S3-Bucket. Sie können die Standardverschlüsselung für einen Bucket so festlegen, dass alle neuen Objekte verschlüsselt werden, wenn sie im Bucket gespeichert werden. Die Verschlüsselung der Objekte erfolgt serverseitig mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) oder Kundenstammschlüsseln (CMKs), die in AWS Key Management Service (AWS KMS) gespeichert sind.

Wenn Sie die serverseitige Verschlüsselung verwenden, verschlüsselt Amazon S3 ein Objekt vor dem Speichern auf der Festplatte und entschlüsselt es beim Herunterladen der Objekte. Weitere Informationen zum Schutz von Daten mithilfe der serverseitigen Verschlüsselung und der Verwaltung der Verschlüsselungsschlüssel finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290).

Themen

Wie richte ich die Amazon S3-Standardverschlüsselung für einen S3-Bucket ein?

Dieser Abschnitt beschreibt, wie die Amazon S3-Standardverschlüsselung eingerichtet wird. Sie können die AWS-SDKs, die Amazon S3-REST-API, die AWS Command Line Interface (AWS CLI) oder die Amazon S3-Konsole verwenden, um die Standardverschlüsselung zu aktivieren. Die einfachste Möglichkeit zum Einrichten der Standardverschlüsselung für einen S3-Bucket bietet die AWS Management Console.

Um die Standardverschlüsselung für einen Bucket einzurichten, können Sie jede dieser Methoden verwenden:

- Verwendung der Amazon S3-Konsole. Weitere Informationen finden Sie unter Wie aktiviere ich die Standardverschlüsselung für einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
- Verwenden Sie die REST API PUT Bucket encryption-Operation, um die Standardverschlüsselung zu aktivieren und die Art der serverseitigen Verschlüsselung (SSE-S3 oder SSE-KMS) festzulegen.
- Verwenden Sie die AWS CLI und die AWS SDKs. Weitere Informationen finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).

Nachdem Sie die Standardverschlüsselung für einen Bucket aktiviert haben, gilt das folgende Verschlüsselungsverhalten:

- Es gibt keine Änderung der Verschlüsselung der Objekte, die vor der Aktivierung der Standardverschlüsselung im Bucket vorhanden waren.
- Wenn Sie Objekte nach der Aktivierung der Standardverschlüsselung hochladen:
  - Wenn die Header Ihrer `PUT`-Anforderung keine Verschlüsselungsinformationen enthalten, verwendet Amazon S3 die Standardverschlüsselungseinstellungen des Buckets, um die Objekte zu verschlüsseln.
  - Wenn die Header Ihrer `PUT`-Anforderung Verschlüsselungsinformationen enthalten, verwendet Amazon S3 die Verschlüsselungsinformationen der `PUT`-Anforderung, um Objekte zu verschlüsseln, bevor sie in Amazon S3 gespeichert werden.
- Wenn Sie die SSE-KMS-Option für Ihre Standardverschlüsselungskonfiguration verwenden, unterliegen Sie den Limits der Anforderungen pro Sekunde (RPS) von AWS KMS. Weitere Informationen zu den AWS KMS-Limits und der Anforderung einer Erhöhung des Limits finden Sie unter AWS KMS-Limits.

Sie können Ihre vorhandenen Amazon S3-Objekte mithilfe der Amazon S3-Stapeloptionen mit einer einzelnen Anforderung verschlüsseln. Sie stellen S3 Stapeloperationen eine Liste von Objekten zur Verfügung, für die Operationen ausgeführt werden sollen, und Stapeloperationen ruft die entsprechende API auf, um den angegebenen Vorgang auszuführen. Mit dem Kopiervorgang können Sie vorhandenen

nicht verschlüsselte Objekte kopieren und die neuen verschlüsselten Objekte in denselben Bucket schreiben. Ein einziger Stapeloperationen-Auftrag kann die festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen.

Note

Amazon S3-Buckets mit Standard-Bucket-Verschlüsselung mittels SSE-KMS können nicht als Ziel-Buckets für Server access logging (Server-Zugriffsprotokollierung) (p. 777) verwendet werden. Für Zielbuckets des Serverzugriffsprotokolls wird nur die Standardverschlüsselung SSE-S3 unterstützt.

Verwenden von Verschlüsselung für kontenübergreifende Operationen

Beachten Sie Folgendes, wenn Sie kontoübergreifende Operationen verschlüsseln:

- Der AWS/s3 AWS-verwaltete CMK wird verwendet, wenn weder zum Zeitpunkt der Anforderung noch über die Standard-Verschlüsselungskonfiguration des Buckets ein CMK-ARN oder -Alias angegeben wird.
- Wenn Sie Ihren eigenen CMK angeben, sollten Sie einen vollqualifizierten CMK-Schlüssel-ARN verwenden. Beachten Sie bei der Verwendung eines CMK-Alias, dass KMS den Schlüssel innerhalb des Kontos des Anforderers auflöst. Dies kann dazu führen, dass Daten mit einem CMK verschlüsselt werden, der dem Anforderer und nicht dem Bucket-Administrator gehört.
- Sie müssen einen Schlüssel angeben, für den Ihnen (dem Anforderer) die Berechtigung `Encrypt` erteilt wurde. Weitere Informationen finden Sie unter Ermöglicht Schlüsselbenutzern die Verwendung eines CMK für kryptografische Operationen.

Verwenden der Standardverschlüsselung mit der Replikation

Nachdem Sie die Standardverschlüsselung für einen Replikations-Ziel-Bucket aktiviert haben, gilt das folgende Verschlüsselungsverhalten:

- Wenn Objekte im Quell-Bucket nicht verschlüsselt sind, werden die Replikatobjekte im Ziel-Bucket mithilfe der Einstellungen der Standardverschlüsselung des Ziel-Buckets verschlüsselt. Daher unterscheidet sich das `ETag` des Quell-Objekts von dem `ETag` des Replikatobjekts. Sie müssen die Anwendungen, die das `ETag` verwenden, zur Anpassung an diesen Unterschied aktualisieren.

- Wenn Objekte im Quell-Bucket mithilfe von SSE-S3 oder SSE-KMS verschlüsselt werden, verwenden die Replikatobjekte im Ziel-Bucket dieselbe Verschlüsselung wie die Verschlüsselung des Quellobjekts. Die Einstellungen der Standardverschlüsselung des Ziel-Buckets werden nicht verwendet.

Weitere Informationen über die Verwendung der Standardverschlüsselung mit SSE-KMS finden Sie unter Replizieren verschlüsselter Objekte (p. 693).

Überwachen der Standardverschlüsselung mit CloudTrail und CloudWatch

Sie können die KonfigurationsAnforderungen der Standardverschlüsselung über AWS CloudTrail-Ereignisse verfolgen. Die API-Ereignisnamen, die in CloudTrail-Protokollen verwendet werden, sind `PutBucketEncryption`, `GetBucketEncryption` und `DeleteBucketEncryption`. Zudem können

Sie Amazon CloudWatch Events mit Operationen auf S3-Bucket-Ebene als Ereignistyp erstellen. Weitere Informationen zu den CloudTrail-Ereignissen finden Sie unter Wie aktiviere ich die Protokollierung auf Objektebene für einen S3-Bucket mit CloudTrail-Datenereignissen?

Sie können CloudTrail-Protokolle für Amazon S3-Aktionen auf Objektebene für die Verfolgung von `PUT`- und `POST`-Anforderungen an Amazon S3 verwenden, um zu überprüfen, ob die Standardverschlüsselung für das Verschlüsseln von Objekten verwendet wird, wenn eingehende `PUT`-Anforderungen über keine Verschlüsselungs-Header verfügen.

Wenn Amazon S3 ein Objekt mit den Einstellungen der Standardverschlüsselung verschlüsselt, enthält das Protokoll das folgende Feld als Name-Wert-Paar: `"SSEApplied":"Default_SSE_S3" or "SSEApplied":"Default_SSE_KMS"`.

Wenn Amazon S3 ein Objekt mit den `PUT`-Verschlüsselungs-Headern verschlüsselt, enthält das Protokoll das folgende Feld als Name-Wert-Paar: `"SSEApplied":"SSE_S3", "SSEApplied":"SSE_KMS` oder `"SSEApplied":"SSE_C"`. Diese Informationen sind für mehrteilige Uploads in den `InitiateMultipartUpload`-API-Anforderungen enthalten. Weitere Informationen zur Verwendung von CloudTrail und CloudWatch finden Sie unter Überwachung von Amazon S3 (p. 738).

Weitere Infos

- PUT-Bucket-Verschlüsselung
- DELETE-Bucket-Verschlüsselung
- GET-Bucket-Verschlüsselung

Amazon S3 Transfer Acceleration

Amazon S3 Transfer Acceleration unterstützt schnelle, einfache und sichere Übertragungen von Dateien zwischen Ihrem Client und einem S3-Bucket über große Entfernungen. Transfer Acceleration nutzt die global verteilten Edge-Standorte von Amazon CloudFront. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Ihren Amazon S3 -Bucket weitergeleitet.

Bei Verwendung von Transfer Acceleration können zusätzliche Gebühren für die Datenübertragung anfallen. Weitere Informationen zu Preisen finden Sie unter Amazon S3-Preise.

Themen

Warum Amazon S3 Transfer Acceleration verwenden?

Es gibt verschiedene Gründe für die Verwendung von Transfer Acceleration auf einem Bucket. Dazu gehören:

- Sie haben Kunden, die Uploads in einen zentralen Bucket aus der ganzen Welt vornehmen.
- Sie übertragen regelmäßig mehrere Gigabyte bis Terabyte von Daten über mehrere Kontinente hinweg.
- Sie nutzen beim Hochladen in Amazon S3 die verfügbare Bandbreite über das Internet nicht.

Weitere Informationen darüber, wann Sie Transfer Acceleration verwenden sollten, finden Sie unter Amazon S3-FAQs.

Verwendung des Amazon S3 Transfer Acceleration Speed Comparison-Tools

Sie können das Amazon S3 Transfer Acceleration Speed Comparison Tool verwenden, um beschleunigte und nicht beschleunigte Upload-Geschwindigkeiten über Amazon S3-Regionen hinweg zu vergleichen. Das Speed Comparison Tool verwendet mehrteilige Uploads, um eine Datei von Ihrem Browser in verschiedene Amazon S3-Regionen mit und ohne Verwendung von Transfer Acceleration zu übertragen.

Sie können mit einer der folgenden Methoden auf das Speed Comparison-Tool zugreifen:

- Kopieren Sie die folgende URL in Ihr Browser-Fenster, und ersetzen Sie *region* durch die von Ihnen verwendete Region (z. B. us-west-2), und *yourBucketName* durch den Namen des auszuwertenden Buckets:

  ```
  https://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparsion.html?region=region&origBucketName=yourBucketName
  ```

  Eine Liste der von Amazon S3 unterstützten Regionen finden Sie unter Regionen und Endpunkte im Allgemeine Amazon Web Services-Referenz.
- Verwendung der Amazon S3-Konsole. Für Einzelheiten vgl. Aktivieren von Transfer Acceleration im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Erste Schritte mit Amazon S3 Transfer Acceleration

Für die ersten Schritte bei der Verwendung von Amazon S3 Transfer Acceleration gehen Sie wie folgt vor:

1. Aktivieren von Transfer Acceleration für einen Bucket – Damit Ihr Bucket mit Transfer Acceleration arbeiten kann, muss sein name den DNS-Benennungsanforderungen entsprechen und darf keine Punkte („.“) enthalten.

   Sie können Transfer Acceleration wie folgt für einen Bucket aktivieren:

   - Verwendung der Amazon S3-Konsole. Weitere Informationen finden Sie unter Aktivieren von Transfer Acceleration im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
   - Verwendung der REST API PUT Bucket accelerate-Operation.
   - Verwendung der AWS CLI und AWS SDKs. Weitere Informationen finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).
2. Übertragung von Daten in und aus dem beschleunigungsfähigen Bucket durch einen der folgenden s3-accelerate Endpunkt-Domänennamen:

   - *bucketname*`.s3-accelerate.amazonaws.com` – für den Zugriff auf einen beschleunigungsfähigen Bucket.
   - *bucketname*`.s3-accelerate.dualstack.amazonaws.com` – für den Zugriff auf einen beschleunigungsfähigen Bucket über IPv6. Amazon S3-Dual-Stack-Endpunkte unterstützen Anforderungen an S3-Buckets über IPv6 und IPv4. Der Transfer Acceleration Dual-Stack-Endpunkt verwendet nur den virtuellen Hosting-Endpunktnamen. Weitere Informationen finden Sie unter Erste Schritte für Anforderungen über IPv6 (p. 13) und Verwenden von Amazon S3-Dual-Stack-Endpunkten (p. 16).

     Important

     Derzeit unterstützt nur das AWS Java SDK den beschleunigten Dual-Stack-Endpunkt. Die Unterstützung der AWS CLI und anderer AWS SDKs ist in Kürze verfügbar.

Note

Sie können neben den beschleunigten Endpunkten weiterhin den regulären Endpunkt verwenden.

Sie können in Ihren Amazon S3- PUT object- und GET object-Anfragen auf den s3-accelerate Endpunkt-Domänennamen verweisen, nachdem Sie Transfer Acceleration aktiviert haben. Angenommen, Sie haben derzeit eine REST API-Anwendung, die PUT Object mit dem Hostnamen mybucket.s3.us-east-1.amazonaws.com in der `PUT`-Anforderung verwendet. Um `PUT` zu beschleunigen, ändern Sie einfach den Hostnamen in Ihrer Anfrage zu mybucket.s3-accelerate.amazonaws.com. Um wieder die Standardgeschwindigkeit für den Upload zu verwenden, ändern Sie einfach den Namen wieder zurück zu mybucket.s3.us-east-1.amazonaws.com.

Nachdem Transfer Acceleration aktiviert wurde, kann es bis zu 20 Minuten dauern, bis Sie den Leistungsvorteil wahrnehmen. Der beschleunigte Endpunkt steht jetzt zur Verfügung, sobald Sie Transfer Acceleration aktivieren.

Sie können den beschleunigten Endpunkt in der AWS CLI, den AWS SDKs und anderen Tools nutzen, die Daten in und von Amazon S3 übertragen. Wenn Sie die AWS SDKs verwenden, verwenden einige der unterstützten Sprachen ein Flag für eine Client-Konfiguration mit beschleunigtem Endpunkt, Sie müssen also den Endpunkt für Transfer Acceleration nicht explizit auf *`bucketname`*.s3-accelerate.amazonaws.com setzen. Beispiele für das Flag für eine Client-Konfiguration mit beschleunigtem Endpunkt finden Sie unter Beispiele für Amazon S3 Transfer Acceleration (p. 78).

Sie können alle Amazon S3-Operationen durch die übertragungsbeschleunigten Endpunkte verwenden, außer den folgenden:GET Service (list buckets), PUT Bucket (create bucket) und DELETE Bucket. Darüber hinaus unterstützt Amazon S3 Transfer Acceleration keine regionsübergreifenden Kopien mit PUT Object - Copy.

Voraussetzungen für die Verwendung von Amazon S3 Transfer Acceleration

Die folgenden Anforderungen gelten für die Verwendung von Transfer Acceleration für einen S3-Bucket:

- Transfer Acceleration wird nur in virtuellen Anfragen unterstützt. Weitere Informationen über virtuelle Anfragen finden Sie unter Senden von Anforderungen unter Verwendung der REST-API (p. 46).
- Der Name des für Transfer Acceleration verwendeten Buckets muss DNS-konform sein, und er darf keine Punkte (".") enthalten.
- Transfer Acceleration muss für den Bucket aktiviert sein. Nach der Aktivierung von Transfer Acceleration für einen Bucket kann es bis zu 20 Minuten dauern, bis sich die Datenübertragungsgeschwindigkeit in den Bucket erhöht.
- Für den Zugriff auf einen für Transfer Acceleration aktivierten Bucket müssen Sie den Endpunkt *`bucketname`*`.s3-accelerate.amazonaws.com` oder den Dual-Stack-Endpunkt *`bucketname`*`.s3-accelerate.dualstack.amazonaws.com` verwenden, um den aktivierten Bucket über IPv6 zu verbinden.
- Sie müssen der Bucket-Eigentümer sein, um den Transfer Acceleration-Status festlegen zu können. Der Bucket-Eigentümer kann anderen Benutzern Berechtigungen erteilen, um ihnen zu gestatten, den Beschleunigungsstatus für einen Bucket einzurichten. Die `s3:PutAccelerateConfiguration`-Berechtigung gestattet Benutzern, Transfer Acceleration für einen Bucket zu aktivieren oder zu deaktivieren. Die `s3:GetAccelerateConfiguration`-Berechtigung gestattet Benutzern, den Transfer Acceleration-Status eines Buckets zurückzugeben, `Enabled` oder `Suspended`.. Weitere Informationen über diese Berechtigungen finden Sie unter Beispiel – Bucket-Subressourcen-Operationen (p. 381) und Identity and Access Management in Amazon S3 (p. 329).

Weitere Infos

- GET Bucket accelerate
- PUT Bucket accelerate

Beispiele für Amazon S3 Transfer Acceleration

Dieser Abschnitt bietet Beispiele für die Aktivierung von Amazon S3 Transfer Acceleration und die Verwendung des beschleunigten Endpunkts für den aktivierten Bucket. Wenn Sie die AWS SDK verwenden, verwenden einige der unterstützten Sprachen (z. B. Java und .NET) ein Flag für eine Client-Konfiguration mit beschleunigtem Endpunkt, Sie müssen also den Endpunkt für Transfer Acceleration nicht explizit auf *bucketname*.s3-accelerate.amazonaws.com setzen. Mehr über Transfer Acceleration erfahren Sie unter Amazon S3 Transfer Acceleration (p. 75).

Themen

Verwenden der Amazon S3-Konsole

Weitere Informationen über die Aktivierung von Transfer Acceleration für einen Bucket unter Verwendung der Amazon S3-Konsole finden Sie unter Aktivieren von Transfer Acceleration im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwendung von Transfer Acceleration mit AWS Command Line Interface (AWS CLI)

Dieser Abschnitt enthält Beispiele für AWS CLI-Befehle für Transfer Acceleration. Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Einrichten der AWS-CLI (p. 808).

Aktivierung von Transfer Acceleration für einen Bucket mit der AWS CLI

Verwenden Sie den AWS CLI-Befehl put-bucket-accelerate-configuration, um Transfer Acceleration für einen Bucket zu aktivieren oder auszusetzen. Das folgende Beispiel setzt `Status=Enabled` so, dass Transfer Acceleration für einen Bucket aktiviert wird. Sie verwenden `Status=Suspended`, um Transfer Acceleration auszusetzen.

Example

```
$ aws s3api put-bucket-accelerate-configuration --bucket bucketname --accelerate-
configuration Status=Enabled
```

Verwendung von Transfer Acceleration mit der AWS CLI

Bei der Einstellung des Konfigurationswerts `use_accelerate_endpoint` auf `true` in einem Profil in Ihrer AWS Config Datei werden alle Amazon S3-Anfragen von s3- und s3api AWS CLI-Befehlen an den

Beschleunigungsendpunkt geleitet: `s3-accelerate.amazonaws.com`. Transfer Acceleration muss auf Ihrem Bucket aktiviert sein, damit der Beschleunigungsendpunkt verwendet werden kann.

Alle Anfragen werden unter Verwendung der virtuellen Bucket-Adressierungsform gesendet `my-bucket.s3-accelerate.amazonaws.com`. `ListBuckets`-, `CreateBucket`- und `DeleteBucket`-Anfragen werden nicht an den Beschleunigungsendpunkt gesendet, weil der Endpunkt diese Operationen nicht unterstützt. Für weitere Informationen zu `use_accelerate_endpoint` vgl. AWS CLI-S3-Konfiguration.

Das folgende Beispiel setzt `use_accelerate_endpoint` im Standardprofil auf `true`.

Example

```
$ aws configure set default.s3.use_accelerate_endpoint true
```

Wenn Sie den beschleunigten Endpunkt für manche AWS CLI-Befehle, aber nicht für alle verwenden wollen, können Sie eine der beiden folgenden Methoden anwenden:

- Sie können den beschleunigten Endpunkt per Befehl verwenden, indem sie den `--endpoint-url` Parameter für jeden s3- oder s3api-Befehl auf `https://s3-accelerate.amazonaws.com` oder `http://s3-accelerate.amazonaws.com` setzen.
- Sie können in Ihrer AWS-Konfigurationsdatei separate Profil einrichten. Legen Sie beispielsweise ein Profil an, das `use_accelerate_endpoint` auf `true` setzt, und ein Profil, das `use_accelerate_endpoint` nicht setzt. Wenn Sie einen Befehl ausführen, geben Sie an, welches Profil Sie verwenden wollen, abhängig davon, ob Sie den beschleunigten Endpunkt verwenden wollen oder nicht.

AWS CLI Beispiele für das Hochladen eines Objekts in einen Bucket, der für Transfer Acceleration geeignet ist

Das folgende Beispiel lädt eine Datei in einen Bucket hoch, der für Transfer Acceleration konfiguriert ist. Dazu verwendet es das Standardprofil, das für die Verwendung des beschleunigten Endpunkts konfiguriert wurde.

Example

```
$ aws s3 cp file.txt s3://bucketname/keyname --region region
```

Das folgende Beispiel lädt eine Datei in einen Bucket hoch, der für Transfer Acceleration konfiguriert ist. Dazu verwendet es den Parameter `--endpoint-url` für die Angabe des Beschleunigungsendpunkts.

Example

```
$ aws configure set s3.addressing_style virtual
$ aws s3 cp file.txt s3://bucketname/keyname --region region --endpoint-url http://s3-accelerate.amazonaws.com
```

Verwendung von Transfer Acceleration aus AWS SDK for Java

Example

Das folgende Beispiel zeigt, wie Sie einen beschleunigten Endpunkt für das Hochladen eines Objekts in Amazon S3 verwenden. Das Beispiel erledigt Folgendes:

- Erstellt einen `AmazonS3Client`, der für die Verwendung von beschleunigten Endpunkten konfiguriert ist. Für alle Buckets, auf die der Client zugreift, muss Transfer Acceleration aktiviert sein.
- Aktiviert Transfer Acceleration auf einem angegebenen Bucket. Dieser Schritt ist nur erforderlich, wenn für den von Ihnen angegebenen Bucket nicht bereits Transfer Acceleration aktiviert ist.
- Überprüft, ob Transfer Acceleration für den angegebenen Bucket aktiviert ist.
- Lädt ein neues Objekt in den angegebenen Bucket hoch und verwendet dazu den beschleunigten Endpunkt des Buckets.

Weitere Informationen zur Verwendung von Transfer Acceleration finden Sie unter Erste Schritte mit Amazon S3 Transfer Acceleration (p. 76). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketAccelerateConfiguration;
import com.amazonaws.services.s3.model.BucketAccelerateStatus;
import com.amazonaws.services.s3.model.GetBucketAccelerateConfigurationRequest;
import com.amazonaws.services.s3.model.SetBucketAccelerateConfigurationRequest;

public class TransferAcceleration {
    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";

        try {
            // Create an Amazon S3 client that is configured to use the accelerate
 endpoint.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .enableAccelerateMode()
                    .build();

            // Enable Transfer Acceleration for the specified bucket.
            s3Client.setBucketAccelerateConfiguration(
                    new SetBucketAccelerateConfigurationRequest(bucketName,
                            new BucketAccelerateConfiguration(
                                    BucketAccelerateStatus.Enabled)));

            // Verify that transfer acceleration is enabled for the bucket.
            String accelerateStatus = s3Client.getBucketAccelerateConfiguration(
                    new GetBucketAccelerateConfigurationRequest(bucketName))
                    .getStatus();
            System.out.println("Bucket accelerate status: " + accelerateStatus);

            // Upload a new object using the accelerate endpoint.
            s3Client.putObject(bucketName, keyName, "Test object for transfer
 acceleration");
            System.out.println("Object \"" + keyName + "\" uploaded with transfer
 acceleration.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwendung von Transfer Acceleration vom AWS SDK for .NET aus

Das folgende Beispiel zeigt, wie Sie Transfer Acceleration mit AWS SDK für .NET für einen Bucket aktivieren. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TransferAccelerationTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            EnableAccelerationAsync().Wait();
        }

        static async Task EnableAccelerationAsync()
        {
                try
                {
                    var putRequest = new PutBucketAccelerateConfigurationRequest
                    {
                        BucketName = bucketName,
                        AccelerateConfiguration = new AccelerateConfiguration
                        {
                            Status = BucketAccelerateStatus.Enabled
                        }
                    };
                    await s3Client.PutBucketAccelerateConfigurationAsync(putRequest);

                    var getRequest = new GetBucketAccelerateConfigurationRequest
                    {
                        BucketName = bucketName
                    };
                    var response = await
 s3Client.GetBucketAccelerateConfigurationAsync(getRequest);

                    Console.WriteLine("Acceleration state = '{0}' ", response.Status);
                }
                catch (AmazonS3Exception amazonS3Exception)
                {
                    Console.WriteLine(
```

```
                        "Error occurred. Message:'{0}' when setting transfer acceleration",
                        amazonS3Exception.Message);
                }
            }
        }
    }
```

Wenn Sie ein Objekt in einen Bucket hochladen, für den Transfer Acceleration aktiviert ist, verwenden Sie für die Angabe des beschleunigten Endpunkts den Zeitpunkt, an dem ein Client erstellt wurde (siehe unten):

```
var client = new AmazonS3Client(new AmazonS3Config
            {
                RegionEndpoint = TestRegionEndpoint,
                UseAccelerateEndpoint = true
            }
```

Verwendung von Transfer Acceleration vom AWS SDK for JavaScript aus

Ein Beispiel für die Aktivierung von Transfer Acceleration durch das AWS SDK for JavaScript finden Sie unter Aufruf der Operation putBucketAccelerateConfiguration in der AWS SDK for JavaScript-API-Referenz.

Verwendung von Transfer Acceleration mit AWS SDK for Python (Boto)

Ein Beispiel für die Aktivierung von Transfer Acceleration mit SDK for Python finden Sie unter put_bucket_accelerate_configuration im AWS SDK for Python (Boto 3) API Reference.

Verwenden anderer AWS SDKs

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Buckets mit Zahlung durch Auftraggeber

Themen

Im Allgemeinen zahlen Bucket-Eigentümer für alle Amazon S3-Speicher- und Datenübertragungskosten, die ihrem Bucket zuzuordnen sind. Ein Bucket-Eigentümer kann jedoch einen Bucket für die Zahlung durch den Anforderer konfigurieren. Bei Buckets mit Zahlung durch den Anforderer zahlt der Auftraggeber statt des Bucket-Eigentümers die Kosten für die Anforderung und den Daten-Download aus dem Bucket. Der Bucket-Eigentümer zahlt immer die Kosten für das Speichern der Daten.

In der Regel konfigurieren Sie Buckets mit Zahlung durch den Anforderer, wenn Sie Daten teilen wollen, aber nicht die Gebühren übernehmen wollen, wenn andere auf die Daten zugreifen. Sie könnten beispielsweise Buckets mit Zahlung durch den Anforderer verwenden, wenn Sie große Datenmengen bereitstellen, wie beispielsweise Zip-Code-Verzeichnisse, Referenzdaten, Koordinatensystem Informationen oder Web-Crawling-Daten.

Important

Wenn Sie die Zahlung durch den Anforderer für einen Bucket aktivieren, ist kein anonymer Zugriff auf den Bucket zulässig.

Sie müssen alle Anforderungen für Buckets mit Zahlung durch den Anforderer authentifizieren. Die Anforderungsauthentifizierung ermöglicht Amazon S3, den Auftraggeber zu identifizieren und ihm seine Verwendung des Buckets mit Zahlung durch den Anforderer in Rechnung zu stellen.

Wenn der Auftraggeber eine AWS Identity and Access Management (IAM)-Rolle übernimmt, bevor er seine Anforderung stellt, wird das Konto, dem die Rolle gehört, mit den Gebühren belastet. Weitere Informationen zu IAM-Rollen finden Sie unter IAM -Rollen im IAM-Benutzerhandbuch.

Nachdem Sie ein Bucket als Bucket mit Zahlung durch den Anforderer konfiguriert haben, müssen die Auftraggeber `x-amz-request-payer` in ihre Anforderungen aufnehmen, entweder im Header, wenn es sich um POST-, GET- und HEAD-Anforderungen handelt, oder als Parameter in einer REST-Anforderung, um zu demonstrieren, dass sie wissen, dass ihnen die Gebühren für die Anforderung und den Daten-Download in Rechnung gestellt werden.

Buckets mit Zahlung durch den Anforderer unterstützen die folgenden Funktionen nicht.

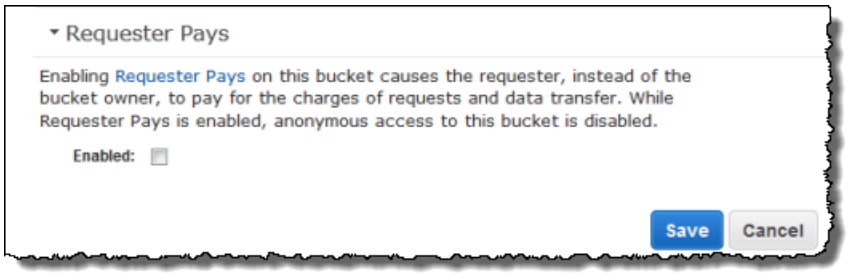
- Anonyme Anforderungen
- BitTorrent
- SOAP-Anforderungen
- Sie können einen Bucket mit Zahlung durch den Anforderer nicht als Ziel-Bucket für die Endbenutzerprotokollierung oder umgekehrt verwenden. Sie können jedoch eine Endbenutzerprotokollierung für einen Bucket mit Zahlung durch den Anforderer aktivieren, wenn der Ziel-Bucket kein Bucket mit Zahlung durch den Anforderer ist.

Konfigurieren des Zahlung durch den Anforderer unter Verwendung der Amazon S3-Konsole

Sie können einen Bucket als Bucket mit Zahlung durch den Anforderer unter Verwendung der Amazon S3-Konsole konfigurieren.

Einen Bucket als Bucket mit Zahlung durch den Anforderer konfigurieren

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Klicken Sie in der Liste Buckets auf das Detailsymbol links vom Bucket-Namen, und klicken Sie dann auf Properties (Eigenschaften), um die Bucket-Eigenschaften anzuzeigen.
3. Klicken Sie im Bereich Properties (Eigenschaften) auf Requester Pays (Zahlung durch den Anforderer).
4. Markieren Sie das Kontrollkästchen Enabled (Aktiviert).



Konfigurieren des Zahlung durch den Anforderer unter Verwendung der REST API

Themen

Einrichtung der Bucket-Konfiguration requestPaymentConfiguration

Nur der Bucket-Eigentümer kann den Konfigurationswert `RequestPaymentConfiguration.payer` eines Buckets auf `BucketOwner` setzen (das ist der Standardwert), oder auf `Requester`. Die Einrichtung der Ressource `requestPayment` ist optional. Standardmäßig ist der Bucket kein Bucket mit Zahlung durch den Anforderer.

Um einen Bucket mit Zahlung durch den Anforderer in einen regulären Bucket umzuwandeln, verwenden Sie den Wert `BucketOwner`. In der Regel verwenden Sie `BucketOwner` zum Hochladen von Daten in den Amazon S3-Bucket. Dann würden Sie den Wert auf `Requester` setzen, bevor Sie die Objekte in dem Bucket veröffentlichen.

Einrichten von requestPayment

- Verwenden Sie eine `PUT`-Anforderung, um den Wert `Payer` für einen bestimmten Bucket auf `Requester` zu setzen.

```
PUT ?requestPayment HTTP/1.1
Host: [BucketName].s3.amazonaws.com
Content-Length: 173
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]

<RequestPaymentConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Payer>Requester</Payer>
</RequestPaymentConfiguration>
```

War die Anforderung erfolgreich, gibt Amazon S3 eine Antwort zurück, die etwa wie folgt aussieht.

```
HTTP/1.1 200 OK
x-amz-id-2: [id]
x-amz-request-id: [request_id]
Date: Wed, 01 Mar 2009 12:00:00 GMT
Content-Length: 0
Connection: close
Server: AmazonS3
x-amz-request-charged:requester
```

Sie können die Zahlung durch den Anforderer nur auf Bucket-Ebene einrichten. Es ist nicht möglich, die Zahlung durch den Anforderer für bestimmte Objekte innerhalb des Buckets einzurichten.

Sie können einen Bucket jederzeit als `BucketOwner` oder `Requester` konfigurieren. Beachten Sie jedoch, dass es eine kleine Verzögerung in der Größenordnung von Minuten geben kann, bevor der neue Konfigurationswert in Kraft tritt.

Note

Bucket-Eigentümer, die vorsignierte URLs ausgeben, sollten gut darüber nachdenken, ob sie einen Bucket für die Zahlung durch den Anforderer konfigurieren, insbesondere dann, wenn die URL eine sehr lange Lebensdauer aufweist. Der Bucket-Eigentümer muss jedes Mal die Gebühren zahlen, wenn der Auftraggeber eine vorsignierte URL verwendet, die die Anmeldeinformationen des Bucket-Eigentümers verwendet.

Abruf der requestPayment-Konfiguration

Sie können den `Payer`-Wert ermitteln, der für einen Bucket eingerichtet ist, indem Sie die Ressource `requestPayment` abrufen.

Rückkehr zur requestPayment-Ressource

- Verwenden Sie eine GET-Anforderung, um die `requestPayment`-Ressource zu erhalten, wie in der folgenden Anforderung gezeigt.

```
GET ?requestPayment HTTP/1.1
Host: [BucketName].s3.amazonaws.com
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]
```

War die Anforderung erfolgreich, gibt Amazon S3 eine Antwort zurück, die etwa wie folgt aussieht.

```
HTTP/1.1 200 OK
x-amz-id-2: [id]
x-amz-request-id: [request_id]
Date: Wed, 01 Mar 2009 12:00:00 GMT
Content-Type: [type]
Content-Length: [length]
Connection: close
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<RequestPaymentConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Payer>Requester</Payer>
</RequestPaymentConfiguration>
```

Diese Antwort zeigt, dass der `payer`-Wert auf `Requester` gesetzt ist.

Herunterladen von Objekten in Buckets mit Zahlung durch den Anforderer

Den Auftraggebern werden die Gebühren für das Herunterladen von Daten aus Buckets mit Zahlung durch den Anforderer in Rechnung gestellt, deshalb müssen sie einen speziellen Parameter enthalten, `x-amz-request-payer`, der bestätigt, dass der Auftraggeber weiß, dass er Gebühren für den Download zahlen muss. Um auf Objekte in Buckets mit Zahlung durch den Anforderer zuzugreifen, müssen die Anforderungen eine der folgenden Informationen enthalten.

- Für GET-, HEAD- und POST-Anforderungen muss `x-amz-request-payer : requester` in den Header aufgenommen werden
- Für signierte URLs muss `x-amz-request-payer=requester` in die Anforderung aufgenommen werden

Ist die Anforderung erfolgreich und dem Auftraggeber werden Gebühren in Rechnung gestellt, enthält die Antwort den Header `x-amz-request-charged:requester`. Ist `x-amz-request-payer` nicht in der Anforderung enthalten, gibt Amazon S3 einen 403-Fehler zurück und stellt dem Bucket-Eigentümer die Gebühren für die Anforderung in Rechnung.

> **Note**
>
> Bucket-Eigentümer müssen ihren Anforderungen `x-amz-request-payer` nicht hinzufügen. Stellen Sie sicher, dass Sie `x-amz-request-payer` und seinen Wert in die Berechnung Ihrer Signatur aufgenommen haben. Weitere Informationen finden Sie unter Konstruieren des CanonicalizedAmzHeaders-Elements (p. 825).

Herunterladen von Objekten aus einem Bucket mit Zahlung durch den Anforderer

- Verwenden Sie eine `GET`-Anforderung, um ein Objekt aus einem Bucket mit Zahlung durch den Anforderer herunterzuladen, wie in der folgenden Anforderung gezeigt.

```
GET / [destinationObject] HTTP/1.1
Host: [BucketName].s3.amazonaws.com
x-amz-request-payer : requester
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]
```

Ist die GET-Anforderung erfolgreich und dem Auftraggeber werden Gebühren in Rechnung gestellt, enthält die Antwort `x-amz-request-charged:requester`.

Amazon S3 kann für Anforderungen, die versuchen, Objekte aus einem Bucket mit Zahlung durch den Anforderer abzurufen, einen `Access Denied`-Fehler zurückgeben. Weitere Informationen finden Sie unter Fehlermeldungen.

Gebührendetails

Die Gebühren für erfolgreiche Anforderungen mit Zahlung durch den Anforderer sind ganz einfach zu realisieren: der Auftraggeber zahlt für die Datenübertragung und die Anforderung; der Bucket-Eigentümer zahlt für die Speicherung der Daten. Dem Bucket-Eigentümer werden jedoch nur unter den folgenden Bedingungen Gebühren für die Anforderung in Rechnung gestellt:

- Der Auftraggeber gibt den Parameter `x-amz-request-payer` nicht im Header (GET, HEAD oder POST) oder als Parameter (REST) in der Anforderung (HTTP-Code 403) an.
- Die Authentifizierung der Anforderung schlägt fehl (HTTP-Code 403).
- die Anforderung ist anonym (HTTP-Code 403).
- die Anforderung ist eine SOAP-Anforderung.

Buckets und Zugriffskontrolle

Jedem Bucket ist eine Zugriffssteuerungsrichtlinie zugeordnet. Diese Richtlinie regelt das Erstellen, Löschen und Auflisten von Objekten in dem Bucket. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3 (p. 329).

Fakturierungs- und Nutzungsberichte für S3-Buckets

Bei Verwendung von Amazon Simple Storage Service (Amazon S3) müssen Sie keine Vorabgebühren bezahlen oder sich zur Speicherung einer bestimmten Menge von Inhalten verpflichten. Wie bei anderen Amazon Web Services (AWS)-Services erfolgt die Zahlung nutzungsabhängig und richtet sich nach Ihrer tatsächlichen Nutzung der Services.

AWS bietet die folgenden Berichte für Amazon S3:

- Fakturierungsberichte – Mehrere Berichte, die eine allgemeine Übersicht über alle Aktivitäten für die AWS-Services bereitstellen, die Sie verwenden, einschließlich Amazon S3. AWS stellt die Gebühren immer dem Eigentümer des S3-Buckets für Amazon S3 in Rechnung, es sei denn, der Bucket wurde als Requester Pays-Bucket erstellt. Weitere Informationen über Requester Pays finden Sie unter Buckets mit Zahlung durch Auftraggeber (p. 82). Weitere Informationen über Fakturierungsberichte finden Sie unter AWS-Fakturierungsberichte für Amazon S3 (p. 87).
- Nutzungsbericht – Eine Zusammenfassung von Aktivitäten für einen bestimmten Service, zusammengefasst nach Stunde, Tag und Monat. Sie können wählen, welche Nutzungsart und welche Operation aufgenommen werden soll. Sie können auch festlegen, wie die Daten aggregiert werden. Weitere Informationen finden Sie unter AWS-Nutzungsbericht für Amazon S3 (p. 89).

Die folgenden Themen enthalten Informationen zur Fakturierung und Nutzungsberichte für Amazon S3.

Themen

AWS-Fakturierungsberichte für Amazon S3

In Ihrer monatlichen AWS-Rechnung sind die Nutzungsinformationen und die entsprechenden Kosten nach AWS-Services und -Funktionen aufgeteilt. Es gibt mehrere AWS-Fakturierungsberichte, den Kostenzuordnungsbericht, den monatlichen Bericht und die detaillierten Rechnungsberichte. Informationen zum Anzeigen Ihrer Fakturierungsberichte finden Sie unter Anzeige Ihrer Rechnung im Benutzerhandbuch für AWS Billing and Cost Management.

Sie können auch einen Nutzungsbericht herunterladen, der Ihnen weitere Informationen über Ihre Amazon S3-Nutzung mitteilt, als die Fakturierungsberichte. Weitere Informationen finden Sie unter AWS-Nutzungsbericht für Amazon S3 (p. 89).

Die folgende Tabelle listet die Gebühren für die Amazon S3-Nutzung auf.

Amazon S3-Nutzungsgebühren

Gebühr	Kommentare
Speicher	Sie zahlen für das Speichern von Objekten in S3-Buckets. Der Tarif, der Ihnen in Rechnung gestellt wird, ist von der Größe Ihres Objekts abhängig sowie davon, wie lange die Objekte während des Monats gespeichert wurden, und der Speicherklasse – STANDARD,

Gebühr	Kommentare
	INTELLIGENT_TIERING, STANDARD_IA (IA für "Infrequent Access" (Seltener Zugriff)), ONEZONE_IA, S3 Glacier, S3 Glacier Deep Archive oder RRS (Reduced Redundancy Storage). Weitere Informationen über Speicherklassen finden Sie unter Amazon S3-Speicherklassen (p. 122).
Überwachen und Automatisieren	Sie zahlen eine monatliche Überwachungs- und Automatisierungsgebühr pro Objekt, das in der INTELLIGENT_TIERING-Speicherklasse gespeichert ist, um Zugriffsmuster zu überwachen und Objekte zwischen den Zugriffsstufen in INTELLIGENT_TIERING zu verschieben.
Anforderungen	Sie zahlen für Anfragen, z. B. GET-Anfragen für Ihre S3-Buckets und -Objekte. Dazu zählen auch Lebenszyklus-Anfragen. Die Tarife für Anfragen hängen davon ab, welche Art von Anfragen Sie vornehmen. Weitere Informationen zu den Anfragepreisen erhalten Sie unter Amazon S3 – Preise.
Abrufe	Sie zahlen für das Abrufen von Objekten, die sich im STANDARD_IA-, ONEZONE_IA-, S3 Glacier- und S3 Glacier Deep Archive-Speicher befinden.
Vorzeitiges Löschen	Wenn Sie ein Objekt im INTELLIGENT_TIERING-, STANDARD_IA-, ONEZONE_IA-, S3 Glacier- oder S3 Glacier Deep Archive-Speicher löschen, bevor die minimale Speicherzusage abgelaufen ist, zahlen Sie eine Gebühr für das vorzeitige Löschen für dieses Objekt.
Speicherverwaltung	Sie zahlen für die Speicher-Management-Funktionen (Amazon S3 Bestand, Analyse und Objekt-Tagging), die für die Buckets in Ihrem Konto aktiviert sind.
Bandbreite	Sie zahlen für alle Bandbreite in und aus Amazon S3, mit Ausnahme der folgenden: • Datenübertragung aus dem Internet • Ausgehende Datenübertragungen zu einer Amazon Elastic Compute Cloud (Amazon EC2)-Instance, wenn sich die Instance in derselben AWS-Region wie der S3-Bucket befindet • Ausgehende Datenübertragungen zu Amazon CloudFront (CloudFront) Sie bezahlen auch eine Gebühr für die Übertragung von Daten mit Amazon S3 Transfer Acceleration.

Detaillierte Informationen zu Amazon S3-Nutzungsgebühren für die Speicherung, die Datenübertragung und die Services finden Sie unter Amazon S3 – Preise und Häufig gestellte Fragen zu Amazon S3.

Weitere Informationen über Codes und Abkürzungen in den Fakturierungs- und Nutzungsberichten für Amazon S3 finden Sie unter Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91).

Weitere Infos

- AWS-Nutzungsbericht für Amazon S3 (p. 89)
- Verwenden von Kostenzuordnungs-Tags für S3-Buckets (p. 100)
- AWS-Fakturierung und Kostenmanagement
- Amazon S3 – Preise
- Häufig gestellte Fragen zu Amazon S3
- S3 Glacier – Preise

AWS-Nutzungsbericht für Amazon S3

Für weitere Informationen zu Ihrer Amazon S3-Speichernutzung laden Sie dynamisch generierte AWS-Nutzungsberichte herunter. Sie können wählen, welche Nutzungsart, welche Operation und welcher Zeitraum aufgenommen werden soll. Sie können auch festlegen, wie die Daten aggregiert werden.

Wenn Sie einen Nutzungsbericht herunterladen, können Sie die Nutzungsdaten nach Stunde, Tag und Monat zusammenfassen. Der Amazon S3-Nutzungsbericht listet Operationen nach Nutzungstyp und AWS-Region auf, z. B. ausgehende Datenübertragungen der Region Asien-Pazifik (Sydney).

Der Amazon S3-Nutzungsbericht enthält die folgenden Informationen:

- Service – `Amazon Simple Storage Service`
- Operation – Die Operation, die für Ihren Bucket oder Ihr Objekt durchgeführt wird. Eine detaillierte Erklärung der Amazon S3-Operationen finden Sie unter Nachverfolgen von Operationen in Ihren Nutzungsberichten (p. 99).
- UsageType – Einer der folgenden Werte:
  - Ein Code zur Identifizierung des Speichertyps
  - Ein Code zur Identifizierung der Anfrage
  - Ein Code zur Identifizierung des Abruftyps
  - Ein Code zur Identifizierung des Typs der Datenübertragung
  - Ein Code, der vorzeitiges Löschen aus INTELLIGENT_TIERING-, STANDARD_IA-, ONEZONE_IA-, S3 Glacier- oder S3 Glacier Deep Archive-Speicher identifiziert
  - `StorageObjectCount`: Die Anzahl der Objekte in einem bestimmten Bucket

  Eine detaillierte Erklärung der Amazon S3-Nutzungstypen finden Sie unter Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91).
- Resource – Der Name des Buckets, der der aufgelisteten Nutzung zugeordnet ist.
- Startzeit – Die Startzeit des Tages, auf den sich die Nutzung bezieht, in UTC (koordinierter Weltzeit).
- Endzeit – Die Endzeit des Tages, auf den sich die Nutzung bezieht, in UTC (koordinierter Weltzeit).
- UsageValue – Einer der folgenden Volumenwerte:
  - Die Anzahl der Anforderungen in dem angegebenen Zeitraum
  - Die übertragene Datenmenge in Bytes
  - Die gespeicherte Datenmenge in Byte-Stunden, das ist die Anzahl der Bytes, die innerhalb einer bestimmten Stunde gespeichert wurden

- Die Menge der Daten im Zusammenhang mit Wiederherstellungen aus dem S3 Glacier Deep Archive-, S3 Glacier-, STANDARD_IA- oder ONEZONE-IA-Speicher in Byte

  Tip

  Detaillierte Informationen über alle Anfragen, die Amazon S3 für Ihre Objekte empfängt, aktivieren Sie die Server-Zugriffsprotokollierung für Ihre Buckets Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).

Sie können einen Nutzungsbericht als XML- oder CSV-Datei herunterladen. Das folgende Beispiel zeigt einen CSV-Nutzungsbericht, der in einer Tabellenkalkulation geöffnet ist.

Service	Operation	UsageType	Resource	StartTime	EndTime	UsageValue
AmazonS3	HeadBucket	USW2-C3DataTransfer-Out-Bytes	admin-created3	6/1/2017 0:00	7/1/2017 0:00	15309
AmazonS3	PutObject	USW2-C3DataTransfer-In-Bytes	admin-created3	6/1/2017 0:00	7/1/2017 0:00	19062
AmazonS3	HeadBucket	USW2-Requests-Tier2	admin-created3	6/1/2017 0:00	7/1/2017 0:00	68
AmazonS3	PutObjectForRepl	USW1-Requests-SIA-Tier1	ca-example-bucket	6/1/2017 0:00	7/1/2017 0:00	178294
AmazonS3	PutObjectForRepl	USW1-USW2-AWS-In-Bytes	ca-example-bucket	6/1/2017 0:00	7/1/2017 0:00	387929083
AmazonS3	GetObjectForRepl	USW2-Requests-NoCharge	admin-created3	6/1/2017 0:00	7/1/2017 0:00	108
AmazonS3	GetObjectForRepl	USW2-USW1-AWS-Out-Bytes	my-test-bucket-bash	6/1/2017 0:00	7/1/2017 0:00	387910021

Weitere Informationen über die Nutzungsbericht finden Sie unter Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91).

Den AWS-Nutzungsbericht herunterladen

Sie können einen Nutzungsbericht als XML- oder CSV-Datei herunterladen.

Den Nutzungsbericht herunterladen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Titelleiste Ihren AWS Identity and Access Management (IAM)-Benutzernamen aus und anschließend My Billing Dashboard (Mein Fakturierungs-Dashboard).
3. Wählen Sie im Navigationsbereich AWS Cost and Usage Reports (Kosten- und Nutzungsberichte) aus.
4. Wählen Sie im Abschnitt Other Reports (Andere Berichte) die Option AWS Usage Report (AWS-Nutzungsbericht) aus.
5. Für Services: wählen Sie Amazon Simple Storage Service aus.
6. Für Download Usage Report wählen Sie die folgenden Einstellungen:
   - Verwendungstypen – Eine detaillierte Erläuterung der Amazon S3-Verwendungstypen finden Sie unter Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91).
   - Operation – Eine detaillierte Erläuterung der Amazon S3-Operationen finden Sie unter Nachverfolgen von Operationen in Ihren Nutzungsberichten (p. 99).
   - Zeitraum – Der Zeitraum, der im Bericht erfasst werden soll.
   - Berichtsgranularität – Gibt an, ob in dem Bericht Zwischensummen nach Stunde, Tag oder Monat angezeigt werden sollen.
7. Um das Format für den Bericht zu wählen, wählen Sie Download für dieses Format und folgen dann den Anweisungen, um den Bericht zu speichern.

Weitere Infos

- Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91)
- AWS-Fakturierungsberichte für Amazon S3 (p. 87)

Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen

Amazon S3-Fakturierungs- und Nutzungsberichte verwenden Codes und Abkürzungen. Für den Nutzungstyp beispielsweise, der in der folgenden Tabelle definiert ist, wird *region* durch eine der folgenden Abkürzungen ersetzt:

- APE1: Asien-Pazifik (Hongkong)
- APN1: Asien-Pazifik (Tokio)
- APN2: Asien-Pazifik (Seoul)
- APS1: Asien-Pazifik (Singapur)
- APS2: Asien-Pazifik (Sydney)
- APS3: Asien-Pazifik (Mumbai)
- CAN1: Kanada (Zentral)
- CPT: Afrika (Kapstadt)
- EUN1: Europa (Stockholm)
- EUC1: Europa (Frankfurt)
- EU: Europa (Irland)
- EUW2: Europa (London)
- EUW3: Europa (Paris)
- MES1: Naher Osten (Bahrain)
- SAE1: Südamerika (São Paulo)
- UGW1: AWS GovCloud (US-West)
- UGE1: AWS GovCloud (USA Ost)
- USE1 (or no prefix): USA Ost (Nord-Virginia)
- USE2: USA Ost (Ohio)
- USW1: USA West (Nordkalifornien)
- USW2: USA West (Oregon)

Weitere Informationen zu den Preisen nach AWS-Region finden Sie unter Amazon S3 – Preise.

Die erste Spalte in der folgenden Tabelle listet die Nutzungstypen auf, die in Ihren Fakturierungs- und Nutzungsberichten vorkommen.

Verwendungstypen

Verwendungstyp	Einheiten	Granularität	Beschreibung
region1-*region2*-AWS-In-Bytes	Byte	Stundensatz	Die Menge der in AWS Region1 aus AWS Region2 übertragenen Daten
region1-*region2*-AWS-Out-Bytes	Byte	Stundensatz	Die Menge der von AWS Region1 in AWS Region2 übertragenen Daten
region-BatchOperations-Jobs	Anzahl	Stundensatz	Anzahl der ausgeführten S3 Stapeloperationen-Aufträge.
region-BatchOperations-Objects	Anzahl	Stundensatz	Die Anzahl der von S3 Stapeloperationen

Verwendungstyp	Einheiten	Granularität	Beschreibung
			ausgeführten Objektoperationen.
region-DataTransfer-In-Bytes	Byte	Stundensatz	Die Menge der aus dem Internet zu Amazon S3 übertragenen Daten
region-DataTransfer-Out-Bytes	Byte	Stundensatz	Die Menge der von Amazon S3 ins Internet übertragenen Daten[1]
region-C3DataTransfer-In-Bytes	Byte	Stundensatz	Die Menge der von Amazon EC2 zu Amazon S3 in derselben AWS-Region übertragenen Daten
region-C3DataTransfer-Out-Bytes	Byte	Stundensatz	Die Menge der von Amazon S3 zu Amazon EC2 in derselben AWS-Region übertragenen Daten
region-S3G-DataTransfer-In-Bytes	Byte	Stundensatz	Die Menge der in Amazon S3 aus Wiederherstellungsobjekten aus dem S3 Glacier- oder S3 Glacier Deep Archive-Speicher übertragenen Daten
region-S3G-DataTransfer-Out-Bytes	Byte	Stundensatz	Die Menge der aus Amazon S3 in Übergangsobjekte zum S3 Glacier- oder S3 Glacier Deep Archive-Speicher übertragenen Daten
region-DataTransfer-Regional-Bytes	Byte	Stundensatz	Die Menge der von Amazon S3 zu AWS-Ressourcen in derselben AWS-Region übertragenen Daten
StorageObjectCount	Anzahl	Täglich	Die Anzahl der Objekte in einem bestimmten Bucket
region-CloudFront-In-Bytes	Byte	Stundensatz	Die Menge der in eine AWS-Region aus einer CloudFront-Verteilung übertragenen Daten
region-CloudFront-Out-Bytes	Byte	Stundensatz	Die Menge der aus einer AWS-Region in eine CloudFront-Verteilung übertragenen Daten

Verwendungstyp	Einheiten	Granularität	Beschreibung
region-EarlyDelete-ByteHrs	Byte-Stunden[2]	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem S3 Glacier-Speicher gelöscht wurden, bevor die 90-tägige Mindestnutzung endete[3]
region-EarlyDelete-GDA	Byte-Stunden[2]	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem S3 Glacier Deep Archive-Speicher gelöscht wurden, bevor die 180-tägige Mindestnutzung endete[3]
region-EarlyDelete-SIA	Byte-Stunden	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem STANDARD_IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete[4]
region-EarlyDelete-ZIA	Byte-Stunden	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem ONEZONE_IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete[4]
region-EarlyDelete-SIA-SmObjects	Byte-Stunden	Stundensatz	Anteilige Speichernutzung für kleine Objekte (kleiner als 128 KB), die aus dem STANDARD_IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete[4]
region-EarlyDelete-ZIA-SmObjects	Byte-Stunden	Stundensatz	Anteilige Speichernutzung für kleine Objekte (kleiner als 128 KB), die aus dem ONEZONE_IA-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete[4]
region-Inventory-ObjectsListed	Objekte	Stundensatz	Die Anzahl der Objekte, die für eine Objektgruppe (Objekte werden nach Bucket oder Präfix gruppiert) in einer Lagerbestandsliste aufgelistet werden

Verwendungstyp	Einheiten	Granularität	Beschreibung
region-Requests-S3 Glacier-Tier1	Anzahl	Stundensatz	Die Anzahl an PUT-, COPY-, POST-, InitiateMultipartUpload-, UploadPart- oder CompleteMultipartUpload-Anforderungen auf S3 Glacier-Objekten
region-Requests-S3 Glacier-Tier2	Anzahl	Stundensatz	Die Anzahl an GET-Anforderungen und allen anderen nicht auf S3 Glacier-Objekten aufgelisteten Anforderungen
region-Requests-SIA-Tier1	Anzahl	Stundensatz	Die Anzahl der PUT-, COPY-, POST- oder LIST-Anfragen für STANDARD_IA-Objekte
region-Requests-ZIA-Tier1	Anzahl	Stundensatz	Die Anzahl der PUT-, COPY-, POST- oder LIST-Anfragen für ONEZONE_IA-Objekte
region-Requests-SIA-Tier2	Anzahl	Stundensatz	Die Anzahl von GET- und allen anderen Nicht-SIA-Tier1-Anfragen für STANDARD_IA-Objekte
region-Requests-ZIA-Tier2	Anzahl	Stundensatz	Die Anzahl von GET- und allen anderen Nicht-ZIA-Tier1-Anfragen für ONEZONE_IA-Objekte
Region-Requests-Tier1	Anzahl	Stundensatz	Die Anzahl der PUT-, COPY-, POST- oder LIST-Anforderungen für STANDARD, RRS und Tags
Region-Requests-Tier2	Anzahl	Stundensatz	Die Anzahl von GET- und allen anderen Nicht-Tier1-Anfragen
Region-Requests-Tier3	Anzahl	Stundensatz	Die Anzahl der Lebenszyklusanforderungen zur S3 Glacier- oder S3 Glacier Deep Archive- und S3 Glacier-Standardwiederherstellung
region-Requests-Tier4	Anzahl	Stundensatz	Die Anzahl der Lebenszyklus-Übergänge in INTELLIGENT_TIERING-, STANDARD_IA- oder ONEZONE_IA-Speicher

Verwendungstyp	Einheiten	Granularität	Beschreibung
Region-Requests-Tier5	Anzahl	Stundensatz	Die Anzahl der Anforderungen zur S3 Glacier-Massenwiederherstellung
region-Requests-GDA-Tier1	Anzahl	Stundensatz	Die Anzahl an PUT-, COPY-, POST-, InitiateMultipartUpload-, UploadPart- oder CompleteMultipartUpload-Anforderungen auf DEEP-Archivobjekten
region-Requests-GDA-Tier2	Anzahl	Stundensatz	Die Anzahl der GET-, HEAD- und LIST-Anforderungen
region-Requests-GDA-Tier3	Anzahl	Stundensatz	Die Anzahl der Anforderungen zur S3 Glacier Deep Archive-Standardwiederherstellung
region-Requests-GDA-Tier5	Anzahl	Stundensatz	Die Anzahl der Anforderungen zur S3 Glacier Deep Archive-Massenwiederherstellung
region-Requests-Tier6	Anzahl	Stundensatz	Die Anzahl der Anforderungen zur S3 Glacier-Expresswiederherstellung
region-Bulk-Retrieval-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit S3 Glacier- oder S3 Glacier Deep Archive-Massenanforderungen abgerufen wurden
region-Requests-INT-Tier1	Anzahl	Stundensatz	Die Anzahl an PUT-, COPY-, POST- oder LIST-Anforderungen auf INTELLIGENT_TIERING-Objekten
region-Requests-INT-Tier2	Anzahl	Stundensatz	Die Anzahl an GET-Anforderungen und allen anderen Nicht-Tier1-Anforderungen für INTELLIGENT_TIERING-Objekte

Verwendungstyp	Einheiten	Granularität	Beschreibung
region-Select-Returned-INT-Bytes	Byte	Stundensatz	Die Anzahl an Bytes der Daten, die mit Select-Anforderungen aus INTELLIGENT_TIERING-Speicher zurückgegeben wurden
region-Select-Scanned-INT-Bytes	Byte	Stundensatz	Die Anzahl an Bytes der Daten, die mit Select-Anforderungen aus INTELLIGENT_TIERING-Speicher gescannt wurden
region-EarlyDelete-INT	Byte-Stunden	Stundensatz	Anteilige Speichernutzung für Objekte, die aus dem INTELLIGENT_TIERING-Speicher gelöscht wurden, bevor die 30-tägige Mindestnutzung endete
region-Monitoring-Automation-INT	Objekte	Stundensatz	Die Anzahl an eindeutigen Objekten, die in der INTELLIGENT_TIERING-Speicherklasse überwacht und automatisch abgestuft werden
region-Expedited-Retrieval-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit S3 Glacier-Expressanforderungen abgerufen wurden
region-Standard-Retrieval-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit S3 Glacier- oder S3 Glacier Deep Archive-Standardanforderungen abgerufen wurden
region-Retrieval-SIA	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die aus dem STANDARD_IA-Speicher abgerufen wurden
region-Retrieval-ZIA	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die aus dem ONEZONE_IA-Speicher abgerufen wurden
region-StorageAnalytics-ObjCount	Objekte	Stundensatz	Die Anzahl der eindeutigen Objekte in jeder Objektgruppe (wobei Objekte nach Bucket oder Präfix gruppiert werden), die durch eine Speicheranalyse überwacht werden

Verwendungstyp	Einheiten	Granularität	Beschreibung
region-Select-Scanned-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit Select-Anforderungen aus STANDARD-Speicher gescannt wurden
region-Select-Scanned-SIA-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit Select-Anforderungen aus STANDARD_IA-Speicher gescannt wurden
region-Select-Scanned-ZIA-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit Select-Anforderungen aus ONEZONE_IA-Speicher gescannt wurden
region-Select-Returned-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit Select-Anforderungen aus STANDARD-Speicher zurückgegeben wurden
region-Select-Returned-SIA-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit Select-Anforderungen aus STANDARD_IA-Speicher zurückgegeben wurden
region-Select-Returned-ZIA-Bytes	Byte	Stundensatz	Die Anzahl der Bytes der Daten, die mit Select-Anforderungen aus ONEZONE_IA-Speicher zurückgegeben wurden
region-TagStorage-TagHrs	Tag-Hours	Täglich	Die Gesamtzahl der Tags für alle Objekte im Bucket, gemeldet nach Stunde
region-TimedStorage-ByteHrs	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im STANDARD-Speicher gespeichert wurden
region-TimedStorage-S3 GlacierByteHrs	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im S3 Glacier-Speicher gespeichert wurden

Verwendungstyp	Einheiten	Granularität	Beschreibung
region-TimedStorage-GlacierStaging	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im S3 Glacier-Speicher gespeichert wurden
region-TimedStorage-GDA-ByteHrs	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im S3 Glacier Deep Archive-Speicher gespeichert wurden
region-TimedStorage-GDA-Staging	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im S3 Glacier Deep Archive-Speicher gespeichert wurden
region-TimedStorage-INT-FA-ByteHrs	Byte-Stunden	Täglich	Die Anzahl an Byte-Stunden, die angibt, wie lange die Daten in der Stufe für häufigen Zugriff im INTELLIGENT_TIERING-Speicher gespeichert wurden
region-TimedStorage-INT-IA-ByteHrs	Byte-Stunden	Täglich	Die Anzahl an Byte-Stunden, die angibt, wie lange die Daten in der Stufe für seltenen Zugriff im INTELLIGENT_TIERING-Speicher gespeichert wurden
region-TimedStorage-RRS-ByteHrsb	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im RRS (Reduced Redundancy Storage)-Speicher gespeichert wurden
region-TimedStorage-SIA-ByteHrs	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im STANDARD_IA-Speicher gespeichert wurden
region-TimedStorage-ZIA-ByteHrs	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, wie lange die Daten im ONEZONE_IA-Speicher gespeichert wurden

Verwendungstyp	Einheiten	Granularität	Beschreibung
region-TimedStorage-SIA-SmObjects	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, die kleine Objekte (kleiner als 128 KB) im STANDARD_IA-Speicher gespeichert wurden
region-TimedStorage-ZIA-SmObjects	Byte-Stunden	Täglich	Die Anzahl der Byte-Stunden, die kleine Objekte (kleiner als 128 KB) im ONEZONE_IA-Speicher gespeichert wurden

Hinweise:

1. Wenn Sie eine Übertragung vor dem Abschluss beenden, kann die Menge der übertragenen Daten die Menge der von Ihrer Anwendung erhaltenen Daten überschreiten. Zu dieser Diskrepanz kann es etwa kommen, weil eine Aufforderung zum Beenden einer Übertragung nicht umgehend ausgeführt werden kann und sich eine gewisse Datenmenge in der Zeit bis zur Ausführung der Anforderung möglicherweise noch in der Übertragung befindet. Diese Daten, die sich in der Übertragung befinden, werden als "ausgehende" Daten abgerechnet.
2. Weitere Informationen über die Byte-Stunden-Einheit finden Sie unter Umwandlung von Byte-Stunden in in Rechnung gestellte GB-Monate (p. 100).
3. Wenn Objekte, die in der Speicherklasse S3 Glacieroder S3 Glacier Deep Archive gespeichert sind, gelöscht, überschrieben oder an eine andere Speicherklasse übertragen werden, bevor die Mindestspeicherdauer abgelaufen ist, d. h. 90 Tage für S3 Glacieroder 180 Tage für S3 Glacier Deep Archive, fällt eine anteilige Gebühr pro Gigabyte für die verbleibenden Tage an.
4. Für kleine Objekte, die in der Speicherklasse INTELLIGENT_TIERING, STANDARD_IA oder ONEZONE_IA archiviert sind, wird bei einer vorzeitigen Löschung, einem Überschreiben oder einem Übergang in eine andere Speicherklasse vor Ablauf von 30 Tagen eine anteilige Gebühr pro GB für die verbleibenden Tage berechnet.
5. Für kleine Objekte (kleiner als 128 KB), die in der Speicherklasse STANDARD_IA oder ONEZONE_IA archiviert sind, wird bei einer vorzeitigen Löschung, einem Überschreiben oder einem Übergang in eine andere Speicherklasse vor 30 Tagen eine anteilige Gebühr pro GB für die verbleibenden Tage berechnet.
6. Für Objekte in der INTELLIGENT_TIERING-Speicherklasse gibt es keine kostenpflichtige Mindestobjektgröße, aber Objekte unter 128 KB sind nicht für die automatische Abstufung geeignet und werden immer mit den Preisen der Stufe für häufigen Zugriff im INTELLIGENT_TIERING-Speicher in Rechnung gestellt.

Nachverfolgen von Operationen in Ihren Nutzungsberichten

Operationen beschreiben die Aktion, die für Ihr AWS-Objekt oder Ihren Bucket durch den angegebenen Nutzungstyp ausgeführt wurde. Operationen werden durch selbsterklärende Codes beschrieben, z. B. `PutObject` oder `ListBucket`. Um zu prüfen, welche Aktionen für Ihren Bucket einen bestimmten Nutzungstyp generiert haben, verwenden Sie diese Codes. Wenn Sie einen Nutzungsbericht erstellen, können Sie wählen, All Operations aufzunehmen oder eine bestimmte Operation, z. B. GetObject, für die ein Bericht erstellt werden soll.

Umwandlung von Byte-Stunden in in Rechnung gestellte GB-Monate

Die Speicherplatz, den wir Ihnen jeden Monat in Rechnung stellen, basiert auf der durchschnittlichen, im Laufe des Monats genutzten Speicherkapazität. Sie zahlen für alle Objektdaten sowie Metadaten, die in den Buckets gespeichert werden, die Sie unter Ihrem AWS-Konto erstellt haben. Weitere Informationen über Metadaten erhalten Sie unter Objektschlüssel und Metadaten (p. 117).

Wir messen Ihre Speichernutzung in Speicherdauer-Byte-Stunden, die am Monatsende addiert werden, um Ihre monatlichen Gebühren zu berechnen. Der Nutzungsbericht meldet Ihre Speichernutzung in Byte-Stunden, die Fakturierungsberichte melden die Speichernutzung in GB-Monate. Um Ihren Nutzungsbericht mit Ihren Fakturierungsberichten in Verbindung zu bringen, müssen Sie Byte-Stunden in GB-Monate umrechnen.

Wenn Sie beispielsweise 100 GB (107.374.182.400 Byte) Amazon S3-STANDARD-Speicherdaten im Bucket für die ersten 15 Tage im März speichern und 100 TB (109.951.162.777.600 Byte) Amazon S3-STANDARD-Speicherdaten für die letzten 16 Tage im März, haben Sie 42.259.901.212.262.400 Byte-Stunden verbraucht.

Berechnen Sie zunächst die gesamte Byte-Stunden-Nutzung:

```
[107,374,182,400 bytes x 15 days x (24 hours/day)]
    + [109,951,162,777,600 bytes x 16 days x (24 hours/day)]
    = 42,259,901,212,262,400 byte-hours
```

Anschließend wandeln Sie die Byte-Stunden in GB-Monate um:

```
42,259,901,212,262,400 byte-hours/1,073,741,824 bytes per GB/24 hours per day
     /31 days in March
     =52,900 GB-Months
```

Weitere Infos

- AWS-Nutzungsbericht für Amazon S3 (p. 89)
- AWS-Fakturierungsberichte für Amazon S3 (p. 87)
- Amazon S3 – Preise
- Häufig gestellte Fragen zu Amazon S3
- S3 Glacier – Preise
- S3 Glacier – Häufig gestellte Fragen

Verwenden von Kostenzuordnungs-Tags für S3-Buckets

Um die Speicherkosten oder andere Kriterien für einzelne Projekte oder Gruppen von Projekten nachzuverfolgen, kennzeichnen Sie Ihre Amazon S3-Buckets unter Verwendung der Kostenzuordnungs-Tags. Ein Kostenzuordnungs-Tag ist ein Schlüssel-Wert-Paar, das Sie definieren und mit einem S3-Bucket verknüpfen. Nachdem Sie Kostenzuordnungs-Tags aktiviert haben, verwendet AWS die Tags, um Ihre Ressourcenkosten in Ihrem Kostenzuordnungsbericht zu gruppieren. Kostenzuordnungs-Tags können nur zur Kennzeichnung von Buckets verwendet werden. Weitere Informationen zu Tags, die zur Kennzeichnung von Objekten verwendet werden, finden Sie unter Markieren von Objekten (p. 130).

Der Kostenzuordnungsbericht listet die AWS-Nutzung für Ihr Konto nach Produktkategorie und AWS Identity and Access Management (IAM)-Benutzer auf. Der Bericht enthält die gleichen Einzelposten wie der detaillierte Fakturierungsbericht (siehe Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91)), und zusätzliche Spalten für Ihre Tag-Schlüssel.

AWS bietet zwei Arten von Kostenzuordnungs-Tags, ein von AWS generiertes Tag und benutzerdefinierten Tags. AWS definiert, erstellt und verwendet das von AWS erstellte `createdBy`-Tag für Sie nach einem Amazon S3 CreateBucket-Ereignis. Sie definieren, erstellen und verwenden benutzerdefinierte Tags für Ihren S3-Bucket.

Sie müssen beide Tag-Typen in der Fakturierungs- und Kostenverwaltungskonsole separat aktivieren, damit sie in Ihren monatlichen Berichten angezeigt werden können. Weitere Informationen über von AWS generierte Tags finden Sie unter Von AWS generierte Kostenzuordnungs-Tags. Weitere Informationen zur Aktivierung von Tags finden Sie unter Verwendung von Kostenzuordnungs-Tags im Benutzerhandbuch für AWS Billing and Cost Management.

Ein benutzerdefiniertes Kostenzuordnungs-Tag umfasst folgende Komponenten:

- Der Tag-Schlüssel. Der Tag-Schlüssel ist der Name des Tags. Im Tag project/Trinity beispielsweise ist project der Schlüssel. Die Tag-Schlüssel ist eine Zeichenfolge, in der zwischen Groß- und Kleinschreibung unterschieden wird und die 1 bis 128 Unicode-Zeichen enthalten kann.

- Der Tag-Wert. Der Tag-Wert ist eine erforderliche Zeichenfolge. Im Tag project/Trinity beispielsweise ist Trinity der Wert. Die Tag-Wert ist eine Zeichenfolge, in der zwischen Groß- und Kleinschreibung unterschieden wird und die 0 bis 256 Unicode-Zeichen enthalten kann.

Weitere Informationen zu den zulässigen Zeichen für benutzerdefinierte Tags und anderen Einschränkungen finden Sie unter Einschränkungen benutzerdefinierter Tags im Benutzerhandbuch für AWS Billing and Cost Management.

Jeder S3-Bucket verfügt über einen Tag-Satz. Ein Tag-Satz enthält alle Tags, die diesem Bucket zugewiesen sind. Ein Tag-Satz kann bis zu 50 Tags enthalten oder leer sein. Schlüssel müssen eindeutig innerhalb eines Tag-Satzes sein, aber die Werte in einem Tag-Satz müssen nicht eindeutig sein. Sie können beispielsweise denselben Wert in Tag-Sätzen mit den Namen project/Trinity und cost-center/Trinity haben.

Wenn Sie in einem Bucket ein Tag mit demselben Schlüssel wie für ein vorhandenes Tag hinzufügen, wird der alte Wert mit dem neuen überschrieben.

AWS ordnet Ihren Tags keine semantische Bedeutung zu. Wir interpretieren Tags streng als Zeichenfolgen.

Um Tags hinzuzufügen, aufzulisten, zu bearbeiten oder zu löschen, können Sie die Amazon S3-Konsole, die AWS Command Line Interface (AWS CLI) oder die Amazon S3-API verwenden.

Weitere Informationen zum Erstellen dieser Tags finden Sie unter dem relevanten Thema:

- Weitere Informationen zum Erstellen von Tags in der Konsole finden Sie unter Wie kann ich die Eigenschaften für einen S3-Bucket anzeigen? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
- Weitere Informationen zum Erstellen von Tags mit der Amazon S3-API finden Sie unter PUT Bucket tagging in der Amazon Simple Storage Service API Reference.
- Weitere Informationen zum Erstellen von Tags mit der AWS-CLI finden Sie unter put-bucket-tagging in der AWS CLI Command Reference.

Weitere Informationen über benutzerdefinierte Tags finden Sie unter Benutzerdefinierte Kostenzuordnungs-Tags im Benutzerhandbuch für AWS Billing and Cost Management.

Weitere Infos

- Verwendung von Kostenzuordnungs-Tags im Benutzerhandbuch für AWS Billing and Cost Management
- Ihre AWS-Fakturierungs- und Nutzungsberichte für Amazon S3 verstehen (p. 91)
- AWS-Fakturierungsberichte für Amazon S3 (p. 87)

Verwalten des Datenzugriffs mit Amazon S3 Access Points

Amazon S3 Access Points vereinfachen das skalierbare Verwalten des Datenzugriffs für freigegebene Datasets in S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie S3-Objektoperationen ausführen können, z. B. `GetObject` und `PutObject`. Jeder Zugriffspunkt verfügt über unterschiedliche Berechtigungen und Netzwerkkontrollen, die S3 für alle Anforderungen anwendet, die über diesen Zugriffspunkt eingehen. Jeder Zugriffspunkt erzwingt eine benutzerdefinierte Zugriffspunktrichtlinie, die in Verbindung mit der Bucket-Richtlinie funktioniert, die dem zugrunde liegenden Bucket zugeordnet ist. Sie können jeden Zugriffspunkt so konfigurieren, dass nur Anforderungen aus einer Virtual Private Cloud (VPC) akzeptiert werden, um Amazon S3-Datenzugriff auf ein privates Netzwerk zu beschränken. Sie können auch benutzerdefinierte Block Public Access-Einstellungen für jeden Zugriffspunkt konfigurieren.

Note

- Sie können Zugriffspunkte nur zum Ausführen von Operationen an Objekten verwenden. Sie können mit Zugriffspunkten keine anderen Amazon S3-Operationen auszuführen, z. B. das Ändern oder Löschen von Buckets. Eine vollständige Liste der S3-Operationen, die Zugriffspunkte unterstützen, finden Sie unter Access Point-Kompatibilität mit S3-Operationen und AWS-Services (p. 112).
- Zugriffspunkte funktionieren mit einigen, aber nicht allen AWS-Services und -Funktionen. Beispielsweise können Sie die regionsübergreifende Replikation nicht so konfigurieren, dass sie über einen Zugriffspunkt ausgeführt wird. Eine vollständige Liste der AWS-Services, die mit S3 Access Points kompatibel sind, finden Sie unter Access Point-Kompatibilität mit S3-Operationen und AWS-Services (p. 112).

In diesem Abschnitt wird erläutert, wie Sie mit Amazon S3 Access Points arbeiten. Hinweise zum Arbeiten mit Buckets finden Sie unter Arbeiten mit Amazon S3-Buckets (p. 57). Weitere Informationen zur Arbeit mit Objekten finden Sie unter Arbeiten mit Amazon S3-Objekten (p. 116).

Themen

Erstellen von Zugriffspunkten

Amazon S3 bietet Funktionen zum Erstellen und Verwalten von Zugriffspunkten. Standardmäßig können Sie für jedes Ihrer AWS-Konten bis zu 1.000 Zugriffspunkte pro Region erstellen. Wenn Sie mehr als 1.000 Zugriffspunkte für ein einzelnes Konto in einer Region benötigen, können Sie eine Erhöhung der Servicekontingente beantragen. Weitere Informationen zu Servicekontingenten und zum Beantragen einer Erhöhung finden Sie unter AWS-Servicekontingente in der AWS General Reference.

Sie können S3 Access Points mit der AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDKs oder Amazon S3 REST-API erstellen. Die folgenden Beispiele veranschaulichen, wie Sie einen Zugriffspunkt mit dem AWS CLI und AWS SDK for Java erstellen.

Informationen zum Erstellen von Zugriffspunkten mithilfe der AWS Management Console finden Sie unter Einführung in Amazon S3 Access Points im Konsolenbenutzerhandbuch für Amazon Simple Storage Service. Weitere Informationen zum Erstellen von Zugriffspunkten mit der REST-API finden Sie unter CreateAccessPoint im Amazon Simple Storage Service API Reference.

Note

Da Sie möglicherweise Ihren Zugriffspunktnamen veröffentlichen möchten, um Benutzern die Verwendung des Zugriffspunkts zu gestatten, sollten Sie vertrauliche Informationen im Namen des Zugriffspunkts vermeiden.

Regeln für die Benennung von Amazon S3 Access Points

Zugriffspunktnamen müssen die folgenden Bedingungen erfüllen:

- Muss innerhalb eines einzelnen Kontos und einer einzelnen Region in AWS eindeutig sein
- Muss die DNS-Benennungsbeschränkungen erfüllen
- Muss mit einer Zahl oder einem Kleinbuchstaben beginnen
- Muss zwischen 3 und 50 Zeichen lang sein.
- Kann nicht mit einem Bindestrich beginnen oder enden
- Darf keine Unterstriche, Großbuchstaben oder Punkte enthalten

Example

Beispiel: Erstellen eines Zugriffspunkts

Im folgenden Beispiel wird ein Zugriffspunkt mit dem Namen `example-ap` für den Bucket `example-bucket` im Konto `123456789012` erstellt. Zum Erstellen des Zugriffspunkts senden Sie eine Anforderung an Amazon S3, die den Namen des Zugriffspunkts, den Namen des Buckets, dem Sie den Zugriffspunkt zuordnen möchten, und die Konto-ID für das AWS-Konto angibt, das Eigentümer des Buckets ist. Informationen zu Benennungsregeln finden Sie unter the section called “Regeln für die Benennung von Amazon S3 Access Points” (p. 104).

AWS CLI

```
aws s3control create-access-point --name example-ap --account-id 123456789012 --bucket
 example-bucket
```

Erstellen von Zugriffspunkten, die auf eine Virtual Private Cloud beschränkt sind

Wenn Sie einen Zugriffspunkt erstellen, können Sie ihn über das Internet zugänglich machen, oder Sie können angeben, dass alle über diesen Zugriffspunkt eingehenden Anforderungen aus einer bestimmten Virtual Private Cloud (VPC) stammen müssen. Ein Zugangspunkt, der über das Internet zugänglich ist, soll einen Netzwerkursprung von `Internet` haben. Er kann von überall im Internet verwendet werden, vorbehaltlich anderer Zugriffsbeschränkungen für den Zugriffspunkt, den zugrunde liegenden Bucket und die zugehörige Ressourcen, z. B. die angeforderten Objekte. Ein Zugriffspunkt, auf den nur von einer angegebenen VPC zugegriffen werden kann, hat den Netzwerkursprung `VPC`. Amazon S3 weist jede Anforderung an den Zugriffspunkt zurück, die nicht von dieser VPC stammt.

Important

Sie können den Netzwerkursprung eines Zugriffspunkts nur angeben, wenn Sie den Zugriffspunkt erstellen. Nachdem Sie den Zugriffspunkt erstellt haben, können Sie seinen Netzwerkursprung nicht mehr ändern.

Um einen Zugriffspunkt auf reinen VPC-Zugriff zu beschränken, fügen Sie den Parameter `VpcConfiguration` in die Anforderung zum Erstellen des Zugriffspunkts ein. Im Parameter `VpcConfiguration` geben Sie die VPC-ID an, der die Verwendung des Zugriffspunkts gestattet werden soll. Amazon S3 lehnt dann alle über diesen Zugriffspunkt eingehenden Anforderungen ab, wenn sie nicht von dieser VPC stammen.

Sie können den Netzwerkursprung eines Zugriffspunkts mithilfe der APIs AWS CLI, AWS SDKs oder REST-APIs abrufen. Wenn für einen Zugriffspunkt eine VPC-Konfiguration angegeben ist, lautet der Netzwerkursprung `VPC`. Andernfalls ist der Netzwerkursprung des Zugriffspunkts `Internet`.

Example

Beispiel: Erstellen eines Zugriffspunkts, der auf VPC-Zugriff beschränkt ist

Im folgenden Beispiel wird ein Zugriffspunkt mit dem Namen `example-vpc-ap` für Bucket `example-bucket` in Konto `123456789012` erstellt, der den Zugriff nur von der VPC `vpc-1a2b3c` aus zulässt. Das Beispiel überprüft dann, ob der neue Zugriffspunkt den Netzwerkursprung `VPC` hat.

AWS CLI

```
aws s3control create-access-point --name example-vpc-ap --account-id 123456789012 --
bucket example-bucket --vpc-configuration VpcId=vpc-1a2b3c
```

```
aws s3control get-access-point --name example-vpc-ap --account-id 123456789012

{
    "Name": "example-vpc-ap",
    "Bucket": "example-bucket",
    "NetworkOrigin": "VPC",
    "VpcConfiguration": {
        "VpcId": "vpc-1a2b3c"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2019-11-27T00:00:00Z"
}
```

Um einen Zugriffspunkt mit einer VPC zu verwenden, müssen Sie die Zugriffsrichtlinie für Ihren VPC-Endpunkt ändern. VPC-Endpunkte ermöglichen den Datenfluss von Ihrer VPC zu Amazon S3. Sie verfügen über Zugriffssteuerungsrichtlinien, die steuern, wie Ressourcen innerhalb der VPC mit S3 interagieren dürfen. Anforderungen von Ihrer VPC an S3 werden nur dann über einen Zugriffspunkt erfolgreich ausgeführt, wenn die VPC-Endpunktrichtlinie sowohl Zugriff auf den Zugriffspunkt als auch auf den zugrunde liegenden Bucket gewährt.

In der folgenden Beispielrichtlinienanweisung wird ein VPC-Endpunkt so konfiguriert, dass Aufrufe von `GetObject` für einen Bucket mit dem Namen `example-bucket` und einen Zugriffspunkt mit dem Namen `example-vpc-ap` zulässig sind.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
    {
        "Principal": "*",
        "Action": [
            "s3:GetObject"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::example-bucket/*",
            "arn:aws:s3:us-west-2:123456789012:accesspoint/example-vpc-ap/object/*"
        ]
    }]
}
```

Note

Die `"Resource"`-Deklaration in diesem Beispiel verwendet einen Amazon-Ressourcennamen (ARN) zur Angabe des Zugriffspunkts. Weitere Informationen zu Zugriffspunkt-ARNs finden Sie unter Verwenden von Zugriffspunkten (p. 111).

Weitere Informationen zu VPC-Endpunktrichtlinien finden Sie unter Verwenden von Endpoint-Richtlinien für Amazon S3 im Amazon Virtual Private Cloud Benutzerhandbuch.

Verwalten des öffentlichen Zugriffs auf Zugriffspunkte

Amazon S3 Access Points unterstützen unabhängige Block Public Access-Einstellungen für Zugriffspunkt. Wenn Sie einen Zugriffspunkt erstellen, können Sie die Block Public Access-Einstellungen für diesen Zugriffspunkt festlegen. Für jede Anforderung, die über einen Zugriffspunkt eingeht, wertet Amazon S3 die Block Public Access-Einstellungen für diesen Zugriffspunkt, den zugrunde liegenden Bucket und das Konto des Bucket-Eigenümers aus. Wenn eine dieser Einstellungen darauf hinweist, dass die Anforderung gesperrt werden soll, lehnt Amazon S3 die Anforderung ab.

Weitere Hinweise zur Funktion S3 Block Public Access finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493).

Important

- Alle Block Public Access-Einstellungen sind standardmäßig für Zugriffspunkte aktiviert. Sie müssen alle Einstellungen, die Sie nicht auf einen Zugriffspunkt anwenden möchten, explizit deaktivieren.
- Amazon S3 unterstützt derzeit nicht das Ändern der Public Block Access-Einstellungen eines Zugriffspunkts, nachdem der Zugriffspunkt erstellt wurde.

Example

Beispiel: Erstellen eines Zugriffspunkts mit benutzerdefinierten Block Public Access-Einstellungen

In diesem Beispiel wird ein Zugriffspunkt mit dem Namen `example-ap` für Bucket `example-bucket` in Konto `123456789012` mit nicht standardmäßigen Block Public Access-Einstellungen erstellt. Das Beispiel ruft dann die Konfiguration des neuen Zugriffspunkts ab, um seine Block Public Access-Einstellungen zu überprüfen.

AWS CLI

```
aws s3control create-access-point --name example-ap --account-id
 123456789012 --bucket example-bucket --public-access-block-configuration
 BlockPublicAcls=false,IgnorePublicAcls=false,BlockPublicPolicy=true,RestrictPublicBuckets=true
```

```
aws s3control get-access-point --name example-ap --account-id 123456789012

{
    "Name": "example-ap",
    "Bucket": "example-bucket",
    "NetworkOrigin": "Internet",
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": false,
        "IgnorePublicAcls": false,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2019-11-27T00:00:00Z"
}
```

Konfigurieren von IAM-Richtlinien für die Verwendung von Zugriffspunkten

Amazon S3 Access Points unterstützt AWS Identity and Access Management (IAM)-Ressourcenrichtlinien, mit denen Sie die Verwendung des Zugriffspunkts nach Ressource, Benutzer oder anderen Bedingungen steuern können. Damit eine Anwendung oder ein Benutzer über einen Zugriffspunkt auf Objekte zugreifen kann, müssen sowohl der Zugriffspunkt als auch der zugrunde liegende Bucket die Anforderung zulassen.

> Important
>
> Das Hinzufügen eines S3 Access Points zu einem Bucket ändert nicht das Verhalten des Buckets, wenn über den vorhandenen Bucket-Namen oder ARN auf ihn zugegriffen wird. Alle vorhandenen Operationen für den Bucket funktionieren weiterhin wie zuvor. Einschränkungen, die Sie in eine Zugriffspunktrichtlinie einschließen, gelten nur für Anforderungen, die über diesen Zugriffspunkt eingehen.

Bedingungsschlüssel

S3 Access Points führen drei neue Bedingungsschlüssel ein, die in IAM-Richtlinien zum Steuern des Zugriffs auf Ihre Ressourcen verwendet werden können:

s3:DataAccessPointArn

Dies ist eine Zeichenfolge, die Sie mit einem Zugriffspunkt-ARN abgleichen können. Das folgende Beispiel gleicht alle Zugriffspunkte für das AWS-Konto `123456789012` in Region `us-west-2` ab:

```
"Condition" : {
    "StringLike": {
        "s3:DataAccessPointArn": "arn:aws:s3:us-west-2:123456789012:accesspoint/*"
    }
}
```

s3:DataAccessPointAccount

Dies ist ein Zeichenfolgenoperator, mit dem Sie die Konto-ID des Eigentümers eines Zugriffspunkts abgleichen können. Das folgende Beispiel entspricht allen Zugangspunkten im Besitz von AWS-Konto `123456789012`.

```
"Condition" : {
    "StringEquals": {
        "s3:DataAccessPointAccount": "123456789012"
    }
```

```
}
```

s3:AccessPointNetworkOrigin

Dies ist ein Zeichenfolgenoperator, den Sie verwenden können, um für den Netzwerkursprung entweder `Internet` oder `VPC` abzugleichen. Im folgenden Beispiel werden nur Zugriffspunkte mit einem VPC-Ursprung abgeglichen.

```
"Condition" : {
    "StringEquals": {
        "s3:AccessPointNetworkOrigin": "VPC"
    }
}
```

Weitere Informationen zur Verwendung von Bedingungsschlüsseln mit Amazon S3 finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Delegieren der Zugangskontrolle an Zugriffspunkte

Sie können die Zugriffssteuerung für einen Bucket an die Zugriffspunkte des Buckets delegieren. Die folgende Beispiel-Bucket-Richtlinie ermöglicht Vollzugriff auf alle Zugangspunkte, die dem Konto des Bucket-Besitzers gehören. Somit wird der gesamte Zugriff auf diesen Bucket durch die an seine Zugriffspunkte angehängten Richtlinien gesteuert. Wir empfehlen, Ihre Buckets auf diese Weise für alle Anwendungsfälle zu konfigurieren, die keinen direkten Zugriff auf den Bucket erfordern.

Example Bucket-Richtlinie zum Delegieren der Zugriffssteuerung an Zugriffspunkte

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : "*",
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Bucket owner's account ID" }
        }
    }]
}
```

Beispiele von -Zugriffspunktrichtlinien

In den folgenden Beispielen wird veranschaulicht, wie IAM-Richtlinien zum Steuern von Anforderungen erstellt werden, die über einen Zugriffspunkt eingehen.

Note

Berechtigungen, die in einer Zugriffspunktrichtlinie erteilt werden, sind nur wirksam, wenn der zugrunde liegende Bucket auch denselben Zugriff zulässt. Sie können dies auf zwei Arten erreichen:

1. (Empfohlen) Delegieren Sie die Zugriffssteuerung vom Bucket an den Zugriffspunkt, wie unter Delegieren der Zugangskontrolle an Zugriffspunkte (p. 108) beschrieben.
2. Fügen Sie der Richtlinie des zugrunde liegenden Buckets dieselben Berechtigungen hinzu, die in der Zugriffspunktrichtlinie enthalten sind. Das erste Beispiel für eine Zugriffspunktrichtlinie veranschaulicht, wie die zugrunde liegende Bucket-Richtlinie abgeändert wird, damit der erforderliche Zugriff gewährt wird.

Example Berechtigungserteilung über Zugangspunkt-Richtlinien

Die folgende Zugriffspunktrichtlinie gewährt IAM-Benutzer `Alice` in Konto `123456789012` Berechtigungen für `GET` - und `PUT`-Objekte mit dem Präfix `Alice/` über Zugriffspunkt `my-access-point` in Konto `123456789012`.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Alice"
        },
        "Action": ["s3:GetObject", "s3:PutObject"],
        "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point/object/
Alice/*"
    }]
}
```

Note

Damit die Zugriffspunkt-Richtlinie effektiv Zugriff auf `Alice` gewährt, muss der zugrunde liegende Bucket `Alice` auch denselben Zugriff gewähren. Sie können die Zugriffssteuerung vom Bucket an den Zugriffspunkt delegieren, wie unter Delegieren der Zugangskontrolle an Zugriffspunkte (p. 108) beschrieben. Oder Sie können dem zugrunde liegenden Bucket die folgende Richtlinie hinzufügen, um Alice die erforderlichen Berechtigungen zu erteilen. Beachten Sie, dass sich der `Resource`-Eintrag zwischen den Zugriffspunkt- und Bucket-Richtlinien unterscheidet.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Alice"
        },
        "Action": ["s3:GetObject", "s3:PutObject"],
        "Resource": "arn:aws:s3:::example-bucket/Alice/*"
    }]
}
```

Example Zugriffspunktrichtlinie mit Tag-Bedingung

Die folgende Zugriffspunktrichtlinie gewährt IAM-Benutzer `Bob` in Konto `123456789012` Berechtigungen für `GET`- -Objekte über Zugriffspunkt `my-access-point` in Konto `123456789012`, für deren Tag-Schlüssel `data` der Wert `finance` festgelegt ist.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect":"Allow",
        "Principal" : {
            "AWS": "arn:aws:iam::123456789012:user/Bob"
        },
        "Action":"s3:GetObject",
        "Resource" : "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point/object/
*",
        "Condition" : {
```

```
            "StringEquals": {
                "s3:ExistingObjectTag/data": "finance"
            }
        }
    }]
}
```

Example Zugriffspunktrichtlinie, die eine Bucket-Auflistung ermöglicht

Die folgende Zugriffspunktrichtlinie berechtigt den IAM-Benutzer `Charles` im Konto `123456789012` dazu, die Objekte in dem Bucket anzuzeigen, der Zugriffspunkt `my-access-point` im Konto `123456789012` zugrunde liegt.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Charles"
        },
        "Action": "s3:ListBucket",
        "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point"
    }]
}
```

Example Service-Kontrollrichtlinie

Die folgende Service-Steuerungsrichtlinie erfordert, dass alle neuen Zugriffspunkte mit einem VPC-Netzwerkursprung erstellt werden. Mit dieser Richtlinie können Benutzer in Ihrer Organisation keine neuen Zugriffspunkte erstellen, auf die über das Internet zugegriffen werden kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Deny",
        "Action": "s3:CreateAccessPoint",
        "Resource": "*",
        "Condition": {
            "StringNotEquals": {
                "s3:AccessPointNetworkOrigin": "VPC"
            }
        }
    }]
}
```

Example Bucket-Richtlinie zur Begrenzung von S3-Operationen für VPC-Netzwerkursprünge

Die folgende Bucket-Richtlinie beschränkt den Zugriff auf alle S3-Objektoperationen für Bucket `examplebucket` auf Zugriffspunkte mit einem VPC-Netzwerkursprung.

> **Important**
>
> Bevor Sie eine Anweisung wie dieses Beispiel verwenden, stellen Sie sicher, dass Sie keine Funktionen verwenden müssen, die von Zugriffspunkten nicht unterstützt werden, z. B. regionsübergreifende Replikation.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:BypassGovernanceRetention",
                "s3:DeleteObject",
                "s3:DeleteObjectTagging",
                "s3:DeleteObjectVersion",
                "s3:DeleteObjectVersionTagging",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:GetObjectTagging",
                "s3:GetObjectVersion",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging",
                "s3:ListMultipartUploadParts",
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectLegalHold",
                "s3:PutObjectRetention",
                "s3:PutObjectTagging",
                "s3:PutObjectVersionAcl",
                "s3:PutObjectVersionTagging",
                "s3:RestoreObject"
            ],
            "Resource": "arn:aws:s3:::examplebucket/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:AccessPointNetworkOrigin": "VPC"
                }
            }
        }
    ]
}
```

Verwenden von Zugriffspunkten

Sie können auf die Objekte in einem Amazon S3-Bucket mit einem Zugriffspunkt über die AWS Management Console, AWS CLI, AWS SDKs oder die S3 REST-APIs zugreifen.

Zugriffspunkte haben Amazon-Ressourcennamen (ARN). Zugriffspunkt-ARNs ähneln Bucket-ARNs, werden jedoch explizit eingegeben und kodieren die Region des Zugriffspunkts und die AWS-Konto-ID des Eigentümers des Zugriffspunkts. Weitere Informationen zur Verwendung von ARNs finden Sie unter Amazon-Ressourcennamen (ARN) im AWS General Reference.

Zugriffspunkt-ARNs verwenden das Format `arn:aws:s3:`*`region`*`:`*`account-id`*`:accesspoint/`*`resource`*. Zum Beispiel:

- arn:aws:s3:us-west- 2:123456789012:accesspoint/test repräsentiert den Zugriffspunkt `test` im Besitz von Konto `123456789012` in Region `us-west-2`.
- arn:aws:s3:us-west- 2:123456789012:accesspoint/* repräsentiert alle Zugriffspunkte unter Konto `123456789012` in Region `us-west-2`.

ARNs für Objekte, auf die über einen Zugriffspunkt zugegriffen wird, haben das Format `arn:aws:s3:`*`region`*`:`*`account-id`*`:accesspoint/`*`access-point-name`*`/object/`*`resource`*. Zum Beispiel:

- arn:aws:s3:us-west- 2:123456789012:accesspoint/test/object/unit-01 repräsentiert das Objekt `unit-01`, auf das über den Zugriffspunkt `test` im Besitz von Konto `123456789012` in Region `us-west-2` zugegriffen wird.
- arn:aws:s3:us-west- 2:123456789012:accesspoint/test/object/* repräsentiert alle Objekte für Zugriffspunkt `test` in Konto `123456789012` in Region `us-west-2`.
- arn:aws:s3:us-west- 2:123456789012:accesspoint/test/object/unit-01/finance/* repräsentiert alle Objekte unter Präfix `unit-01/finance/` für Zugriffspunkt `test` in Konto `123456789012` und Region `us-west-2`.

Access Point-Kompatibilität mit S3-Operationen und AWS-Services

S3 Access Points sind mit einer Teilmenge von S3-Operationen und anderen AWS-Services kompatibel. In den folgenden Abschnitten werden die kompatiblen Services und S3-Operationen aufgeführt.

AWS-Services

Sie können S3 Access Points mit AWS CloudFormation verwenden.

Weitere Informationen zu AWS CloudFormation zu finden Sie unter Was ist AWS CloudFormation? im AWS CloudFormation Benutzerhandbuch.

S3-Operationen

Sie können Zugriffspunkte verwenden, um unter Verwendung der folgenden Teilmenge von Amazon S3-APIs auf einen Bucket zuzugreifen:

- `AbortMultipartUpload`
- `CompleteMultipartUpload`
- `CreateMultipartUpload`
- `DeleteObject`
- `DeleteObjectTagging`
- `GetObject`
- `GetObjectAcl`
- `GetObjectLegalHold`
- `GetObjectRetention`
- `GetObjectTagging`
- `HeadObject`
- `ListMultipartUploads`
- `ListObjectsV2`
- `ListParts`
- `PutObject`
- `PutObjectLegalHold`
- `PutObjectRetention`
- `PutObjectAcl`
- `PutObjectTagging`
- `RestoreObject`
- `UploadPart`

Überwachung und Protokollierung

Amazon S3 protokolliert Anforderungen, die über Zugriffspunkte ausgeführt werden, und Anforderungen an die APIs, die Zugriffspunkte verwalten, z. B. `CreateAccessPoint` und `GetAccessPointPolicy`.

Anforderungen, die über einen Zugriffspunkt an Amazon S3 gesendet werden, werden in den Zugriffsprotokollen und AWS CloudTrail-Protokollen des S3-Servers mit dem Hostnamen des Zugriffspunkts angezeigt. Der Hostname eines Zugriffspunkts hat das Format *`access_point_name`*`-`*`account_id`*`.s3-accesspoint.`*`Region`*`.amazonaws.com`. Angenommen, Sie haben die folgende Bucket- und Zugriffspunktkonfiguration:

- Ein Bucket mit dem Namen `my-bucket` in Region `us-west-2`, die Objekt `my-image.jpg` enthält
- Ein Zugriffspunkt mit dem Namen `my-bucket-ap`, der mit `my-bucket` verknüpft ist
- Ihre AWS-Konto-ID ist `123456789012`.

Eine Anforderung, die gesendet wird, um `my-image.jpg` direkt über den Bucket anzurufen, wird in Ihren Protokollen mit dem Hostnamen `my-bucket.s3.us-west-2.amazonaws.com` angezeigt. Wenn Sie die Anforderung stattdessen über den Zugriffspunkt durchführen, ruft Amazon S3 dasselbe Objekt ab, protokolliert die Anforderung jedoch mit dem Hostnamen `my-bucket-ap-123456789012.s3-accesspoint.us-west-2.amazonaws.com`.

Weitere Informationen zu S3 Server-Zugriffsprotokollen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777). Weitere Informationen zu AWS CloudTrail finden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail User Guide.

> Note
>
> S3 Access Points sind derzeit nicht mit Amazon CloudWatch-Metriken kompatibel.

Beispiele

Example

Beispiel: Anfordern eines Objekts über einen Zugriffspunkt

Das folgende Beispiel veranschaulicht, wie Sie das Objekt `my-image.jpg` über den Zugangspunkt `prod` anfordern, der der Konto-ID `123456789012` in Region `us-west-2` gehört, und speichert die heruntergeladene Datei unter `download.jpg`.

AWS CLI

```
aws s3api get-object --key my-image.jpg --bucket arn:aws:s3:us-
west-2:123456789012:accesspoint/prod download.jpg
```

Example

Beispiel: Hochladen eines Objekts über einen Zugriffspunkt

Im folgenden Beispiel wird veranschaulicht, wie das Objekt `my-image.jpg` über den Zugriffspunkt `prod` im Besitz von Konto-ID `123456789012` in Region `us-west-2` hochgeladen wird.

AWS CLI

```
aws s3api put-object --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod --key
 my-image.jpg --body my-image.jpg
```

Example

Beispiel: Löschen eines Objekts über einen Zugriffspunkt

Im folgenden Beispiel wird veranschaulicht, wie das Objekt `my-image.jpg` über den Zugriffspunkt `prod` im Besitz von Konto-ID `123456789012` in Region `us-west-2` gelöscht wird.

AWS CLI

```
aws s3api delete-object --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod --
key my-image.jpg
```

Example

Beispiel: Auflisten von Objekten über einen Zugriffspunkt

Im folgenden Beispiel wird veranschaulicht, wie Objekte über den Zugriffspunkt `prod` im Besitz von Konto-ID `123456789012` in Region `us-west-2` aufgelistet werden.

AWS CLI

```
aws s3api list-objects-v2 --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod
```

Example

Beispiel: Hinzufügen eines Tag-Satzes zu einem Objekt über einen Zugriffspunkt

Das folgende Beispiel veranschaulicht, wie Sie dem vorhandenen Objekt `my-image.jpg` über den Zugangspunkt `prod` im Besitz von Konto-ID `123456789012` in Region `us-west-2` einen Tag-Satz hinzufügen.

AWS CLI

```
aws s3api put-object-tagging --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/
prod --key my-image.jpg --tagging TagSet=[{Key="finance",Value="true"}]
```

Example

Beispiel: Gewähren von Zugriffsberechtigungen über einen Zugriffspunkt mithilfe einer ACL

Im folgenden Beispiel wird veranschaulicht, wie eine ACL auf das vorhandene Objekt `my-image.jpg` über den Zugangspunkt `prod` im Besitz von Konto-ID `123456789012` in Region `us-west-2` angewandt wird.

AWS CLI

```
aws s3api put-object-acl --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod --
key my-image.jpg --acl private
```

Einschränkungen von Access Points

Es gelten die folgenden Einschränkungen und Begrenzungen für Amazon S3-Zugriffspunkte:

- Sie können nur Zugriffspunkte für Buckets erstellen, die Sie besitzen.

- Jeder Zugriffspunkt ist genau einem Bucket zugeordnet, den Sie beim Erstellen des Zugriffspunkts angeben müssen. Nachdem Sie einen Zugriffspunkt erstellt haben, können Sie ihn keinem anderen Bucket zuordnen. Sie können jedoch einen Zugriffspunkt löschen und dann einen anderen mit demselben Namen erstellen, der einem anderen Bucket zugeordnet ist.
- Nachdem Sie einen Zugriffspunkt erstellt haben, können Sie die VPC-Konfiguration (Virtual Private Cloud) nicht mehr ändern.
- Zugriffspunkt-Richtlinien sind auf eine Größe von 20 KB beschränkt.
- Sie können maximal 1.000 Access Points pro AWS-Konto pro Region erstellen. Wenn Sie mehr als 1.000 Zugriffspunkte für ein einzelnes Konto in einer Region benötigen, können Sie eine Erhöhung der Servicekontingente beantragen. Weitere Informationen zu Servicekontingenten und zum Beantragen einer Erhöhung finden Sie unter AWS-Servicekontingente in der AWS General Reference.
- Sie können einen Zugriffspunkt nicht als Ziel für die bereichsübergreifende Replikation in S3 verwenden. Weitere Informationen zur Replikation finden Sie unter Replikation (p. 669).
- Zugriffspunkte können nur mit URLs im Virtual-Host-Stil adressiert werden. Weitere Informationen zur Adressierung im Virtual-Host-Stil finden Sie unter Zugreifen auf einen Bucket.
- APIs, die die Zugriffspunktfunktionalität steuern (z. B. `PutAccessPoint` und `GetAccessPointPolicy`), unterstützen keine kontenübergreifende Aufrufe.
- Sie müssen AWS Signature Version 4 verwenden, wenn Sie Anforderungen an einen Zugriffspunkt mithilfe der REST-APIs senden. Weitere Informationen zum Authentifizieren von Anforderungen finden Sie unter Authentifizieren von Anforderungen (AWS Signature Version 4) im Amazon Simple Storage Service API Reference.
- Zugriffspunkte unterstützen nur den Zugriff über HTTPS.
- Zugriffspunkte unterstützen keinen anonymen Zugriff.

Arbeiten mit Amazon S3-Objekten

Amazon S3 ist ein einfacher Schlüssel-Wert-Speicher, der darauf ausgelegt ist, so viele Objekte zu speichern, wie Sie wollen. Sie speichern diese Objekte in einem oder mehreren Buckets. Jedes Objekt kann eine Größe von bis zu 5 TB haben. Ein Objekt besteht aus Folgendem:

- Schlüssel – Der Name, den Sie einem Objekt zuweisen. Zum Abrufen des Objekts verwenden Sie den Objektschlüssel.

  Weitere Informationen finden Sie unter Objektschlüssel und -Metadaten (p. 117).
- Versions-ID – In einem Bucket wird ein Objekt durch einen Schlüssel und eine Versions-ID eindeutig identifiziert.

  Die Versions-ID ist eine Zeichenfolge, die Amazon S3 generiert, wenn Sie das Objekt einem Bucket hinzufügen. Weitere Informationen finden Sie unter Objekt-Versioning (p. 127).
- Wert – Der Inhalt, den Sie speichern.

  Ein Objektwert kann eine beliebige Bytefolge sein. Objekte können von 0 bis 5 TB groß sein. Weitere Informationen finden Sie unter Objekte hochladen (p. 193).
- Metadaten – Ein Satz von Name-Wert-Paaren, mit dem Sie Informationen zum Objekt speichern können.

  Sie können Ihren Objekten in Amazon S3 Metadaten zuweisen, auch als benutzerdefinierte Metadaten bezeichnet. Amazon S3 weist diesen Objekten auch System-Metadaten hinzu, die es für die Verwaltung von Objekten verwendet. Weitere Informationen finden Sie unter Objektschlüssel und -Metadaten (p. 117).
- Subressourcen – Amazon S3 verwendet den Subressourcenmechanismus, um objektspezifische zusätzliche Informationen zu speichern.

  Subressourcen sind Objekten untergeordnet, deshalb sind werden immer einer anderen Entity zugeordnet, wie beispielsweise einem Objekt oder einem Bucket. Weitere Informationen finden Sie unter Objekt-Subressourcen (p. 127).
- Zugriffskontrollinformationen – Sie können den Zugriff auf die von Ihnen in Amazon S3 gespeicherten Objekte kontrollieren.

  Amazon S3 unterstützt sowohl eine ressourcenbasierte Zugriffskontrolle, wie beispielsweise eine Zugriffskontrollliste (ACL, Access Control List), Bucket-Richtlinien und benutzerbasierte Zugriffskontrolle. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3 (p. 329).

Weitere Informationen zur Arbeit mit Objekten finden Sie in den folgenden Abschnitten. Ihre Amazon S3-Ressourcen (wie z. B. Buckets und Objekte) sind standardmäßig privat. Sie müssen anderen ausdrücklich eine Berechtigung erteilen, damit sie auf diese Ressourcen zugreifen dürfen. Beispielsweise könnten Sie ein Video oder ein Foto, das Sie in Ihrem Amazon S3-Bucket gespeichert haben, auf Ihrer Website teilen. Das funktioniert nur, wenn Sie das Objekt öffentlich machen oder eine vorsignierte URL auf Ihrer Website verwenden. Weitere Informationen über das Teilen von Objekten finden Sie unter Ein Objekt mit anderen teilen (p. 190).

Themen

Objektschlüssel und Metadaten

Jedes Amazon S3-Objekt hat Daten, einen Schlüssel und Metadaten. Der Objektschlüssel (oder Schlüsselname) identifiziert das Objekt in einem Bucket eindeutig. Objekt-Metadaten bestehen aus Name/Wert-Paaren. Sie können Objekt-Metadaten beim Hochladen festlegen. Nachdem Sie das Objekt hochgeladen haben, können Sie Objekt-Metadaten nicht mehr ändern. Die einzige Methode, wie Sie Objekt-Metadaten ändern können, ist es, eine Kopie des Objekts anzulegen und die Metadaten festzulegen.

Themen

Objektschlüssel

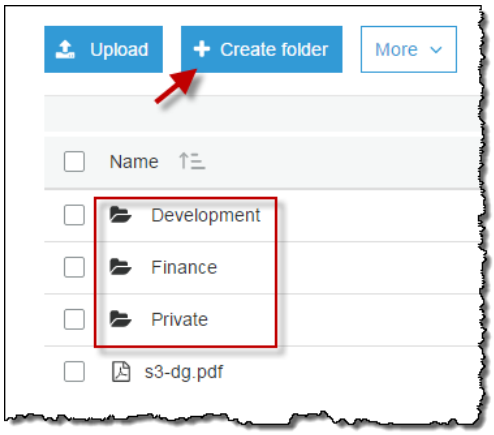
Wenn Sie ein Objekt erstellen, geben Sie den Schlüsselnamen an, der das Objekt in dem Bucket eindeutig identifiziert. Wenn Sie in der Amazon S3-Konsole einen Bucket markieren, erscheint beispielsweise eine Liste der Objekte in Ihrem Bucket. Diese Namen sind die Objektschlüssel. Der Name für einen Schlüssel ist eine Sequenz aus Unicode-Zeichen, deren UTF-8-Codierung maximal 1.024 Byte lang ist.

Das Amazon S3-Datenmodell ist eine Flatfile-Struktur: Sie erstellen einen Bucket und der Bucket speichert Objekte. Es gibt keine Hierarchie von Unterbuckets oder Unterordnern. Sie können jedoch mit den Schlüsselnamenpräfixen und Trennzeichen eine logische Hierarchie erschließen, wie dies die Amazon S3-Konsole tut. Die Amazon S3-Konsole unterstützt ein Ordnerkonzept.

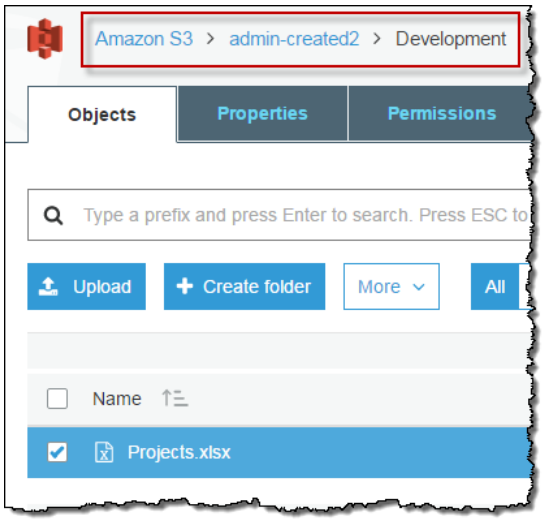
Angenommen, Ihr Bucket (`admin-created`) enthält vier Objekte mit den folgenden Objektschlüsseln:

```
Development/Projects.xls

Finance/statement1.pdf

Private/taxdocument.pdf

s3-dg.pdf
```

Die Konsole verwendet die Schlüsselpräfixe (`Development/`, `Finance/` und `Private/`) und das Trennzeichen ('/'), um eine Ordnerstruktur wie dargestellt anzuzeigen.



Der `s3-dg.pdf`-Schlüssel hat kein Präfix, deshalb erscheint sein Objekt direkt auf Root-Ebene des Buckets. Wenn Sie den Ordner `Development/` öffnen, werden Sie darin das Objekt `Projects.xlsx` sehen.



Note

Amazon S3 unterstützt Buckets und Objekte und es gibt keine Hierarchie. Die Präfixe und Trennzeichen in einem Objektschlüsselnamen gestatten der Amazon S3-Konsole und den AWS SDKs jedoch, eine Hierarchie abzuleiten und das Ordnerkonzept einzuführen.

Richtlinien für Objektschlüsselnamen

Sie können in einem Objektschlüsselnamen jedes beliebige UTF-8-Zeichen verwenden. Die Verwendung bestimmter Zeichen in Schlüsselnamen kann jedoch bei manchen Anwendungen und Protokollen zu Problemen führen. Die folgenden Richtlinien helfen Ihnen, die Konformität mit DNS, web-sicheren Zeichen, XML-Parsern und anderen APIs zu maximieren.

Sichere Zeichen

Die folgenden Zeichensätze sind allgemein sicher für die Verwendung in Schlüsselnamen.

Alphanumerische Zeichen	• 0-9 • a-z • A-Z
Sonderzeichen	• ! • - • _ • . • * • ' • (•)

Nachfolgend finden Sie Beispiele für gültige Objektschlüsselnamen:

- `4my-organization`

- `my.great_photos-2014/jan/myvacation.jpg`
- `videos/2014/birthday/video1.wmv`

> **Important**
>
> Wenn der Schlüsselname eines Objekts mit einem Punkt (.) oder zwei Punkten (..) endet, können Sie das Objekt nicht über die Amazon S3-Konsole herunterladen. Um ein Objekt mit einem Schlüsselnamen herunterzuladen, der mit „.“ oder „..“ endet, müssen Sie die AWS Command Line Interface (AWS CLI), AWS SDKs oder die REST API verwenden.

Zeichen, die möglicherweise eine Sonderverarbeitung benötigen

Die folgenden Zeichen in einem Schlüsselnamen erfordern möglicherweise eine zusätzliche Verarbeitung im Code oder müssen URL-codiert oder als HEX angegeben werden. Einige davon sind nicht darstellbare Zeichen, und Ihr Browser kann sie ggf. nicht verarbeiten, was zudem einer speziellen Vorgehensweise bedarf:

- Ampersand ("&")
- Dollar ("$")
- ASCII-Zeichenbereiche 00–1F hex (0–31 dezimal) und 7F (127 dezimal)
- 'At'-Symbol ("@")
- Gleichheitszeichen ("=")
- Semikolon (";")
- Doppelpunkt (":")
- Plus ("+")
- Leerzeichen – Wichtige Leerzeichenfolgen gehen möglicherweise bei bestimmten Verwendungszwecken verloren (insbesondere Mehrfachleerzeichen).
- Komma (",")
- Fragezeichen ("?")

Zeichen, die Sie vermeiden sollten

Sie sollten in Schlüsselnamen die folgenden Zeichen vermeiden, weil sie einen maßgeblichen Arbeitsaufwand erfordern, um konsistent über alle Anwendungen zu sein.

- Umgekehrter Schrägstrich ("\")
- Linke geschweifte Klammer ("{")
- Nicht darstellbare ASCII-Zeichen (128–255 Dezimalzeichen)
- Caret ("^")
- Rechte geschweifte Klammer ("}")
- Prozentzeichen ("%")
- Accent Grave ("`")
- Rechte eckige Klammer ("]")
- Anführungszeichen
- Größersymbol (">")
- Linke eckige Klammer ("[")
- Tilde ("~")
- Kleiner als-Zeichen („<“)
- Pfundzeichen ("#")
- Vertikaler Strich ("|")

Objekt-Metadaten

Es gibt zweierlei Arten von Metadaten: System-Metadaten und benutzerdefinierte Metadaten.

Systemdefinierte Objektmetadaten

Amazon S3 verwaltet für jedes in einem Bucket gespeicherte Objekt einen Satz Systemmetadaten. Amazon S3 verarbeitet diese Systemmetadaten nach Bedarf. Beispielsweise verwaltet Amazon S3 das Erstellungsdatum eines Objekts und seine Größe als Metadaten und verwendet dies Information für die Objektverwaltung.

Es gibt zwei Kategorien von Systemmetadaten:

1. Metadaten wie beispielsweise das Erstellungsdatum werden vom System gesteuert, und nur Amazon S3 kann den Wert ändern.
2. Andere Systemmetadaten, wie beispielsweise die für das Objekt konfigurierte Speicherklasse, und ob für das Objekt eine serverseitige Verschlüsselung aktiviert ist, sind Beispiele für Metadaten, deren Werte Sie steuern können. Wenn Ihr Bucket als Website konfiguriert ist, wollen Sie eine Seite möglicherweise irgendwann auf eine andere Seite oder zu einer externen URL umleiten. In diesem Fall ist eine Webseite ein Objekt in Ihrem Bucket. Amazon S3 speichert den Wert für die Seitenumleitung als Systemmetadaten, deren Wert Sie steuern können

   Wenn Sie Objekte erstellen, können Sie Werte dieser Systemmetadaten konfigurieren oder sie nach Bedarf aktualisieren. Weitere Informationen über Speicherklassen finden Sie unter Amazon S3-Speicherklassen (p. 122).

   Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Datenschutz durch Verschlüsselung (p. 290).

Die folgenden Tabelle enthält eine Liste der vom System definierten Metadaten, und gibt an, ob Sie sie aktualisieren können.

Name	Beschreibung	Kann der Benutzer den Wert ändern?
Datum	Aktuelles Datum und Uhrzeit.	Nein
Content-Length	Objektgröße in Bytes.	Nein
Content-Type	Objekttyp.	Ja
Letzte Änderung	Datum, zu dem das Objekt erstellt wurde, oder das letzte Änderungsdatum, je nachdem, welcher Wert der neueste ist.	Nein
Inhalt-MD5	Der base64-verschlüsselte 128-Bit-MD5-Digest des Objekts.	Nein
x-amz-server-side-encryption	Gibt an, ob für das Objekt die serverseitige Verschlüsselung aktiviert ist und ob diese Verschlüsselung über die von AWS Key Management Service (AWS KMS) oder von Amazon S3 verwalteten Verschlüsselung (SSE-S3) erfolgt. Weitere Informationen finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290).	Ja
x-amz-version-id	Objektversion. Wenn Sie das Versioning für einen Bucket aktivieren, weist Amazon S3 allen Objekten, die dem Bucket hinzugefügt werden, eine Versions-ID zu.	Nein

Name	Beschreibung	Kann der Benutzer den Wert ändern?
	Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514).	
x-amz-delete-marker	In einem Bucket, für den das Versioning aktiviert ist, gibt dieses Boolesche Kennzeichen an, ob das Objekt eine Löschmarkierung ist.	Nein
x-amz-storage-class	Speicherklasse, für die Speicherung des Objekts verwendet wird. Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122).	Ja
x-amz-website-redirect-location	Leitet Anforderungen für das jeweilige Objekt auf ein anderes Objekt im selben Bucket oder zu einer externen URL um. Weitere Informationen finden Sie unter (Optional) Konfigurieren einer Webseiten-Umleitung (p. 614).	Ja
x-amz-server-side-encryption-aws-kms-key-id	Wenn x-amz-server-side-encryption mit dem Wert `aws:kms` vorhanden ist, wird dadurch die ID des AWS KMS-Kundenmasterschlüssels (CMK) angegeben, der für das Objekt verwendet wurde.	Ja
x-amz-server-side-encryption-customer-algorithm	Gibt an, ob die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) aktiviert ist. Weitere Informationen finden Sie unter Schützen von Daten durch eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) (p. 307).	Ja

Benutzerdefinierte Objektmetadaten

Wenn Sie ein Objekt hochladen, können Sie ihm Metadaten zuweisen. Sie geben diese optionalen Informationen als Name-Wert-Paar (Schlüssel-Wert) an, wenn Sie eine PUT- oder POST-Anforderung senden, um das Objekt zu erstellen. Wenn Sie Objekte mit der REST API hochladen, müssen die optionalen benutzerdefinierten Metadatennamen mit "x-amz-meta-" beginnen, damit sie von den anderen HTTP-Header unterschieden werden können. Wenn Sie das Objekt mit der REST API abrufen, wird dieses Präfix zurückgegeben. Wenn Sie Objekte mit der SOAP API hochladen, ist das Präfix nicht erforderlich. Wenn Sie das Objekt mit der SOAP API abrufen, wird das Präfix entfernt, unabhängig davon, welche API Sie zum Hochladen des Objekts verwendet haben.

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Wenn Metadaten über die REST API abgerufen werden, kombiniert Amazon S3 Header, die denselben Namen haben (wobei die Groß-/Kleinschreibung ignoriert wird) in einer durch Kommas getrennten Liste. Wenn einige Metadaten nicht darstellbare Zeichen enthalten, werden sie nicht zurückgegeben. Stattdessen wird der `x-amz-missing-meta`-Header mit der Anzahl der nicht darstellbaren Metadateneinträge als Wert zurückgegeben.

Benutzerdefinierte Metadaten sind Schlüssel/Wert-Paare. Amazon S3 speichert benutzerdefinierte Metadatenschlüssel in Kleinbuchstaben.

Amazon S3 erlaubt beliebige Unicode-Zeichen in Ihren Metadatenwerten.

Um Probleme bei der Darstellung dieser Metadatenwerte zu vermeiden, sollten Sie US-ASCII-Zeichen verwenden, wenn REST verwendet wird, und UTF-8-Zeichen, wenn SOAP oder browserbasierte Uploads über POST verwendet werden.

Wenn Sie Nicht-US-ASCII-Zeichen in Ihren Metadatenwerten verwenden, wird die bereitgestellte Unicode-Zeichenfolge auf Nicht-US-ASCII-Zeichen untersucht. Wenn die Zeichenfolge nur US-ASCII Zeichen enthält, wird sie so dargestellt, wie sie ist. Wenn die Zeichenfolge Nicht-US-ASCII Zeichen enthält, wird sie zuerst mit UTF-8 zeichencodiert und dann in US-ASCII codiert.

Beispiel:

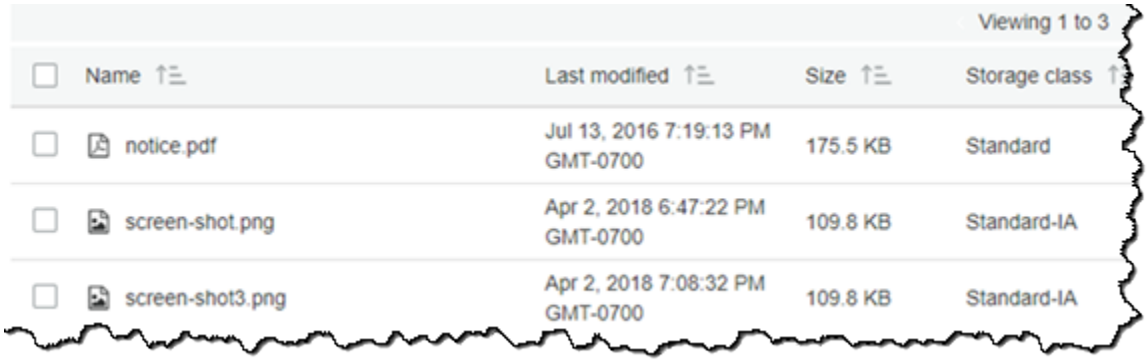
```
PUT /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-nonascii: ÄMÄZÕÑ S3

HEAD /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-nonascii: =?UTF-8?B?w4PChE3Dg8KEWsODwpXDg8KRIFMz?=

PUT /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3

HEAD /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3
```

> **Note**
>
> Der PUT-Anforderungs-Header ist auf eine Größe von 8 KB begrenzt. Innerhalb des PUT-Anforderungsheaders sind die benutzerdefinierten Metadaten auf eine Größe von 2 KB begrenzt. Die Größe der benutzerdefinierten Metadaten wird anhand der Summe der Byteanzahl der UTF-8-Codierung jedes Schlüssels und Werts gemessen.

Informationen dazu, wie Sie einem Objekt Metadaten hinzufügen, nachdem es hochgeladen wurde, finden Sie unter Wie füge ich einem S3-Objekt Metadaten hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Amazon S3-Speicherklassen

Jedem Objekt in Amazon S3 ist eine Speicherklasse zugeordnet. Wenn Sie beispielsweise alle Objekte in einem S3-Bucket auflisten, zeigt die Konsole die Speicherklasse für alle Objekte in der Liste an.



Amazon S3 bietet eine Reihe von Speicherklassen für die von Ihnen gespeicherten Objekte an. Sie wählen eine Speicherklasse abhängig von Ihrem Anwendungsszenario und den Leistungsanforderungen für den Zugriff aus. Alle diese Speicherklassen bieten höchste Haltbarkeit.

Themen

Speicherklassen für Objekte mit häufigem Zugriff

Für leistungssensible Anwendungsfälle (die eine Zugriffszeit von Millisekunden erfordern) und Daten mit häufigem Zugriff stellt Amazon S3 die folgenden Speicherklassen bereit:

- S3 Standard – Die Standard-Speicherklasse. Wenn Sie beim Upload eines Objekts keine Speicherklasse angeben, weist Amazon S3 die S3 Standard-Speicherklasse zu.
- Reduced Redundancy – Die Reduced Redundancy Storage (RRS)-Speicherklasse ist für nicht kritische, reproduzierbare Daten vorgesehen, die mit einer geringeren Redundanz gespeichert werden können als die ––Speicherklasse.

  Important

  Wir empfehlen, diese Speicherklasse nicht zu verwenden. Die S3 Standard-Speicherklasse ist kostengünstiger.

  In Bezug auf die Zuverlässigkeit gilt für RRS-Objekte ein jährlich zu erwartender Objektverlust von 0,01 %. Geht ein RRS-Objekt verloren, gibt Amazon S3 einen 405-Fehler bei Anforderungen für dieses Objekt zurück.

Speicherklasse, die häufig und weniger häufig verwendete Objekte optimiert

Die S3 Intelligent-Tiering-Speicherklasse sorgt für eine Optimierung der Speicherkosten, indem Daten automatisch auf die kostengünstigste Zugriffsebene übertragen werden, ohne dass sich dies auf die Leistungsfähigkeit oder den Betriebsaufwand auswirkt. S3 Intelligent-Tiering ermöglicht automatische Kosteneinsparungen durch die Verschiebung von Daten auf detaillierter Objektebene zwischen zwei Zugriffsebenen, einer Ebene für häufigere Zugriffe und einer kostengünstigeren Ebene für weniger häufige Zugriffe, wenn sich die Zugriffsmuster ändern. Die Intelligent-Tiering-Speicherklasse ist ideal, wenn Sie die Speicherkosten für langlebige Daten automatisch optimieren möchten, wenn die Zugriffsmuster unbekannt oder nicht vorhersehbar sind.

Die S3 Intelligent-Tiering-Speicherklasse speichert Objekte in zwei Zugriffsebenen: Eine Ebene ist für häufige Zugriffe optimiert, während die andere, kostengünstigere, Ebene für Daten mit weniger häufigen Zugriffen optimiert ist. Für eine geringe monatliche Überwachungs- und Automatisierungsgebühr pro Objekt überwacht Amazon S3 die Zugriffsmuster der Objekte in der S3 Intelligent-Tiering-Speicherklasse und verschiebt Objekte, auf die 30 Tage in Folge nicht zugegriffen wurde, in die Ebene für seltenere Zugriffe.

Bei der Verwendung der S3 Intelligent-Tiering-Speicherklasse fallen keine Abrufgebühren an. Wenn auf ein Objekt in der Stufe für seltenere Zugriffe zugegriffen wird, wird es automatisch auf die Stufe für häufige

Zugriffe zurück verschoben. Es werden keine Zusatzgebühren erhoben, wenn Objekte innerhalb der S3 Intelligent-Tiering-Speicherklasse zwischen Zugriffsstufen verschoben werden.

> Note
>
> Die S3 Intelligent-Tiering-Speicherklasse ist für Objekte mit mehr als 128 KB geeignet, die Sie für mindestens 30 Tage aufbewahren möchten. Wenn ein Objekt kleiner als 128 KB ist, ist es nicht für das Auto-Tiering geeignet. Kleinere Objekte können gespeichert werden, für diese fallen aber stets die Gebühren für häufigere Zugriffe in der S3 Intelligent-Tiering-Speicherklasse an.
> Wenn Sie ein Objekt vor dem Ablauf der 30-tägigen Mindestspeicherdauer löschen, werden Ihnen 30 Tage berechnet. Preisinformationen finden Sie unter Amazon S3 – Preise.

Speicherklassen für Objekte mit seltenem Zugriff

Die S3 Standard-IA- und S3 One Zone-IA-Speicherklassen sind für langlebige Daten, auf die selten zugegriffen wird, konzipiert. (IA steht für Infrequent Access, seltener Zugriff.) S3 Standard-IA- und S3 One Zone-IA-Objekte stehen für den Zugriff in Millisekunden zur Verfügung (ebenso wie die Speicherklasse S3 Standard). Da Amazon S3 eine Abrufgebühr für diese Objekte erhebt, sind sie am besten für Daten mit selteneren Zugriffen geeignet. Preisinformationen finden Sie unter Amazon S3-Preise.

Sie können beispielsweise die S3 Standard-IA- und S3 One Zone-IA-Speicherklassen auswählen:

- Zum Speichern von Sicherungen.
- Für ältere Daten, auf die selten zugegriffen wird, für die aber dennoch ein Zugriff in Millisekunden erforderlich ist. Beim Upload von Daten können Sie beispielsweise die Amazon S3-Speicherklasse auswählen und dann S3 Standard mithilfe der Lebenszykluskonfiguration anweisen, die Objekte in die S3 Standard-IA- oder S3 One Zone-IA-Klasse zu überführen.

  Weitere Informationen zur Lebenszyklusverwaltung finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

> Note
>
> Die S3 Standard-IA- und S3 One Zone-IA-Speicherklassen sind für Objekte mit mehr als 128 KB geeignet, die Sie für mindestens 30 Tage aufbewahren möchten. Ist ein Objekt kleiner als 128 KB, fallen Amazon S3-Gebühren für 128 KB an. Wenn Sie ein Objekt vor dem Ablauf der 30-tägigen Mindestspeicherdauer löschen, werden Ihnen 30 Tage berechnet. Preisinformationen finden Sie unter Amazon S3 – Preise.

Diese Speicherklassen weisen folgende Unterschiede auf:

- S3 Standard-IA – Amazon S3 speichert die Objektdaten redundant über mehrere geografisch getrennte Availability Zones (ähnlich der S3 Standard-Speicherklasse). S3 Standard-IA-Objekte sind widerstandsfähig gegen den Verlust einer Availability Zone. Diese Speicherklasse bietet mehr Verfügbarkeit und Stabilität als die S3 One Zone-IA-Klasse.
- S3 One Zone-IA – Amazon S3 speichert die Objektdaten nur in einer Availability Zone, daher ist diese Variante kostengünstiger als S3 Standard-IA. Allerdings sind die Daten bei einem physischen Ausfall der Availability Zone (aufgrund von Katastrophen wie z. B. Erdbeben und Überflutungen) nicht sicher. Die S3 One Zone-IA-Speicherklasse bietet dieselbe Zuverlässigkeit wie Standard_IA, ist jedoch weniger verfügbar und weniger ausfallsicher. Einen Vergleich der Zuverlässigkeit und Verfügbarkeit von Speicherklassen finden Sie in der entsprechenden Tabelle am Ende dieses Abschnitts. Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

Wir empfehlen Folgendes:

- —-IA – Verwenden Sie diese Klasse für die primäre (oder einzige) Kopie der Daten, die nicht neu erstellt werden können.

- S3 One Zone-IA – Verwenden Sie diese Klasse, wenn Sie die Daten im Falle eines Availability Zone-Ausfalls neu erstellen können sowie für Objekt-Replicas bei regionenübergreifender Replikation (CRR) in S3.

Speicherklassen für die Archivierung von Objekten

Die Speicherklassen S3 Glacier und S3 Glacier Deep Archive sind für die kostengünstige Datenarchivierung konzipiert. Diese Speicherklassen bieten dieselbe Zuverlässigkeit und Stabilität wie die S3 Standard-Speicherklasse. Einen Vergleich der Zuverlässigkeit und Verfügbarkeit von Speicherklassen finden Sie in der entsprechenden Tabelle am Ende dieses Abschnitts.

Diese Speicherklassen weisen folgende Unterschiede auf:

- S3 Glacier – für Archive, bei denen Teile der Daten möglicherweise in wenigen Minuten abgerufen werden müssen. Für in der Speicherklasse S3 Glacier gespeicherte Daten gilt eine Mindestspeicherdauer von 90 Tagen; mit beschleunigtem Abruf sind die Daten in nur 1–5 Minuten abrufbar. Falls Sie ein Objekt vor dem 90-tägigen Minimum gelöscht, überschrieben oder an eine andere Speicherklasse übertragen haben, werden Ihnen 90 Tage in Rechnung gestellt. Preisinformationen finden Sie unter Amazon S3 – Preise.
- S3 Glacier Deep Archive—Wird für die Archivierung von Daten verwendet, auf die nur selten zugegriffen werden muss. Für in der Speicherklasse S3 Glacier Deep Archive gespeicherte Daten gilt eine Mindestspeicherdauer von 180 Tagen, und die Standard-Abrufzeit beträgt 12 Stunden. Falls Sie ein Objekt vor dem 180-tägigen Minimum gelöscht, überschrieben oder an eine andere Speicherklasse übertragen haben, werden Ihnen 180 Tage in Rechnung gestellt. Preisinformationen finden Sie unter Amazon S3-Preise.

  S3 Glacier Deep Archive ist die kostengünstigste Speicheroption in AWS. Die Speicherkosten für die S3 Glacier Deep Archive sind kostengünstiger als die Verwendung der Speicherklasse S3 Glacier. Sie können die Abrufkosten für S3 Glacier Deep Archive durch Massenabrufe senken, bei denen Daten innerhalb von 48 Stunden zurückgegeben werden.

Abrufen archivierter Objekte

Sie können die Speicherklasse eines Objekts auf S3 Glacier oder S3 Glacier Deep Archive setzen, genau so wie für andere Speicherklassen, wie im Abschnitt Einrichten der Speicherklasse eines Objekts (p. 126) beschrieben. Die S3 Glacier- und S3 Glacier Deep Archive-Objekte stehen jedoch nicht für einen Zugriff in Echtzeit zur Verfügung. Sie müssen die S3 Glacier- und S3 Glacier Deep Archive-Objekte erst wiederherstellen, bevor Sie darauf zugreifen können (S3 Standard-, RRS-, S3 Standard-IA-, S3 One Zone-IA- und S3 Intelligent-Tiering-Objekte stehen jederzeit zum Zugriff zur Verfügung). Weitere Informationen zum Abruf archivierter Objekte finden Sie unter Wiederherstellen archivierter Objekte (p. 272).

> Important
>
> Wenn Sie die Speicherklasse S3 Glacier Deep Archive oder S3 Glacier auswählen, bleiben die Objekte in Amazon S3. Sie können nicht direkt über den separaten Amazon S3 Glacier-Service darauf zugreifen.

Weitere Informationen zum Amazon S3 Glacier-Service finden Sie im Amazon S3 Glacier-Entwicklerhandbuch.

Vergleich der Amazon S3-Speicherklassen

Die folgende Tabelle vergleicht die verschiedenen Speicherklassen.

Storage Class	Designed for	Durability (designed for)	Availability (designed for)	Availability Zones	Min storage duration	Min billable object size	Other Considerations
STANDARD	Frequently accessed data	99.999999999%	99.99%	>= 3	None	None	None
STANDARD_IA	Long-lived, infrequently accessed data	99.999999999%	99.9%	>= 3	30 days	128 KB	Per GB retrieval fees apply.
INTELLIGENT_TIERING	Long-lived data with changing or unknown access patterns	99.999999999%	99.9%	>= 3	30 days	None	Monitoring and automation fees per object apply. No retrieval fees.
ONEZONE_IA	Long-lived, infrequently accessed, non-critical data	99.999999999%	99.5%	1	30 days	128 KB	Per GB retrieval fees apply. Not resilient to the loss of the Availability Zone.
GLACIER	Long-term data archiving with retrieval times ranging from minutes to hours	99.999999999%	99.99% (after you restore objects)	>= 3	90 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
DEEP_ARCHIVE	Archiving rarely accessed data with a default retrieval time of 12 hours	99.999999999%	99.99% (after you restore objects)	>= 3	180 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
RRS (Not recommended)	Frequently accessed, non-critical data	99.99%	99.99%	>= 3	None	None	None

Alle Speicherklassen – außer S3 One Zone-IA – sind bei einem simultanen vollständigen Datenverlust in einer einzigen Availability Zone und teilweisem Datenverlust in einer weiteren Availability Zone ausfallsicher.

Neben den Leistungsanforderungen Ihres Anwendungsszenarios sollten Sie auch die Preise berücksichtigen. Preisinformationen für Speicherklassen finden Sie unter Amazon S3-Preise.

Einrichten der Speicherklasse eines Objekts

Die Amazon S3-APIs unterstützen das Einrichten (oder Aktualisieren) von Speicherklassen für Objekte wie folgt:

- Beim Erstellen eines neuen Objekts können Sie dessen Speicherklasse angeben. Wenn Sie beispielsweise neue Objekte mit den APIs PUT Object, POST Object und Initiate Multipart Upload erstellen, fügen Sie zum Spezifizieren einer Speicherklasse den Anforderungs-Header `x-amz-storage-class` hinzu. Falls Sie diesen Header weglassen, verwendet Amazon S3 die Standardspeicherklasse Standard.
- Sie können auch die Speicherklasse eines bereits in Amazon S3 gespeicherten Objekts zu einer beliebigen anderen Speicherklasse ändern, indem Sie mit der API PUT Object - Copy eine Kopie des Objekts erstellen. Sie können jedoch nicht PUT Object - Copy verwenden, um in der Speicherklasse S3 Glacier oder S3 Glacier Deep Archive gespeicherte Objekte zu kopieren.

  Sie kopieren das Objekt unter Verwendung desselben Schlüsselnamens in denselben Bucket und geben die Anforderungs-Header wie folgt an:

  - Setzen Sie den Header `x-amz-metadata-directive` auf COPY.
  - Legen Sie `x-amz-storage-class` auf die zu verwendende Speicherklasse fest.

  In einem Bucket mit aktiviertem Versioning kann die Speicherklasse einer spezifischen Objektversion nicht geändert werden. Beim Kopieren gibt ihm Amazon S3 eine neue Versions-ID.
- Sie können Amazon S3 anweisen, die Speicherklasse von Objekten zu ändern, indem Sie einem Bucket eine S3-Lebenszyklus-Konfiguration hinzufügen. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).
- Bei der Einrichtung der Replikation können Sie eine beliebige andere Speicherklasse für die replizierten Objekte einrichten. Sie können jedoch keine Objekte replizieren, die in der Speicherklasse S3 Glacier oder S3 Glacier Deep Archive gespeichert sind. Weitere Informationen finden Sie unter Übersicht über die Replikationskonfiguration (p. 675).

Um Objektspeicherklassen zu erstellen und zu aktualisieren, können Sie die Amazon S3-Konsole, AWS-SDKs oder die AWS Command Line Interface (AWS CLI) verwenden. Die Anforderungen an Amazon S3 werden jeweils mithilfe der Amazon S3-APIs gesendet.

Einschränken von Zugriffsrichtlinienberechtigungen auf eine bestimmte Speicherklasse

Wenn Sie Zugriffsrichtlinienberechtigungen für Amazon S3-Operationen erteilen, können Sie mit dem Bedingungsschlüssel `s3:x-amz-storage-class` einschränken, welche Speicherklasse beim Speichern hochgeladener Objekte verwendet werden soll. Wenn Sie beispielsweise die Berechtigung `s3:PUTObject` erteilen, können Sie das Hochladen von Objekten auf eine bestimmte Speicherklasse einschränken. Eine Beispielrichtlinie finden Sie unter Beispiel 5: Objekt-Uploads auf Objekte mit einer bestimmten Speicherklasse beschränken (p. 389). Weitere Informationen zur Verwendung von Bedingungen in Richtlinien und die vollständige Liste der Amazon S3-Bedingungsschlüssel finden Sie hier:

- Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394)
- Amazon S3-Bedingungsschlüssel (p. 382)

Objekt-Subressourcen

Amazon S3 definiert eine Menge von Subressourcen, die Buckets und Objekten zugeordnet sind. Subressourcen sind Objekten untergeordnet. Dies bedeutet, dass Subressourcen nicht alleine existieren. Sie sind immer einer anderen Entität zugeordnet, z. B. einem Objekt oder einem Bucket.

Die folgende Tabelle listet die Subressourcen auf, die Amazon S3-Objekten zugeordnet sind.

Subressource	Beschreibung
acl	Enthält eine Liste der Rechte, wobei die Empfänger und die erteilten Berechtigungen identifizieren werden. Wenn Sie ein Objekt erstellen, identifiziert die `acl` den Objekteigentümer, der volle Kontrolle über das Objekt hat. Sie können eine Objekt-ACL abrufen oder sie durch eine aktualisierte Liste erteilter Rechte ersetzen. Bei einer Aktualisierung einer ACL müssen Sie die vorhandene ACL ersetzen. Weitere Informationen über ACLs finden Sie in Zugriffsverwaltung mit ACLs (p. 479).
torrent	Amazon S3 unterstützt das BitTorrent-Protokoll. Amazon S3 verwendet die Subressource `torrent` für die Rückgabe der mit dem jeweiligen Objekt verbundenen Torrent-Datei. Zum Abruf einer Torrent-Datei geben Sie die Subressource `torrent` in Ihrer GET-Abfrage an. Amazon S3 erstellt eine Torrent-Datei und gibt sie aus. Sie können nur die `torrent`-Subressource abrufen. Sie können die `torrent`-Subressource nicht erstellen, aktualisieren oder löschen. Weitere Informationen finden Sie unter BitTorrent für Amazon S3 verwenden (p. 766). **Note** Amazon S3 unterstützt das BitTorrent-Protokoll nicht in AWS-Regionen, die nach dem 30. Mai 2016 gestartet wurden.

Objekt-Versioning

Amazon S3-Versioning ermöglicht Ihnen, mehrere Versionen eines Objekts in einem Bucket aufzubewahren. Beispielsweise könnten Sie `my-image.jpg` (Version 111111) und `my-image.jpg`

(Version 222222) in einem einzigen Bucket speichern. S3-Versioning schützt Sie vor den Folgen unbeabsichtigter Überschreibungen und Löschungen. Zudem können es auch zum Archivieren von Objekten verwenden, sodass Sie Zugriff auf Vorgängerversionen erhalten.

> **Note**
>
> Die SOAP-API unterstützt nicht S3-Versioning. Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Neue Amazon S3-Funktionen werden unter SOAP nicht unterstützt.

Zum Anpassen Ihrer Datenaufbewahrungsmethode und zur Speicherkostenkontrolle verwenden Sie das Objekt-Versioning mit Verwaltung des Objektlebenszyklus (p. 139). Informationen zum Erstellen von S3-Lebenszyklus-Richtlinien unter Verwendung der AWS Management Console finden Sie unter Eine Lebenszyklusrichtlinie für einen S3-Bucket erstellen im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Wenn in dem nicht-versionierten Bucket eine Lebenszyklus-Richtlinie für den Ablauf von Objekten vorhanden ist und Sie dasselbe Verhalten hinsichtlich einer dauerhaften Löschung beim Aktivieren des Versionings beibehalten möchten, müssen Sie eine langfristige Ablaufrichtlinie hinzufügen. Die Lebenszyklus-Richtlinie für den langfristigen Ablauf verwaltet das Löschen der langfristigen Objektversionen im versionsfähigen Bucket. (Ein versionsfähiges Bucket behält eine kurzfristige und null oder mehr langfristige Objektversionen.)

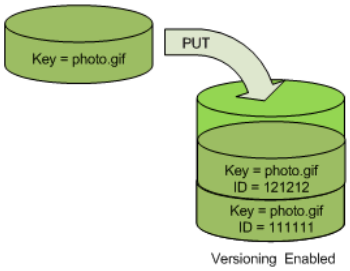
Sie müssen S3-Versioning für Ihren Bucket explizit aktivieren. Standardmäßig ist S3-Versioning deaktiviert. Unabhängig davon, ob für den Bucket das Versioning aktiviert ist, hat jedes Objekt in Ihrem Bucket eine Versions-ID. Falls Sie kein Versioning aktiviert haben, setzt Amazon S3 den Wert der Versions-ID auf null. Wenn S3-Versioning aktiviert ist, weist Amazon S3 dem Objekt einen Versions-ID-Wert zu. Dieser Wert unterscheidet ihn von anderen Versionen desselben Schlüssels.

Die Aktivierung und das Aussetzen des Versionings erfolgt auf Bucket-Ebene. Wenn Sie die Versionierung für einen vorhandenen Bucket aktivieren, bleiben Objekte, die bereits im Bucket gespeichert sind, unverändert. Die Version-IDs (null), Inhalte und Berechtigungen bleiben unverändert. Nachdem Sie S3-Versioning für einen Bucket aktiviert haben, erhält jedes Objekt, das dem Bucket hinzugefügt wird, eine Versions-ID, die sie von anderen Versionen desselben Schlüssels unterscheidet.

Nur Amazon S3 generiert Versions-IDs, und diese können nicht bearbeitet werden. Versions-IDs sind Unicode-, UTF-8-codierte, URL-fähige, nicht einsichtige Zeichenfolgen, die nicht mehr als 1.024 Byte lang sind. Im Folgenden wird ein Beispiel gezeigt: `3/L4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY+MTRCxf3vjVBH40Nr8X8gdRQBpUMLUo`.

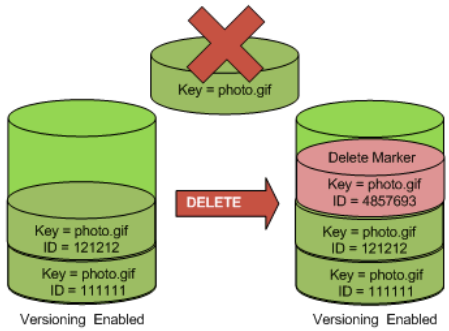
> **Note**
>
> Der Einfachheit halber verwenden alle Beispiele viel kürzere IDs.

Wenn Sie mit `PUT` ein Objekt in einen Bucket mit aktiviertem Versioning schreiben, wird die nicht aktuelle Version nicht überschrieben. Die folgende Abbildung zeigt, dass dann, wenn eine neue Version von `photo.gif` mit `PUT` in einen Bucket geschrieben wird, der bereits ein Objekt desselben Namens enthält, das ursprüngliche Objekt (ID = 111111) im Bucket bleibt und Amazon S3 eine neue Versions-ID erzeugt (121212) und die neuere Version in den Bucket einfügt.

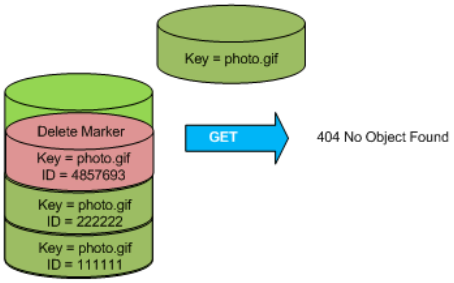


Diese Funktion verhindert, dass Objekte versehentlich gelöscht oder überschrieben werden, und bietet Ihnen die Möglichkeit, eine frühere Version eines Objekts abzurufen.

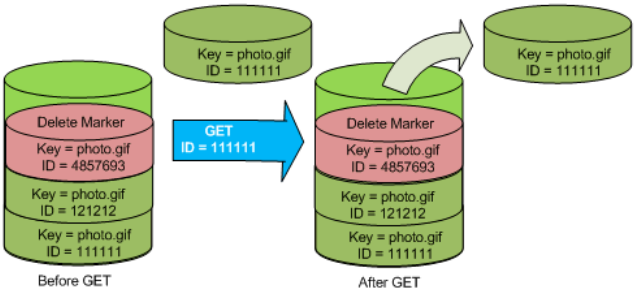
Wenn Sie `DELETE` für ein Objekt ausführen, bleiben alle Versionen in dem Bucket, und Amazon S3 fügt eine Löschmarkierung ein, wie in der folgenden Abbildung gezeigt.



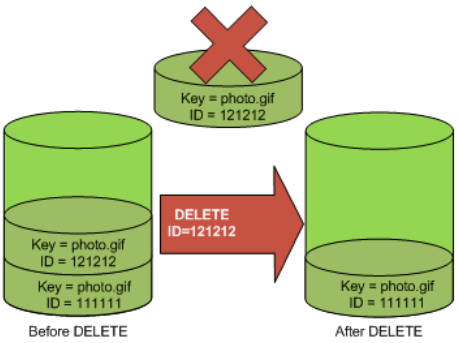
Die Löschmarkierung wird zur aktuellen Version des Objekts. Standardmäßig wird durch `GET`-Anforderungen die zuletzt gespeicherte Version abgerufen. Eine einfache `GET Object`-Anfrage gibt einen `404 Not Found`-Fehler zurück, wenn die aktuelle Version eine Löschmarkierung ist, wie in der folgenden Abbildung gezeigt.



Sie können jedoch mit `GET` eine nicht aktuelle Version eines Objekts abrufen, indem Sie ihre Versions-ID angeben. In der folgenden Abbildung wird `GET` für eine spezifische Objektversion, 111111, durchgeführt. Amazon S3 gibt diese Objektversion zurück, obwohl es sich nicht um die aktuelle Version handelt.



Sie können ein Objekt permanent löschen, indem Sie die Version angeben, die Sie löschen wollen. Nur der Eigentümer eines Amazon S3-Buckets kann eine Version dauerhaft löschen. Die folgende Abbildung zeigt, wie `DELETE versionId` ein Objekt dauerhaft aus einem Bucket löscht, und dass Amazon S3 keine Löschmarkierung einfügt.



Sie können mehr Sicherheit erreichen, indem Sie einen Bucket mit aktiviertem MFA (Multi-Factor Authentication) Delete konfigurieren. In diesem Fall muss der Bucket-Eigentümer zwei Authentifizierungsformen in jede Anforderung aufnehmen, um eine Version zu löschen oder den Versioning-Status des Buckets zu ändern. Weitere Informationen finden Sie unter MFA Delete (p. 516).

> **Important**
>
> Wenn Sie eine deutliche Zunahme von HTTP 503-Verlangsamungsantworten feststellen, die für Amazon S3 PUT- oder DELETE-Objektanforderungen an einen Bucket mit aktiviertem S3-Versioning eingehen, befinden sich möglicherweise ein oder mehrere Objekte im Bucket, für die Millionen von Versionen vorhanden sind. Weitere Informationen finden Sie unter Amazon S3-Fehlerbehebung (p. 773).

Weitere Informationen finden Sie unter Verwendung von Versioning (p. 514).

Markieren von Objekten

Markieren Sie Objekte, um Speicher zu kategorisieren. Jeder Tag ist ein Schlüssel/Wert-Paar. Betrachten Sie die folgenden Beispiele für die Markierung:

- Angenommen, ein Objekt enthält PHI-Daten (Protected Health Information, geschützte Gesundheitsdaten). Sie könnten das Objekt unter Verwendung des folgenden-Schlüssel-Wert-Paares markieren:

```
PHI=True
```

  oder

```
Classification=PHI
```

- Angenommen, Sie speichern Projektdateien in Ihrem S3-Bucket. Sie könnten diese Objekte mit einem Schlüssel namens `Project` und einem Wert markieren, wie nachfolgend gezeigt:

```
Project=Blue
```

- Sie können einem Objekt mehrere Tags hinzufügen, wie nachfolgend gezeigt:

```
Project=x
Classification=confidential
```

Sie können neuen Objekten Tags hinzufügen, wenn Sie sie hochladen, Sie können Tags aber auch vorhandenen Objekten hinzufügen. Beachten Sie Folgendes:

- Sie können einem Objekt bis zu 10 Tags zuordnen. Einem Objekt zugeordnete Tags müssen eindeutige Tag-Schlüssel haben.
- Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen lang sein, und die Tag-Werte können bis zu 256 Unicode-Zeichen lang sein.
- Bei Schlüsseln und Werten wird die Groß-/Kleinschreibung berücksichtigt.
- Weitere Informationen zu Tag-Einschränkungen finden Sie unter Einschränkungen benutzerdefinierter Tags.

Mit Schlüsselnamenpräfixe können Sie auch Speicher kategorisieren. Allerdings sind Präfix-basierte Kategorisierungen eindimensional. Sehen Sie sich die folgenden Objektschlüsselnamen an:

```
photos/photo1.jpg
project/projectx/document.pdf
project/projecty/document2.pdf
```

Dieses Schlüsselnamen haben die Präfixe `photos/`, `project/projectx/` und `project/projecty/`. Diese Präfixe unterstützen eine eindimensionale Kategorisierung. Das bedeutet, alles unter einem Präfix ist eine Kategorie. Beispielsweise identifiziert das Präfix `project/projectx` alle Dokumente, die zu Projekt X gehören.

Mit der Markierung erhalten Sie jetzt eine weitere Dimension. Wenn Sie photo1 in der Kategorie project x anlegen wollen, können Sie das Objekt entsprechend markieren. Neben der Datenklassifizierung bietet die Markierung auch noch weitere Vorteile.

- Objekt-Tags bieten eine differenzierte Zugriffskontrolle für Berechtigungen. Sie könnten z. B. einem IAM-Benutzer Berechtigungen erteilen, nur Objekte mit bestimmten Tags zu lesen.
- Objekt-Tags unterstützen ein differenziertes Objektlebenszyklusmanagement, bei dem Sie in einer Lebenszyklusregel zusätzlich zum Schlüsselnamenpräfix einen auf Tags basierenden Filter angeben können.
- Wenn Sie Amazon S3-Analysen verwenden, können Sie Filter konfigurieren, um Objekte für die Analyse nach Objekt-Tags, nach Schlüsselnamen-Präfix oder nach Präfix und Tags zu gruppieren.
- Sie können außerdem Amazon CloudWatch-Metriken anpassen, um Informationen nach spezifischen Tag-Filtern anzuzeigen. Die folgenden Abschnitte stellen Details bereit.

> **Important**
>
> Es ist akzeptabel, Tags zu verwenden, um Objekte mit vertraulichen Daten zu markieren (z. B. personenbezogene Informationen (PII) oder geschützte Gesundheitsinformationen (PHI)). Diese Tags sollten jedoch selbst keine vertraulichen Daten enthalten.

Zum Hinzufügen von Objekt-Tag-Sets zu mehr als einem Amazon S3-Objekt mit einer einzelnen Anforderung können Sie S3 Stapeloperationen verwenden. Sie stellen S3 Stapeloperationen eine Liste der Objekte zur Verfügung, für die Operationen aufgeführt werden sollen. S3 Stapeloperationen rufen die entsprechende API auf, um die angegebene Operation auszuführen. Eine einzelne S3 Stapeloperationen-Aufgabe kann die festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen.

S3 Stapeloperationen-Stapeloperationen verfolgen den Fortschritt nach, senden Benachrichtigungen und speichern einen detaillierten Abschlussbericht zu allen Aktionen. So erhalten Sie eine vollständig verwaltete, prüfbare und serverlose Umgebung. Sie können S3 Stapeloperationen über die AWS Management Console, die AWS CLI, die AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter the section called “Die Grundlagen: Aufträge” (p. 553).

API-Operationen für die Objektmarkierung

Amazon S3 unterstützt die folgenden API-Operationen, die spezifisch für die Objektmarkierung sind:

Objekt-API-Operationen

- PUT Object tagging – Ersetzt Tags auf einem Objekt. Sie geben die Tags im Anfragerumpf an. Es gibt zwei unterschiedliche Szenarien der Objekt-Tag-Verwaltung unter Verwendung dieser API.
  - Objekt hat keine Tags – Mit Hilfe dieser API können Sie einem Objekt verschiedene Tags hinzufügen (das Objekt hat keine vorherigen Tags).
  - Das Objekt hat einen Satz vorhandener Tags – Um die vorhandene Tag-Menge zu ändern, müssen Sie zuerst den vorhandenen Tag-Satz abrufen, diesen auf der Client-Seite ändern und diese API dann verwenden, um den Tag-Satz zu ersetzen.

    Note

    Wenn Sie diese Anfrage mit einer leeren Tag-Menge senden, löscht Amazon S3 die vorhandene Tag-Menge für das Objekt. Wenn Sie diese Methode verwenden, wird eine Tier 1-Anforderung (PUT) berechnet. Weitere Informationen finden Sie unter Amazon S3-Preise. Die Anforderung DELETE Object tagging wird empfohlen, weil sie das gleiche Ergebnis liefert, aber keine Kosten verursacht.
- GET Object tagging – Gibt den mit einem Objekt verbundenen Tag-Satz zurück. Amazon S3 gibt Objekt-Tags im Antworttext zurück.
- DELETE Object tagging – Löscht den mit einem Tag verbundenen Tag-Satz.

Andere API-Operationen, die Markieren unterstützen

- PUT Object und Initiate Multipart Upload– Sie können beim Erstellen von Objekten Tags angeben. Sie geben Tags unter Verwendung des `x-amz-tagging` Anfrage-Headers an.
- GET Object – Anstatt den Tag-Satz zurückzugeben, gibt Amazon S3 die Anzahl der Objekt-Tags `x-amz-tag-count`-Header zurück (nur dann, wenn der Auftraggeber die Berechtigung hat, Tags zu lesen), weil die Header-Antwortgröße auf 8 KB begrenzt ist. Wenn Sie die Tags anzeigen möchten, erstellen Sie eine weitere Anfrage für die API-Operation GET Object tagging.
- POST Object – Sie können Tags in Ihrer POST-Anfrage angeben.

  So lange die von Ihnen geforderten Tags die 8-KB-Größenbeschränkung der HTTP-Anfrageheader nicht überschreitet, können Sie die `PUT Object` -API verwenden, um Objekte mit Tags zu erstellen. Wenn die von Ihnen angegebenen Tags die Größenbeschränkung des Headers überschreiten, können Sie diese POST-Methode verwenden, wobei Sie die Tags in den Rumpf aufnehmen.

  PUT Object - Copy – Sie können die `x-amz-tagging-directive` in Ihre Anfrage aufnehmen, um Amazon S3 anzuweisen, die Tags zu kopieren (Standardverhalten) oder durch eine neue, in der Anfrage angegebene Tag-Menge zu ersetzen.

Beachten Sie Folgendes:

- Die Markierung folgt dem "Eventual Consistency"-Modell. Das bedeutet, wenn Sie unmittelbar nach dem Hinzufügen von Tags zu einem Objekt versuchen, die Tags abzurufen, erhalten Sie womöglich alte Tags, falls solche für die Objekte vorhanden waren. Ein nachfolgender Aufruf stellt jedoch wahrscheinlich die aktualisierten Tags bereit.

Objektmarkierung und weitere Informationen

Dieser Abschnitt erklärt, was die Objektmarkierung für andere Konfigurationen bedeutet.

Objektmarkierung und Lebenszyklusverwaltung

In der Bucket-Lebenszykluskonfiguration können Sie einen Filter angeben, um eine Untermenge von Objekten auszuwählen, auf die die Regel anzuwenden ist. Sie können einen Filter basierend auf den Schlüsselnamenpräfixen, Objekt-Tags oder beidem angeben.

Angenommen, Sie speichern Fotos (roh und im fertigen Format) in Ihrem Amazon S3-Bucket. Sie könnten diese Objekte wie folgt markieren:

```
phototype=raw
or
phototype=finished
```

Sie könnten festlegen, dass die Rohdaten der Fotos irgendwann nach der Erstellung in S3 Glacier archiviert werden. Sie können eine Lebenszyklusregel mit einem Filter konfigurieren, der die Untermenge der Objekte mit dem Schlüsselnamenpräfix (`photos/`) identifiziert, die ein spezifisches Tag (`phototype=raw`) haben.

Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Objektmarkierung und -replikation

Wenn Sie auf Ihrem Bucket die Replikation konfiguriert haben, repliziert Amazon S3 Tags, sofern Sie Amazon S3 die Berechtigung zum Lesen der Tags erteilt haben. Weitere Informationen finden Sie unter Übersicht über die Replikationseinrichtung (p. 674).

Objektmarkierung und Zugriffskontrollrichtlinien

Sie können außerdem Berechtigungsrichtlinien (Bucket- und Benutzerrichtlinien) verwenden, um Berechtigungen für das Objekt-Tagging zu verwalten. Informationen über Richtlinienaktionen finden Sie in den folgenden Themen:

- Beispiel – Objektoperationen (p. 380)
- Beispiel – Bucket-Operationen (p. 381)

Objekt-Tags bieten eine differenzierte Zugriffskontrolle für die Verwaltung von Berechtigungen. Sie gewähren bedingte Berechtigungen auf der Grundlage von Objekt-Tags. Amazon S3 unterstützt die folgenden Bedingungsschlüssel, die Sie verwenden können, um auf Objekt-Tags basierende bedingte Berechnungen zu erteilen:

- `s3:ExistingObjectTag/<tag-key>` – Verwenden Sie diesen Bedingungsschlüssel, um zu überprüfen, ob ein vorhandener Objekt-Tag den spezifischen Tag-Schlüssel und -Wert besitzt.

  Note

  Wenn Sie Berechtigungen für die `PUT Object`- und `DELETE Object`-Operationen erteilen, wird dieser Bedingungsschlüssel nicht unterstützt. Dies bedeutet, dass Sie keine Richtlinie erstellen können, um einem Benutzer zu gestatten, ein Objekt basierend auf seinen vorhandenen Tags zu löschen oder zu überschreiben.
- `s3:RequestObjectTagKeys` – Verwenden Sie diesen Bedingungsschlüssel, um die Tag-Schlüssel einzuschränken, die Sie für Objekte zulassen wollen. Das ist nützlich, wenn Objekten Tags unter Verwendung von PutObjectTagging und PutObject sowie von POST object-Anfragen hinzugefügt werden.
- `s3:RequestObjectTag/<tag-key>` – Verwenden Sie diesen Bedingungsschlüssel, um die Tag-Schlüssel und -Werte einzuschränken, die Sie für Objekte zulassen wollen. Das ist nützlich, wenn Objekten Tags unter Verwendung von PutObjectTagging und PutObject sowie von POST Bucket-Anfragen hinzugefügt werden.

Die vollständige Liste der für den Amazon S3-Service spezifischen Bedingungsschlüssel finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382). Die folgenden Berechtigungsrichtlinien zeigen, wie die Objektmarkierung eine differenzierte Zugriffsberechtigungsverwaltung ermöglicht.

Example 1: Einem Benutzer nur das Lesen von Objekten gestatten, die ein bestimmtes Tag besitzen

Die folgende Berechtigungsrichtlinie erteilt einem Benutzer die Berechtigung, Objekte zu lesen, aber die Bedingung schränkt die Leseberechtigung auf Objekte mit dem folgenden Tag-Schlüssel und -Wert ein.

```
security : public
```

Beachten Sie, dass die Richtlinie den Amazon S3-Bedingungsschlüssel `s3:ExistingObjectTag/<tag-key>` verwendet, um den Schlüssel und den Wert anzugeben.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect":     "Allow",
      "Action":     "s3:GetObject",
      "Resource":    "arn:aws:s3:::examplebucket/*",
      "Principal":   "*",
      "Condition": {  "StringEquals": {"s3:ExistingObjectTag/security": "public" } }
    }
  ]
}
```

Example 2: Einem Benutzer mit Einschränkungen auf die zulässigen Tag-Schlüssel gestatten, Objekt-Tags hinzuzufügen

Die folgende Berechtigungsrichtlinie erteilt einem Benutzer die Berechtigungen, die `s3:PutObjectTagging`-Aktion auszuführen, die dem Benutzer gestattet, einem vorhandenen Objekt Tags hinzuzufügen. Die Bedingung schränkt die Tag-Schlüssel ein, die der Benutzer verwenden darf. Die Bedingung verwendet den Bedingungsschlüssel `s3:RequestObjectTagKeys`, um die Menge der Tag-Schlüssel zu spezifizieren.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ],
      "Principal":{
        "CanonicalUser":[
            "64-digit-alphanumeric-value"
         ]
       },
      "Condition": {
        "ForAllValues:StringLike": {
          "s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        }
      }
```

```
    }
  ]
}
```

Die Richtlinie stellt sicher, dass eine in der Anfrage angegebene Tag-Menge die spezifizierten Schlüssel enthält. Ein Benutzer könnte eine leere Tag-Menge in `PutObjectTagging` senden, was durch diese Richtlinie erlaubt ist (eine leere Tag-Menge in der Anfrage entfernt alle vorhandenen Tags für das Objekt). Wenn Sie verhindern wollen, dass ein Benutzer die Tag-Menge entfernt, können Sie eine weitere Bedingung hinzufügen, um sicherzustellen, dass der Benutzer mindestens einen Wert angibt. Der `ForAnyValue` in der Bedingung stellt sicher, dass mindestens einer der spezifizierten Werte in der Anfrage enthalten ist.

```
{

  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ],
      "Principal":{
        "AWS":[
            "arn:aws:iam::account-number-without-hyphens:user/username"
         ]
       },
      "Condition": {
        "ForAllValues:StringLike": {
          "s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        },
        "ForAnyValue:StringLike": {
          "s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        }
      }
    }
  ]
}
```

Weitere Informationen finden Sie unter Erstellen einer Bedingung zum Testen von mehreren Schlüsselwerten (Set-Operationen) im IAM-Benutzerhandbuch.

Example 3: Einem Benutzer gestatten, Objekt-Tags hinzuzufügen, die einen spezifischen Tag-Schlüssel und -Wert enthalten

Die folgende Benutzerrichtlinie erteilt einem Benutzer die Berechtigungen, die `s3:PutObjectTagging`-Aktion auszuführen, die dem Benutzer gestattet, einem vorhandenen Objekt Tags hinzuzufügen. Die Bedingung fordert, dass der Benutzer ein spezifisches Tag (`Project`) mit dem Wert `X` angibt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ],
      "Principal":{
        "AWS":[
            "arn:aws:iam::account-number-without-hyphens:user/username"
         ]
       },
      "Condition": {
        "StringEquals": {
          "s3:RequestObjectTag/Project": "X"
        }
      }
    }
  ]
}
```

Verwandte Themen

Verwalten von Objekt-Tags (p. 136)

Verwalten von Objekt-Tags

Dieser Abschnitt beschreibt, wie Sie mit dem AWS SDK for Java oder der Amazon S3-Konsole Objekt-Tags programmgesteuert hinzufügen können.

Themen

Verwalten der Objekt-Tags über die Konsole

Sie können mit der Amazon S3-Konsole neuen Objekten Tags hinzufügen, wenn Sie sie hochladen, Sie können Tags aber auch vorhandenen Objekten hinzufügen. Anweisungen zum Hinzufügen von Tags für Objekte mit der Amazon S3-Konsole finden Sie unter Hinzufügen von Objekt-Tags im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwalten von Tags mit dem AWS SDK for Java

Das folgende Beispiel veranschaulicht, wie Sie mit dem AWS SDK for Java Tags für ein neues Objekt festlegen und Tags für ein vorhandenes Objekt abrufen oder ersetzen. Weitere Informationen über das Markieren von Objekten finden Sie unter Markieren von Objekten (p. 130). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;
```

```
import java.io.File;
import java.util.ArrayList;
import java.util.List;

public class ManagingObjectTags {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";
        String filePath = "*** File path ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Create an object, add two new tags, and upload the object to Amazon S3.
            PutObjectRequest putRequest = new PutObjectRequest(bucketName, keyName, new
 File(filePath));
            List<Tag> tags = new ArrayList<Tag>();
            tags.add(new Tag("Tag 1", "This is tag 1"));
            tags.add(new Tag("Tag 2", "This is tag 2"));
            putRequest.setTagging(new ObjectTagging(tags));
            PutObjectResult putResult = s3Client.putObject(putRequest);

            // Retrieve the object's tags.
            GetObjectTaggingRequest getTaggingRequest = new
 GetObjectTaggingRequest(bucketName, keyName);
            GetObjectTaggingResult getTagsResult =
 s3Client.getObjectTagging(getTaggingRequest);

            // Replace the object's tags with two new tags.
            List<Tag> newTags = new ArrayList<Tag>();
            newTags.add(new Tag("Tag 3", "This is tag 3"));
            newTags.add(new Tag("Tag 4", "This is tag 4"));
            s3Client.setObjectTagging(new SetObjectTaggingRequest(bucketName, keyName, new
 ObjectTagging(newTags)));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwalten von Tags unter Verwendung des AWS SDK for .NET

Das folgende Beispiel veranschaulicht, wie Sie mit dem AWS SDK für .NET die Tags für ein neues Objekt festlegen und die Tags für ein vorhandenes Objekt abrufen oder ersetzen. Weitere Informationen über das Markieren von Objekten finden Sie unter Markieren von Objekten (p. 130).

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    public class ObjectTagsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for the new object ***";
        private const string filePath = @"*** file path ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            PutObjectWithTagsTestAsync().Wait();
        }

        static async Task PutObjectWithTagsTestAsync()
        {
            try
            {
                // 1. Put an object with tags.
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    FilePath = filePath,
                    TagSet = new List<Tag>{
                        new Tag { Key = "Keyx1", Value = "Value1"},
                        new Tag { Key = "Keyx2", Value = "Value2" }
                    }
                };

                PutObjectResponse response = await client.PutObjectAsync(putRequest);
                // 2. Retrieve the object's tags.
                GetObjectTaggingRequest getTagsRequest = new GetObjectTaggingRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };

                GetObjectTaggingResponse objectTags = await
 client.GetObjectTaggingAsync(getTagsRequest);
                for (int i = 0; i < objectTags.Tagging.Count; i++)
                    Console.WriteLine("Key: {0}, Value: {1}", objectTags.Tagging[i].Key,
 objectTags.Tagging[i].Value);


                // 3. Replace the tagset.

                Tagging newTagSet = new Tagging();
                newTagSet.TagSet = new List<Tag>{
                    new Tag { Key = "Key3", Value = "Value3"},
                    new Tag { Key = "Key4", Value = "Value4" }
                };


                PutObjectTaggingRequest putObjTagsRequest = new PutObjectTaggingRequest()
                {
                    BucketName = bucketName,
```

```
                    Key = keyName,
                    Tagging = newTagSet
                };
                PutObjectTaggingResponse response2 = await
client.PutObjectTaggingAsync(putObjTagsRequest);

                // 4. Retrieve the object's tags.
                GetObjectTaggingRequest getTagsRequest2 = new GetObjectTaggingRequest();
                getTagsRequest2.BucketName = bucketName;
                getTagsRequest2.Key = keyName;
                GetObjectTaggingResponse objectTags2 = await
client.GetObjectTaggingAsync(getTagsRequest2);
                for (int i = 0; i < objectTags2.Tagging.Count; i++)
                    Console.WriteLine("Key: {0}, Value: {1}", objectTags2.Tagging[i].Key,
objectTags2.Tagging[i].Value);

            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an object"
                        , e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Encountered an error. Message:'{0}' when writing an object"
                    , e.Message);
            }
        }
    }
}
```

Verwaltung des Objektlebenszyklus

Um Ihre Objekte so zu verwalten, dass diese während ihres gesamten Lebenszyklus kosteneffizient gespeichert werden, konfigurieren Sie deren Amazon S3-Lebenszyklus. Eine S3-Lebenszyklus-Konfiguration besteht aus einer Reihe von Regeln, mit denen Aktionen definiert werden, die Amazon S3 auf eine Gruppe von Objekten anwendet. Es gibt zwei Aktionstypen:

- Überführungsaktionen – Definieren, wann Objekte zu einer anderen Speicherklasse überführt werden sollen. Beispielsweise können Sie festlegen, dass Objekte 30 Tage nach ihrer Erstellung in die S3-STANDARD-IA-Speicherklasse (IA steht für „Infrequent Access“, seltener Zugriff) übergehen oder 1 Jahr nach ihrer Erstellung in der Speicherklasse S3 Glacier archiviert werden sollen.

  Mit Lebenszyklus-Überführungsanforderungen sind Kosten verbunden. Informationen zu den Preisen finden Sie unter Amazon S3 – Preise.

- Ablaufaktionen – Definieren, wann Objekte ablaufen. Amazon S3 löscht abgelaufene Objekte in Ihrem Namen.

  Die Lebenszyklusablaufkosten sind von dem Zeitpunkt abhängig, an dem Objekte ablaufen sollen. Weitere Informationen finden Sie unter Grundlegendes zum Objektablauf (p. 146).

Weitere Informationen zu S3-Lebenszyklus-Regeln finden Sie unter Elemente der Lebenszykluskonfiguration (p. 148).

Wann sollte ich eine Lebenszykluskonfiguration verwenden?

Für Objekte mit vorgegebenem Lebenszyklus können Sie S3-Lebenszyklus-Konfigurationsregeln definieren. Beispiel:

- Wenn Sie regelmäßig Protokolle in einen Bucket hochladen, werden diese möglicherweise nur für eine Woche oder einen Monat von Ihrer Anwendung benötigt. Anschließend sollen diese gelöscht werden.
- Auf einige Dokumente erfolgt für einen begrenzten Zeitraum ein häufiger Zugriff. Danach wird nur selten auf diese zugegriffen. Irgendwann benötigen Sie keinen Echtzeitzugriff mehr auf die Dokumente, aber seitens Ihrer Organisation oder der Gesetzgebung müssen Sie diese noch für einen gewissen Zeitraum archivieren. Nach Ablauf dieser Zeitspanne können Sie sie löschen.
- Sie können auch einige Datentypen primär für Archivierungszwecke in Amazon S3 hochladen. Beispielsweise können Sie digitale Medienarchive, Datensätze des Finanz- und Gesundheitswesens, rohe Genomsequenzdaten, langfristige Datenbanksicherungen oder Daten, die zur Einhaltung gesetzlicher Vorschriften aufbewahrt werden müssen, archivieren.

Mithilfe der S3-Lebenszyklus-Konfigurationsregeln können Sie Amazon S3 anweisen, Objekte in kostengünstigere Speicherklassen zu überführen bzw. zu archivieren oder zu löschen.

Wie konfiguriere ich einen Lebenszyklus?

Eine S3-Lebenszyklus-Konfiguration (eine XML-Datei) besteht aus einem Satz von Regeln mit vordefinierten Aktionen, die Amazon S3 für Objekte während ihrer Lebensdauer ausführen soll.

Amazon S3 stellt einen Satz von API-Operationen für die Verwaltung der Lebenszykluskonfiguration für einen Bucket bereit. Amazon S3 speichert die Konfiguration als eine Lebenszyklus-Unterressource, die Ihrem Bucket angefügt ist. Details dazu finden Sie unter:

PUT Bucket-Lebenszyklus

GET Bucket-Lebenszyklus

DELETE Bucket-Lebenszyklus

Sie können den Lebenszyklus auch über die Amazon S3-Konsole oder programmgesteuert über die AWS SDK-Wrapper-Bibliotheken konfigurieren. Wenn nötig, können Sie die REST-API-Aufrufe auch direkt ausführen. Weitere Informationen finden Sie unter Festlegen der Lebenszykluskonfiguration für einen Bucket (p. 165).

Weitere Informationen finden Sie unter den folgenden Themen:

- Weitere Überlegungen zur Lebenszykluskonfiguration (p. 140)
- Elemente der Lebenszykluskonfiguration (p. 148)
- Beispiele der Lebenszykluskonfiguration (p. 155)
- Festlegen der Lebenszykluskonfiguration für einen Bucket (p. 165)

Weitere Überlegungen zur Lebenszykluskonfiguration

Wenn Sie den Lebenszyklus von Objekten konfigurieren möchten, sollten Sie mit den folgenden Richtlinien für den Übergang von Objekten, das Festlegen von Ablaufdaten und weiteren Objektkonfigurationen vertraut sein.

Themen

Übergang von Objekten mit dem Amazon S3-Lebenszyklus

Sie können in einer S3-Lebenszyklus-Konfiguration Regeln hinzufügen, um Amazon S3 anzuweisen, Objekte in eine andere Amazon S3-Speicherklasse zu überführen. Beispiel:

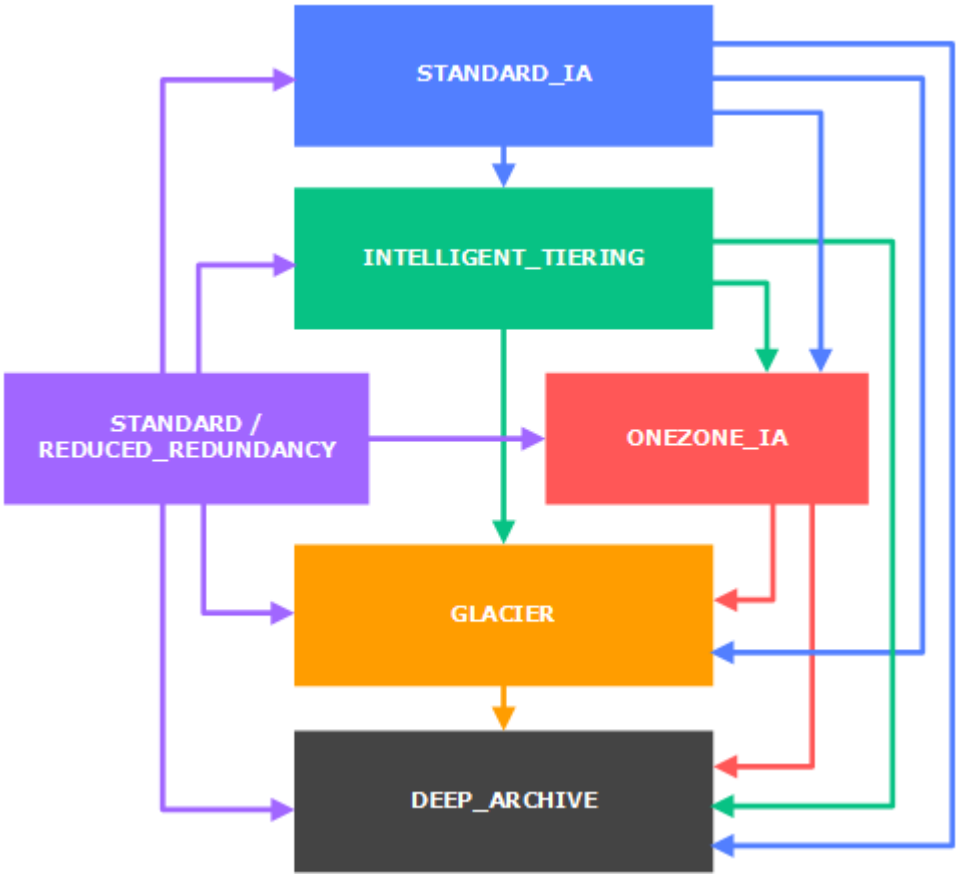
- Wenn Sie wissen, dass nur selten auf bestimmte Objekte zugegriffen wird, können Sie diese in die S3 Standard-IA-Speicherklasse überführen.
- Sie können Objekte, für die kein Echtzeitzugriff mehr benötigt wird, in der Speicherklasse S3 Glacier archivieren.

Die folgenden Abschnitte beschreiben unterstützte Übergange, zugehörige Einschränkungen und den Übergang in die Speicherklasse S3 Glacier.

Unterstützte Transaktionen und zugehörige Einschränkungen

Sie können in einer S3-Lebenszyklus-Konfiguration Regeln definieren, um Objekte von einer Speicherklasse in eine andere Speicherklasse zu überführen, um Speicherkosten zu sparen. Wenn Sie die Zugriffsmuster Ihrer Objekte nicht kennen oder sich Ihre Zugriffsmuster über die Zeit verändern, können Sie die Objekte in die Speicherklasse S3 Intelligent-Tiering überführen, um automatische Kosteneinsparungen zu erzielen. Weitere Informationen über Speicherklassen finden Sie unter Amazon S3-Speicherklassen (p. 122).

Amazon S3 unterstützt das Wasserfallmodell für die Überführung zwischen Speicherklassen wie im folgenden Diagramm gezeigt.



Unterstützte Lebenszyklusübergänge

Amazon S3 unterstützt die folgenden Lebenszyklusübergänge zwischen Speicherklassen mittels einer S3-Lebenszykluskonfiguration.

Die folgenden Übergänge sind möglich:

- Von der S3 Standard-Speicherklasse zur jeder anderen Speicherklasse.
- Jede Speicherklasse zur S3 Glacier- oder S3 Glacier Deep Archive-Speicherklasse.
- Von der S3 Standard-IA-Speicherklasse zu den S3 Intelligent-Tiering- oder S3 One Zone-IA-Speicherklassen.
- Von der S3 Intelligent-Tiering-Speicherklasse zur S3 One Zone-IA-Speicherklasse.
- Die S3 Glacier-Speicherklasse zur S3 Glacier Deep Archive-Speicherklasse.

Nicht unterstützte Lebenszyklusübergänge

Amazon S3 unterstützt keine der folgenden Lebenszyklusübergänge.

Die folgenden Übergänge sind nicht möglich:

- Von jeder Speicherklasse zur S3 Standard-Speicherklasse.
- Von jeder Speicherklasse zur Reduced Redandancy-Speicherklasse.
- Von der S3 Intelligent-Tiering-Speicherklasse zur S3 Standard-IA-Speicherklasse.
- Von der S3 One Zone-IA-Speicherklasse zu den S3 Standard-IA- oder S3 Intelligent-Tiering-Speicherklassen.

Einschränkungen

Für die Übergänge der Lebenszyklus-Speicherklassen gelten folgende Einschränkungen:

Objektgröße und Übergänge von S3 Standard oder S3 Standard-IA auf S3 Intelligent-Tiering, S3 Standard-IA oder S3 One Zone-IA

Wenn Sie Objekte aus den S3 Standard- oder S3 Standard-IA Speicherklassen in S3 Intelligent-Tiering, S3 Standard-IA oder S3 One Zone-IA überführen, gelten die folgenden Objektgrößenbeschränkungen:

- Größere Objekte - Für die folgenden Übergänge gibt es einen Kostenvorteil beim Übergang größerer Objekte:
  - Von den S3 Standard- oder S3 Standard-IA-Speicherklassen zu S3 Intelligent-Tiering.
  - Von der S3 Standard-Speicherklasse zu S3 Standard-IA oder S3 One Zone-IA.
- Objekte kleiner als 128 KB - Für die folgenden Übergänge wird für Objekte, die kleiner als 128 KB sind, kein Übergang in Amazon S3 durchgeführt, da dies nicht kostengünstig ist:
  - Von den S3 Standard- oder S3 Standard-IA-Speicherklassen zu S3 Intelligent-Tiering.
  - Von der S3 Standard-Speicherklasse zu S3 Standard-IA oder S3 One Zone-IA.

Mindesttage für den Übergang von S3 Standard oder S3 Standard-IA in S3 Standard-IA oder S3 One Zone-IA

Bevor Sie Objekte aus den S3 Standard- oder S3 Standard-IA-Speicherklassen in S3 Standard-IA oder S3 One Zone-IA überführen können, müssen Sie sie mindestens 30 Tage lang in der S3 Standard-Speicherklasse speichern. Beispielsweise können Sie keine Lebenszyklusregel erstellen, mit der Objekte einen Tag nach ihrer Erstellung in die Speicherklasse S3 Standard-IA übergehen. Mit Amazon S3 ist kein Übergang von Objekten innerhalb der ersten 30 Tage möglich, da neuere Objekte oft häufiger aufgerufen oder früher gelöscht werden, als dies für S3 Standard-IA- oder S3 One Zone-IA-Speicher geeignet ist.

Beim Übergang von nicht aktuellen Objekten (in versionsfähigen Buckets) können dementsprechend nur Objekte, die seit mindestens 30 Tagen nicht mehr aktuell sind, in S3 Standard-IA- oder S3 One Zone-IA-Speicher übergehen.

Minimale 30-tägige Speichergebühr für S3 Intelligent-Tiering, S3 Standard-IA und S3 One Zone-IA

Für die S3 Intelligent-Tiering-, S3 Standard-IA- und S3 One Zone-IA-Speicherklassen wird eine Mindestspeichergebühr von 30 Tagen berechnet. Daher können Sie keine einzelne Lebenszyklusregel sowohl für einen S3 Intelligent-Tiering-, S3 Standard-IA- oder S3 One Zone-IA-Übergang als auch für einen S3 Glacier- oder S3 Glacier Deep Archive-Übergang angeben, wenn der S3 Glacier- oder S3 Glacier Deep Archive-Übergang weniger als 30 Tage nach dem S3 Intelligent-Tiering-, S3 Standard-IA- oder S3 One Zone-IA-Übergang erfolgt.

Derselbe Mindestzeitraum von 30 Tagen gilt auch für den Übergang von S3 Standard-IA-Speicher in S3 One Zone-IA- oder S3 Intelligent-Tiering-Speicher. Sie können dafür zwei Regeln spezifizieren, jedoch fallen die Mindestgebühren für Speicher an. Weitere Information zu Kostenaspekten finden Sie unter Amazon S3 – Preise.

Verwalten des vollständigen Lebenszyklus eines Objekts

Sie können diese S3-Lebenszyklus-Aktionen kombinieren, um den vollständigen Lebenszyklus eines Objekts zu verwalten. Angenommen, die von Ihnen erstellten Objekte besitzen einen guten definierten Lebenszyklus. Anfänglich erfolgt ein häufiger Zugriff auf die Objekte für einen Zeitraum von 30 Tagen. Danach erfolgt nur noch ein seltener Zugriff für bis zu 90 Tage. Anschließend werden die Objekte nicht mehr benötigt, daher können Sie entscheiden, sie zu archivieren oder zu löschen.

In diesem Szenario erstellen Sie eine S3-Lebenszyklus-Regel, in der Sie eine erste Übergangsaktion in S3 Intelligent-Tiering-, S3 Standard-IA- oder S3 One Zone-IA-Speicher, eine weitere Übergangsaktion zur Archivierung in den S3 Glacier-Speicher und abschließend eine Ablaufaktion festlegen. Beim Übergang

der Objekte von einer Speicherklasse in eine andere können Sie Kosten für Speicher sparen. Weitere Information zu Kostenaspekten finden Sie unter Amazon S3 – Preise.

Übergang zu den Speicherklassen S3 Glacier und S3 Glacier Deep Archive (Objektarchivierung)

Mithilfe der S3-Lebenszyklus-Konfiguration können Sie Objekte zur Archivierung in die Speicherklassen S3 Glacier oder S3 Glacier Deep Archive überführen. Wenn Sie die Speicherklasse S3 Glacier oder S3 Glacier Deep Archive auswählen, bleiben die Objekte in Amazon S3. Sie können nicht direkt über den separaten Amazon S3 Glacier-Service darauf zugreifen.

Bevor Sie Ihre Objekte archivieren, lesen Sie die folgenden Abschnitte, wo Sie weitere relevante Aspekte finden.

Allgemeine Überlegungen

Bevor Sie Objekte archivieren, sollten Sie die folgenden allgemeinen Überlegungen in Betracht ziehen:

- Verschlüsselte Objekte bleiben während des gesamten Übergangsprozesses der Speicherklasse verschlüsselt.
- Objekte, die in den Speicherklassen S3 Glacier oder S3 Glacier Deep Archive gespeichert sind, stehen nicht in Echtzeit zur Verfügung.

  Archivierte Objekte sind Amazon S3-Objekte, aber um auf ein archiviertes Objekt zugreifen zu können, müssen Sie zuerst eine temporäre Kopie davon wiederherstellen. Die wiederhergestellte Objektkopie steht nur für die Dauer zur Verfügung, die Sie in der Wiederherstellungsanforderung angegeben haben. Anschließend löscht Amazon S3 die temporäre Kopie und die Objekte bleiben in Amazon S3 Glacier archiviert.

  Sie können ein Objekt mittels der Amazon S3-Konsole, programmgesteuert mittels der AWS SDK-Wrapper-Bibliotheken oder mittels der Amazon S3-REST-API in Ihrem Code wiederherstellen. Weitere Informationen finden Sie unter Wiederherstellen archivierter Objekte (p. 272).
- Objekte, die in der Speicherklasse S3 Glacier gespeichert sind, können nur in die Speicherklasse S3 Glacier Deep Archive überführt werden.

  Sie können mit einer S3-Lebenszyklus-Konfigurationsregel die Speicherklasse eines Objekts von S3 Glacier nur zur Speicherklasse S3 Glacier Deep Archive ändern. Wenn Sie die Speicherklasse eines in S3 Glacier gespeicherten Objekts zu einer anderen Speicherklasse als S3 Glacier Deep Archive ändern möchten, müssen Sie die Wiederherstellungsoperation verwenden, um zunächst eine temporäre Kopie des Objekts zu erstellen. Anschließend verwenden Sie die Kopieroperation, um das Objekt zu überschreiben und dabei S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-IA oder Reduced Redundancy als Speicherklasse anzugeben.
- Die Überführung von Objekten in die Speicherklasse S3 Glacier Deep Archive ist nicht umkehrbar.

  Sie können eine S3-Lebenszyklus-Konfigurationsregel nicht verwenden, um die Speicherklasse eines Objekts von S3 Glacier Deep Archive in eine andere Speicherklasse zu ändern. Wenn Sie die Speicherklasse eines archivierten Objekts in eine andere Speicherklasse ändern möchten, müssen Sie die Wiederherstellungsoperation verwenden, um zunächst eine temporäre Kopie des Objekts zu erstellen. Überschreiben Sie das Objekt dann mit dem Kopiervorgang, wobei Sie als Speicherklasse S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-IA, S3 Glacier oder Reduced Redundancy angeben.
- Die Objekte, die in den Speicherklassen S3 Glacier und S3 Glacier Deep Archive gespeichert sind, sind sichtbar und stehen nur über Amazon S3 zur Verfügung. Sie sind nicht über den separaten Amazon S3 Glacier-Service verfügbar.

  Es handelt sich dabei um Amazon S3-Objekte und Sie können ausschließlich über die Amazon S3-Konsole oder die Amazon S3-API auf sie zugreifen. Sie können auf die archivierten Objekte nicht über die separate Amazon S3 Glacier-Konsole oder die Amazon S3 Glacier-API zugreifen.

Kostenüberlegungen

Wenn Sie vorhaben, Daten mit seltenem Zugriff für einen Zeitraum von Monaten oder Jahren zu archivieren, können die Speicherklassen S3 Glacier und S3 Glacier Deep Archive Ihre Speicherkosten reduzieren. Um sicherzustellen, dass die Speicherklasse S3 Glacier oder S3 Glacier Deep Archive für Ihre Zwecke geeignet ist, sollten Sie jedoch Folgendes berücksichtigen:

- Gebühren für zusätzlichen Speicheraufwand – Wenn Sie Objekte zur Speicherklasse S3 Glacier oder S3 Glacier Deep Archive überführen, wird jedem Objekt eine feste Speichermenge hinzugefügt, um die Metadaten für die Verwaltung des Objekts zu berücksichtigen.
  - Für jedes in S3 Glacier oder S3 Glacier Deep Archive archivierte Objekt verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten. Amazon S3 speichert diese Metadaten, damit Sie eine Echtzeitliste Ihrer archivierten Objekte über die Amazon S3-API abrufen können. Weitere Informationen finden Sie unter Get Bucket (List Objects). Für diesen zusätzlichen Speicherplatz werden Ihnen Amazon S3-Standardgebühren in Rechnung gestellt.
  - Für jedes Objekt, das in S3 Glacier oder S3 Glacier Deep Archive archiviert wird, fügt Amazon S3 32 KB Speicherplatz für den Index und die zugehörigen Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederherzustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen die S3 Glacier- oder S3 Glacier Deep Archive-Gebühren in Rechnung gestellt.

  Wenn Sie kleine Objekte archivieren, sollten Sie diese Speichergebühren berücksichtigen. Ziehen Sie auch in Betracht, viele kleine Objekte in wenigen großen Objekten zusammenzufassen, um Kosten für den Verwaltungsaufwand zu reduzieren.
- Geplante Archivierungsdauer der Objekte in Tagen – S3 Glacier und S3 Glacier Deep Archive sind Lösungen für die langfristige Archivierung. Die Mindestspeicherdauer beträgt 90 Tage für die Speicherklasse S3 Glacier und 180 Tage für die Speicherklasse S3 Glacier Deep Archive. Das Löschen auf Amazon S3 Glacier gespeicherter Daten ist kostenlos, sofern die zu löschenden Objekte länger als die Mindestspeicherdauer gespeichert wurden. Wenn Sie ein archiviertes Objekt innerhalb der minimalen Speicherdauer löschen oder überschreiben, stellt Amazon S3 eine anteilige Gebühr für das vorzeitige Löschen in Rechnung.
- Gebühren für die Anforderung des Übergangs in die Speicherklassen S3 Glacier und S3 Glacier Deep Archive – Jedes Objekt, das Sie in die S3 Glacier- oder S3 Glacier Deep Archive-Speicherklasse überführen, stellt eine Übergangsanforderung dar. Für jede dieser Anforderungen entstehen Kosten. Wenn Sie vorhaben, sehr viele Objekte zu überführen, sollten Sie die Anforderungskosten in Betracht ziehen. Wenn Sie kleine Objekte archivieren, sollten Sie nach Möglichkeit mehrere kleine Objekte zu großen Objekten zusammenzufassen, um Ihre Archivierungsanforderungskosten zu reduzieren.
- Gebühren für die Wiederherstellung von Daten aus S3 Glacier und S3 Glacier Deep Archive – S3 Glacier und S3 Glacier Deep Archive sind für die langfristige Archivierung von Daten vorgesehen, auf die selten zugegriffen wird. Informationen zu Gebühren für die Wiederherstellung von Daten finden Sie unter Wie viel kostet es, Daten aus Amazon S3 Glacier abzurufen? in den häufig gestellten Fragen zu Amazon S3. Informationen zur Wiederherstellung von Daten aus Amazon S3 Glacier finden Sie unter Wiederherstellen archivierter Objekte (p. 272).

Wenn Sie Objekte unter Verwendung der S3-Lebenszyklus-Verwaltung in Amazon S3 Glacier archivieren, überführt Amazon S3 diese Objekte asynchron. Es kann eine Verzögerung zwischen dem Übergangsdatum in der Lebenszykluskonfigurationsregel und dem Datum des physischen Übergangs geben. Ihnen werden Amazon S3 Glacier-Preise basierend auf dem in der Regel angegebenen Übergangsdatum in Rechnung gestellt. Weitere Informationen finden Sie im Amazon S3 Glacier-Abschnitt unter Amazon S3 – Häufig gestellte Fragen.

Die Amazon S3-Produktdetailseite stellt Preisinformationen und Beispielberechnungen für die Archivierung von Amazon S3-Objekten bereit. Weitere Informationen finden Sie unter den folgenden Themen:

- Wie werden die Speichergebühren für Amazon S3-Objekte berechnet, die in Amazon S3 Glacier archiviert werden?

- Wie wird das Löschen von Objekten aus Amazon S3 Glacier berechnet, die weniger als 3 Monate alt sind?
- Was kostet es, Daten aus Amazon S3 Glacier abzurufen?
- Amazon S3 – Preise für die Speicherkosten der unterschiedlichen Speicherklassen.

Wiederherstellen archivierter Objekte

Ein Zugriff auf archivierte Objekte ist nicht in Echtzeit möglich. Sie müssen zuerst eine Anforderung zur Wiederherstellung initiieren und dann warten, bis eine temporäre Kopie des Objekts für die Dauer bereitgestellt wird, die Sie in der Anforderung angegeben haben. Nachdem Sie eine temporäre Kopie des wiederhergestellten Objekts erhalten haben, bleibt dessen Speicherklasse S3 Glacier oder S3 Glacier Deep Archive. (Eine HEAD-Objekt- oder GET-Objekt-API-Operationsanforderung gibt S3 Glacier oder S3 Glacier Deep Archive als Speicherklasse zurück).

> Note
>
> Bei der Wiederherstellung eines Archivs zahlen Sie sowohl für das Archiv (Tarif S3 Glacier oder S3 Glacier Deep Archive) als auch für die temporär wiederhergestellte Kopie (Reduced Redundancy-Speichertarif). Informationen zu Preisen finden Sie unter Amazon S3 – Preise.

Sie können Objektkopien programmgesteuert oder über die Amazon S3-Konsole wiederherstellen. Amazon S3 verarbeitet jeweils nur eine Wiederherstellungsanforderung pro Objekt. Weitere Informationen finden Sie unter Wiederherstellen archivierter Objekte (p. 272).

Grundlegendes zum Objektablauf

Wenn ein Objekt das Ende seiner Lebensdauer erreicht hat, stellt Amazon S3 sie zum Entfernen in Warteschlangen und entfernt sie asynchron. Es kann eine Verzögerung zwischen dem Ablaufdatum und dem Datum, an dem Amazon S3 ein Objekt entfernt, geben. Es werden keine Gebühren für die Speicherdauer in Rechnung gestellt, die einem abgelaufenen Objekt zugeordnet sind.

Um festzustellen, wann ein Objekt planmäßig abläuft, verwenden Sie die API-Operation HEAD Object oder GET Object. Diese API-Operationen geben Antwort-Header zurück, die entsprechende Informationen enthalten.

Wenn Sie eine S3-Lebenszyklus-Ablaufregel erstellen, die bewirkt, dass Objekte ablaufen, die sich weniger als 30 Tage im S3 Intelligent-Tiering-, S3 Standard-IA- oder S3 One Zone-IA-Speicher befanden, werden Ihnen Gebühren für 30 Tage in Rechnung gestellt. Wenn Sie eine Lebenszyklusablaufregel erstellen, die bewirkt, dass Objekte ablaufen, die sich weniger als 90 Tage im S3 Glacier-Speicher befanden, werden Ihnen Gebühren für 90 Tage in Rechnung gestellt. Wenn Sie eine Lebenszyklusablaufregel erstellen, die bewirkt, dass Objekte ablaufen, die sich weniger als 180 Tage im S3 Glacier Deep Archive-Speicher befanden, werden Ihnen Gebühren für 180 Tage in Rechnung gestellt. Weitere Informationen finden Sie unter Amazon S3 – Preise.

Lebenszyklus- und andere Bucket-Konfigurationen

Neben den S3-Lebenszyklus-Konfigurationen können Sie Ihrem Bucket auch weitere Konfigurationen zuordnen. In diesem Abschnitt wird erläutert, wie sich die S3-Lebenszyklus-Konfiguration auf andere Bucket-Konfigurationen auswirkt.

Lebenszyklus und Versioning

Sie können nicht versionsfähigen Buckets und versionsfähigen Buckets S3-Lebenszyklus-Konfigurationen hinzufügen. Weitere Informationen finden Sie unter Objekt-Versioning (p. 127).

Ein versionsfähiger Bucket behält eine aktuelle Objektversion sowie und null oder mehr langfristige Objektversionen bei. Sie können separate Lebenszyklusrichtlinie für aktuelle und nicht aktuelle Objektversionen definieren.

Weitere Informationen finden Sie unter Elemente der Lebenszykluskonfiguration (p. 148). Weitere Informationen über S3 Versioning finden Sie unter Objekt-Versioning (p. 127).

Lebenszykluskonfiguration auf MFA-fähigen Buckets

Eine Lebenszykluskonfiguration wird auf MFA-fähigen Buckets (Multi-Factor Authentication) nicht unterstützt.

Lebenszyklus und Protokollieren

Amazon S3-Lifecycle-Aktionen werden nicht von der AWS CloudTrail-Protokollierung auf Objektebene erfasst, da CloudTrail API-Anforderungen an externe Amazon S3-Endpunkte erfasst, während S3-Lebenszyklus-Lifecycle-Aktionen mit internen Amazon S3-Endpunkten ausgeführt werden. Amazon S3-Server-Zugriffsprotokolle können in einem S3-Bucket aktiviert werden, um S3-Lebenszyklus-bezogene Aktionen wie Objektumstellung auf eine andere Speicherklasse und Objektablauf zu erfassen, was eine endgültige oder logische Löschung zur Folge hat. Weitere Informationen finden Sie unter Server access logging (Server-Zugriffsprotokollierung) (p. 777).

Wenn die Protokollierung für Ihren Bucket aktiviert ist, melden Amazon S3-Server-Zugriffsprotokolle die Ergebnisse der folgenden Operationen.

Operationsprotokoll	Beschreibung
`S3.EXPIRE.OBJECT`	Amazon S3 löscht das Objekt aufgrund der Lebenszyklusablaufaktion permanent.
`S3.CREATE.DELETEMARKER`	Amazon S3 löscht die aktuelle Version logisch und fügt eine Löschmarkierung in einem Bucket mit aktiviertem Versioning hinzu.
`S3.TRANSITION_SIA.OBJECT`	Amazon S3 überführt das Objekt in die S3 Standard-IA-Speicherklasse.
`S3.TRANSITION_ZIA.OBJECT`	Amazon S3 überführt das Objekt in die S3 One Zone-IA-Speicherklasse.
`S3.TRANSITION_INT.OBJECT`	Amazon S3 überträgt das Objekt in die Speicherklasse „Intelligent-Tiering“.
`S3.TRANSITION.OBJECT`	Amazon S3 initiiert den Übergang von Objekten in die Speicherklasse S3 Glacier.
`S3.TRANSITION_GDA.OBJECT`	Amazon S3 initiiert den Übergang von Objekten in die Speicherklasse S3 Glacier oder S3 Glacier Deep Archive.
`S3.DELETE.UPLOAD`	Amazon S3 bricht nicht vollständige mehrteilige Uploads ab.

> **Note**
>
> Amazon S3-Server-Zugriffsprotokolle werden in der Regel auf Best-Effort-Basis bereitgestellt. Sie sind nicht als vollständige Auflistung aller Amazon S3-Anforderungen vorgesehen.

Weitere Infos

- Elemente der Lebenszykluskonfiguration (p. 148)

- Übergang zu den Speicherklassen S3 Glacier und S3 Glacier Deep Archive (Objektarchivierung) (p. 144)
- Festlegen der Lebenszykluskonfiguration für einen Bucket (p. 165)

Elemente der Lebenszykluskonfiguration

Themen

- ID-Element (p. 148)
- Statuselement (p. 148)
- Filterelement (p. 148)
- Elemente, die Lebenszyklusaktionen beschreiben (p. 151)

Sie geben eine S3-Lebenszyklus-Konfiguration als XML an, die aus einen oder mehreren Lebenszyklusregeln besteht.

```
<LifecycleConfiguration>
    <Rule>
        ...
    </Rule>
    <Rule>
        ...
    </Rule>
</LifecycleConfiguration>
```

Jeder Regel umfasst Folgendes:

- Metadaten für die Regel, mit Regel-ID und einem Status, der anzeigt, ob die Regel aktiviert oder deaktiviert ist. Wenn eine Regel deaktiviert ist, führt Amazon S3 keine in der Regel spezifizierten Aktionen aus.
- Filter, der die Objekte identifiziert, für die die Regel gilt. Sie können einen Filter unter Verwendung eines Objektschlüsselpräfix, eines oder mehrerer Objekt-Tags oder von beidem angeben.
- Eine oder mehrere Übergangs- oder Ablaufaktionen mit einem Datum oder einem Zeitintervall innerhalb der Lebensdauer des Objekts, zu denen Amazon S3 die angegebene Aktion ausführen soll.

In den folgenden Abschnitten werden die XML-Elemente in einer S3-Lebenszyklus-Konfiguration beschrieben. Beispielkonfigurationen finden Sie unter Beispiele der Lebenszykluskonfiguration (p. 155).

ID-Element

Eine S3-Lebenszyklus-Konfiguration kann bis zu 1.000 Regeln enthalten. Das Element <ID> identifiziert eine Regel eindeutig. Die Länge der ID ist auf 255 Zeichen begrenzt.

Statuselement

Der Wert des Elements <Status> kann „Enabled (Aktiviert)“ oder „Disabled (Deaktiviert)“ sein. Wenn eine Regel deaktiviert ist, führt Amazon S3 keine in der Regel definierten Aktionen aus.

Filterelement

Eine Lebenszyklusregel kann für alle Objekte oder eine Untermenge der Objekte in einem Bucket gelten, abhängig vom Element <Filter>, das Sie in der Lebenszyklusregel angeben.

Sie können Objekte nach dem Schlüsselpräfix, Objekt-Tags oder Kombinationen aus beidem filtern (bei einer Kombination verwendet Amazon S3 ein logisches UND für die Filter). Betrachten Sie die folgenden Beispiele:

- Angabe eines Filters unter Verwendung von Schlüsselpräfixen – Dieses Beispiel zeigt eine S3-Lebenszyklus-Regel, die sich auf eine Untermenge von Objekten bezieht, abhängig vom Schlüsselnamenpräfix (`logs/`). Beispielsweise gilt die Lebenszyklusregel für die Objekte `logs/mylog.txt`, `logs/temp1.txt` und `logs/test.txt`. Die Regel gilt nicht für das Objekt `example.jpg`.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>logs/</Prefix>
        </Filter>
        transition/expiration actions.
         ...
    </Rule>
    ...
</LifecycleConfiguration>
```

  Wenn Sie eine Lebenszyklusaktion auf eine Untermenge von Objekten basierend auf unterschiedlichen Schlüsselnamenpräfixen anwenden wollen, müssen Sie separate Regeln angeben. Geben Sie in jeder Regel einen auf einem Präfix basierenden Filter an. Um beispielsweise eine Lebenszyklusaktion für Objekte mit den Schlüsselpräfixen `projectA/` und `projectB/` zu beschreiben, geben Sie zwei Regeln an, die wie folgt aussehen.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>projectA/</Prefix>
        </Filter>
        transition/expiration actions.
         ...
    </Rule>

    <Rule>
        <Filter>
           <Prefix>projectB/</Prefix>
        </Filter>
        transition/expiration actions.
         ...
    </Rule>
</LifecycleConfiguration>
```

  Weitere Informationen über Objektschlüssel finden Sie unter Objektschlüssel (p. 117).

- Angabe eines Filters auf der Basis von Objekt-Tags – Im folgenden Beispiel gibt die Lebenszyklusregel einen Filter basierend auf einem Tag (*key*) und einem Wert (*value*) an. Die Regel wird dann nur auf eine Untermenge von Objekten mit dem spezifischen Tag angewendet.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Tag>
              <Key>key</Key>
              <Value>value</Value>
           </Tag>
        </Filter>
        transition/expiration actions.
        ...
```

```
    </Rule>
</LifecycleConfiguration>
```

Sie können einen Filter auf mehreren Tags basierend angeben. Sie müssen die Tags mit dem Element **<AND>** umschließen wie im folgenden Beispiel gezeigt. Die Regel weist Amazon S3 an, Lebenszyklusaktionen für Objekte mit zwei Tags auszuführen (mit dem spezifischen Tagschlüssel und -wert).

```
<LifecycleConfiguration>
    <Rule>
      <Filter>
         <And>
            <Tag>
               <Key>key1</Key>
               <Value>value1</Value>
            </Tag>
            <Tag>
               <Key>key2</Key>
               <Value>value2</Value>
            </Tag>
             ...
         </And>
      </Filter>
      transition/expiration actions.
    </Rule>
</Lifecycle>
```

Die Lebenszyklusregel gilt für Objekte, für die beide Tags angegeben wurden. Amazon S3 führt eine logische AND-Operation aus. Beachten Sie Folgendes:

- Jedes Tag muss exakt mit dem Schlüssel und dem Wert übereinstimmen.
- Die Regel gilt für eine Untermenge der Objekte, die ein oder mehrere der in der Regel spezifizierte Tags besitzen. Falls im Objekt weitere Tags angegeben sind, ist das nicht relevant.

  Note

  Wenn Sie mehrere Tags in einem Filter spezifizieren, muss jeder Tag-Schlüssel eindeutig sein.

- Angabe eines Filters auf der Basis eines Präfixes und mindestens eines Tags – Sie können in einer Lebenszyklusregel einen Filter angeben, der sowohl auf dem Schlüsselpräfix als auch auf mindestens einem Tag basiert. Auch hier müssen Sie all dies mit dem Element **<And>** umschließen, wie im folgenden Beispiel gezeigt.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
          <And>
             <Prefix>key-prefix</Prefix>
             <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
             </Tag>
             <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
             </Tag>
              ...
          </And>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions.
    </Rule>
```

```
</LifecycleConfiguration>
```

Amazon S3 kombiniert diese Filter unter Verwendung einer logischen UND-Operation. Das bedeutet, die Regel wird auf eine Untermenge von Objekten mit einem spezifischen Schlüsselpräfix und spezifischen Tag angewendet. Ein Filter kann höchstens ein Präfix und null oder mehr Tags aufweisen.

- Sie können einen leeren Filter angeben, dann gilt die Regel für alle Objekte in dem Bucket.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions.
    </Rule>
</LifecycleConfiguration>
```

Elemente, die Lebenszyklusaktionen beschreiben

Sie können Amazon S3 anweisen, spezifische Aktionen innerhalb der Lebensdauer eines Objekts auszuführen, indem Sie eine oder mehrere vordefinierte Aktionen in einer S3-Lebenszyklus-Regel angeben. Die Wirkung dieser Aktionen ist vom Versioning-Status Ihres Buckets abhängig.

- Transition-Aktionselement – Sie geben die `Transition`-Aktion an, um Objekte von einer Speicherklasse in eine andere zu überführen. Weitere Informationen zum Übergang von Objekten finden Sie unter Unterstützte Transaktionen und zugehörige Einschränkungen (p. 141). Wenn ein vorgegebenes Datum oder einem Zeitintervall innerhalb der Lebensdauer des Objekts erreicht ist, führt Amazon S3 den Übergang aus.

  Für einen versionsfähigen Bucket (Bucket mit aktiviertem oder ausgesetztem Versioning) wird die `Transition`-Aktion auf die aktuelle Objektversion angewendet. Für die Verwaltung nicht aktueller Versionen definiert Amazon S3 die `NoncurrentVersionTransition`-Aktion (unten beschrieben).

- Ablaufaktionselement – Die `Expiration`-Aktion lässt Objekte ablaufen, die in der Regel genannt werden, und gilt für entsprechende Objekte in einer der Amazon S3-Speicherklassen. Weitere Informationen zu Speicherklassen finden Sie unter Amazon S3-Speicherklassen (p. 122). Abgelaufene Objekte sind in Amazon S3 nicht mehr verfügbar. Ob die Objekte dauerhaft entfernt werden, ist vom Versioning-Status des Buckets abhängig.

  > **Important**
  >
  > Objektablauf-Lebenszyklusrichtlinien entfernen keine unvollständigen mehrteiligen Uploads. Um unvollständige mehrteilige Uploads zu entfernen, müssen Sie die Lebenszykluskonfigurationsaktion AbortIncompleteMultipartUpload verwenden, die später in diesem Abschnitt beschrieben wird.

  - Bucket ohne Versioning – Die `Expiration`-Aktion führt zur dauerhaften Entfernung des Objekts durch Amazon S3.
  - Bucket mit Versioning – Für einen versionsfähigen Bucket (d. h. einen Bucket mit aktiviertem oder ausgesetztem Versioning) gibt es mehrere Aspekte, die die Behandlung der `expiration`-Aktion durch Amazon S3 festlegen. Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514). Unabhängig vom Versioning-Status gilt Folgendes:
    - Die `Expiration`-Aktion wird nur auf die aktuelle Version angewendet (sie wirkt sich nicht auf nicht aktuelle Objektversionen aus).
    - Amazon S3 führt keine Aktion aus, wenn es eine oder mehrere Objektversionen gibt und die Löschmarkierung die aktuelle Version ist.

- Wenn die aktuelle Objektversion die einzige Objektversion und außerdem eine Löschmarkierung ist (auch als Löschmarkierung eines abgelaufenen Objekts bezeichnet, wobei alle Objektversionen gelöscht wurden und nur noch die Löschmarkierung vorhanden ist), entfernt Amazon S3 die Löschmarkierung des abgelaufenen Objekts. Sie können die Ablaufaktion auch verwenden, um Amazon S3 anzuweisen, alle abgelaufenen Löschmarkierungen zu entfernen. Ein Beispiel finden Sie unter Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte (p. 163).

Berücksichtigen Sie auch die folgenden Punkte, wenn Sie Amazon S3 zur Ablaufverwaltung einrichten:

- Bucket mit aktiviertem Versioning

  Wenn die aktuelle Objektversion keine Löschmarkierung ist, fügt Amazon S3 die Löschmarkierung mit einer eindeutigen Versions-ID hinzu. Damit ist die aktuelle Version nicht mehr aktuell und die Löschmarkierung wird zur aktuellen Version.

- Bucket mit ausgesetztem Versioning

  Für einen Bucket mit ausgesetztem Versioning bewirkt die Ablaufaktion, dass Amazon S3 eine Löschmarkierung mit der Versions-ID null erstellt. Diese Löschmarkierung ersetzt jede Objektversion mi einer Versions-ID von null in der Versionshierarchie, womit das Objekt effektiv gelöscht wird.

Darüber hinaus unterstützt Amazon S3 die folgenden Aktionen, mit denen Sie nicht aktuelle Objektversionen in einem versionsfähigen Bucket verwalten können (d. h. für Buckets mit aktivierten und ausgesetztem Versioning).

- NoncurrentVersionTransition-Aktionselement – Mit dieser Aktion können Sie angeben, wie lange (ab dem Zeitpunkt, ab dem die Objekte nicht mehr aktuell sind) die Objekte in der aktuellen Speicherklasse bleiben sollen, bevor Amazon S3 sie in die angegebene Speicherklasse überträgt. Weitere Informationen zum Übergang von Objekten finden Sie unter Unterstützte Transaktionen und zugehörige Einschränkungen (p. 141).
- NoncurrentVersionExpiration – Mit dieser Aktion können Sie angeben, wie lange (ab dem Zeitpunkt, ab dem die Objekte nicht mehr aktuell sind) nicht aktuelle Objektversionen beibehalten werden sollen, bevor Amazon S3 sie dauerhaft entfernt. Das gelöschte Objekt kann nicht wiederhergestellt werden.

  Das verzögerte Entfernen nicht aktueller Objekte kann hilfreich sein, wenn Sie versehentliche Lösch- oder Überschreibvorgänge korrigieren müssen. Beispielsweise können Sie eine Ablaufregel konfigurieren, um nicht aktuelle Versionen fünf Tage nach dem Zeitpunkt zu löschen, zu dem sie nicht aktuell geworden sind. Angenommen, Sie erstellen am 01.01.2014 um 10:30 AM UTC das Objekt `photo.gif` (Versions-ID 111111). Am 2.1.2014 um 11:30 AM UTC löschen Sie versehentlich `photo.gif` (Versions-ID 111111), wodurch eine Löschmarkierung mit einer neuen Versions-ID erstellt wird (z. B. Versions-ID 4857693). Jetzt haben Sie fünf Tage Zeit, die Originalversion von `photo.gif` (Versions-ID 111111) wiederherzustellen, bevor das Löschen permanent wird. Am 8.1.2014 um 00:00 UTC wird die Lebenszyklusregel für den Ablauf ausgeführt und löscht `photo.gif` (Versions-ID 111111) permanent. Dies erfolgt fünf Tage, nachdem es zu einer nicht aktuellen Version geworden ist.

  Important

  Objektablauf-Lebenszyklusrichtlinien entfernen keine unvollständigen mehrteiligen Uploads. Um unvollständige mehrteilige Uploads zu entfernen, müssen Sie die Lebenszykluskonfigurationsaktion AbortIncompleteMultipartUpload verwenden, die später in diesem Abschnitt beschrieben wird.

Neben den Übergangs- und Ablaufaktionen können Sie die folgende Lebenszykluskonfigurationsaktion verwenden, um Amazon S3 anzuweisen, unvollständige mehrteilige Uploads abzubrechen.

- AbortIncompleteMultipartUpload – Mit diesem Element können Sie eine maximale Zeitdauer (in Tagen) festlegen, für den Sie die Bearbeitung mehrteiliger Uploads zulassen möchten. Wenn die jeweiligen mehrteiligen Uploads (festgelegt durch das in der Lebenszyklusregel angegebene `prefix`

des Schlüsselnamens) nicht innerhalb des vordefinierten Zeitraums erfolgreich abgeschlossen werden, bricht Amazon S3 die unvollständigen mehrteiligen Uploads ab. Weitere Informationen finden Sie unter Abbruch unvollständiger mehrteiliger Uploads unter Verwendung einer Bucket-Lebenszyklusrichtlinie (p. 201).

Note

Sie können diese Lebenszyklusaktion nicht in einer Regel angeben, die einen auf Objekt-Tags basierten Filter verwendet.

- ExpiredObjectDeleteMarker – In einem Bucket mit aktiviertem Versioning wird eine Löschmarkierung, die keine nicht aktuellen Versionen enthält, als Löschmarkierung für abgelaufene Objekte bezeichnet. Sie können diese Lebenszyklusaktion verwenden, um S3 anzuweisen, die Löschmarkierungen der abgelaufenen Objekte zu entfernen. Ein Beispiel finden Sie unter Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte (p. 163).

  Note

  Sie können diese Lebenszyklusaktion nicht in einer Regel angeben, die einen auf Objekt-Tags basierten Filter verwendet.

Vorgehensweise, mit der Amazon S3 berechnet, wie lange ein Objekt schon nicht mehr aktuell ist

In einem versionsfähigen Bucket können Sie mehrere Versionen eines Objekts haben. Es gibt immer eine aktuelle Version und null oder mehr nicht aktuelle Versionen. Immer wenn Sie ein Objekt hochladen, wird die aktuelle Version als die nicht aktuelle Version beibehalten, und die neu hinzugefügte Version, der Nachfolger, wird zur aktuellen Version. Um die Anzahl der Tage zu ermitteln, die ein Objekt nicht mehr aktuell ist, untersucht Amazon S3 das Datum, an dem das Nachfolgeobjekt erstellt wurde. Amazon S3 verwendet die Anzahl der Tage seit Erstellung des Nachfolgeobjekts als Anzahl der Tage, die ein Objekt nicht mehr aktuell ist.

Wiederherstellung vorheriger Versionen eines Objekts bei Verwendung von Lebenszykluskonfigurationen

Wie im Thema Wiederherstellen früherer Versionen (p. 532) detailliert erklärt, können Sie eine der beiden folgenden Methoden verwenden, um vorherige Versionen eines Objekts abzurufen:

1. Durch Kopieren einer nicht aktuellen Version des Objekts in denselben Bucket. Das kopierte Objekt wird zur aktuellen Version dieses Objekts, und alle Objektversionen werden beibehalten.
2. Durch das dauerhafte Löschen der aktuellen Version des Objekts. Wenn Sie die aktuelle Objektversion löschen, wandeln Sie letztlich die nicht aktuelle Version in die aktuelle Version dieses Objekts um.

Wenn Sie S3-Lebenszyklus-Konfigurationsregeln für Buckets mit aktiviertem Versioning verwenden, empfehlen wir Ihnen, die erste Methode zu verwenden, was sich bewährt hat. Aufgrund der Eventually Consistent-Semantik von Amazon S3 verschwindet eine aktuelle Version, die Sie dauerhaft gelöscht haben, möglicherweise erst, wenn die Änderungen propagiert werden (Amazon S3 ist diese Löschoperation möglicherweise nicht bekannt). In der Zwischenzeit kann die Lebenszyklusregel, die Sie für den Ablauf nicht aktueller Objekte konfiguriert haben, die nicht aktuellen Objekte dauerhaft entfernen, auch dasjenige, das Sie wiederherstellen möchten. Das Kopieren der alten Version, wie in der ersten Methode empfohlen, ist deshalb die sicherere Alternative.

Lebenszyklusregeln: Basierend auf dem Alter eines Objekts

Sie können ein Zeitintervall als Anzahl der Tage ab der Erstellung (oder Änderung) der Objekte angeben, wann Amazon S3 die Aktion ausführen kann.

Wenn Sie die Anzahl der Tage in den `Transition`- und `Expiration`-Aktionen in einer S3-Lebenszyklus-Konfiguration angeben, beachten Sie Folgendes:

- Dies ist die Anzahl der Tage nach der Erstellung des Objekts, wann die Aktion stattfindet.
- Amazon S3 berechnet die Zeit, indem es die in der Regel angegebene Anzahl an Tagen zur Zeit der Objekterstellung hinzufügt und die resultierende Zeit auf die UTC des nächsten Tags um Mitternacht rundet. Wurde ein Objekt beispielsweise am 15.1.2014 um 10:30 AM UTC erstellt und Sie geben in einer Übergangsregel 3 Tage an, wird das Übergangsdatum des Objekts für den 19.1.2014 um 00:00 UTC berechnet.

  Note

  Amazon S3 behält nur das letzte Änderungsdatum für jedes Objekt bei. Beispielsweise zeigt die Amazon S3-Konsole das Datum Last Modified (Zuletzt geändert) im Bereich Properties (Eigenschaften) des Objekts an. Wenn Sie ein neues Objekt erstellen, ist dieses Datum das Datum, zu dem das Objekt erstellt wurde. Wenn Sie das Objekt ersetzen, ändert sich das Datum entsprechend. Der Begriff Erstellungsdatum ist daher gleichbedeutend mit dem Begriff letztes Änderungsdatum.

Wenn Sie die Anzahl der Tage in den `NoncurrentVersionTransition`- und `NoncurrentVersionExpiration`-Aktionen in einer Lebenszykluskonfiguration angeben, beachten Sie Folgendes:

- Dies ist die Anzahl der Tage, ab dem Zeitpunkt, an dem die Version des Objekts nicht aktuell wird (d. h. der Zeitpunkt, an dem das Objekt überschrieben oder gelöscht wird). Dies gilt für die Version des Objekts, für das Amazon S3 die Aktion ausführen wird.
- Amazon S3 berechnet die Zeit, indem es die in der Regel angegebene Anzahl an Tagen der Zeit hinzufügt, zu der die neue Nachfolgerversion des Objekts erstellt wurde, und die resultierende Zeit auf die UTC des nächsten Tags um Mitternacht rundet. Angenommen, Sie haben in Ihrem Bucket eine aktuelle Version eines Objekts, das am 1.1.2014 um 10:30 AM UTC erstellt wurde. Wenn die neue Version des Objekts, die die aktuelle Version ersetzt, am 15.1.2014 um 10:30 AM UTC erstellt wird und Sie in einer Übergangsregel 3 Tage angeben, wird das Übergangsdatum für das Objekt für den 19.1.2014 um 00:00 UTC berechnet.

Lebenszyklusregeln: Basierend auf einem spezifischen Datum

Wenn Sie in einer S3-Lebenszyklus-Regel eine Aktion angeben, können Sie ein Datum angeben, wann Amazon S3 die Aktion ausführen soll. Wenn das spezifische Datum erreicht ist, wendet Amazon S3 die Aktion auf alle qualifizierten Objekte an (basierend auf den Filterkriterien).

Wenn Sie eine S3-Lebenszyklus-Aktion mit einem Datum angeben, das in der Vergangenheit liegt, kommen sofort alle qualifizierten Objekte für diese Lebenszyklusaktion in Frage.

Important

Die datumsbasierte Aktion ist keine einmalige Aktion. Amazon S3 wendet weiterhin die datumsbasierte Aktion an, auch nach Ablauf des Datums, solange der Regelstatus `Enabled` lautet.
Angenommen, Sie geben eine auf einem Datum basierende `Expiration`-Aktion an, um alle Objekte zu löschen (unter der Annahme, dass in der Regel kein Filter angegeben ist). Amazon S3 lässt zu dem angegebenen Datum alle Objekte in dem Bucket ablaufen. S3 lässt auch weiterhin alle neuen Objekte ablaufen, die Sie in dem Bucket erstellen. Um die Lebenszyklusaktion zu unterbrechen, müssen Sie die Aktion aus der Lebenszykluskonfiguration entfernen, die Regel deaktivieren oder die Regel aus der Lebenszykluskonfiguration löschen.

Der Datumswert muss konform zum Format ISO 8601 angegeben werden. Die Uhrzeit ist stets Mitternacht UTC.

Note

Sie können die auf dem Datum basierenden Lebenszyklusregeln nicht über die Amazon S3-Konsole erstellen, aber Sie können solche Regeln anzeigen, deaktivieren oder löschen.

Beispiele der Lebenszykluskonfiguration

Dieser Abschnitt enthält Beispiele für die S3-Lebenszyklus-Konfiguration. Jedes Beispiel zeigt, wie Sie in jedem der Beispielszenarien das XML spezifizieren können.

Themen

Beispiel 1: Festlegen eines Filters

Jede S3-Lebenszyklus-Regel enthält einen Filter, mit dem Sie eine Untermenge der Objekte in Ihrem Bucket identifizieren können, auf die sich die Lebenszyklusregel bezieht. Das folgenden S3 Lifecycle-Konfigurationen zeigen Beispiele dafür, wie Sie einen Filter spezifizieren können.

- In dieser Lebenszykluskonfigurationregel spezifiziert der Filter ein Schlüsselpräfix (`tax/`). Aus diesem Grund gilt die Regel für Objekte mit dem Schlüsselnamenpräfix `tax/`, wie beispielsweise `tax/doc1.txt` und `tax/doc2.txt`.

  Die Regel spezifiziert zwei Aktionen, die Amazon S3 zu Folgendem anweisen:

  - Übergang von Objekten in die Speicherklasse S3 Glacier 365 Tage (ein Jahr) nach der Erstellung.
  - Objekte 3.650 Tage (10 Jahre) nach der Erstellung löschen (die `Expiration`-Aktion).

```
<LifecycleConfiguration>
  <Rule>
    <ID>Transition and Expiration Rule</ID>
    <Filter>
       <Prefix>tax/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>365</Days>
      <StorageClass>S3 Glacier</StorageClass>
    </Transition>
    <Expiration>
      <Days>3650</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

Statt das Objektalter in Tagen nach der Erstellung zu spezifizieren, können Sie für jede Aktion ein Datum festlegen. Sie können `Date` und `Days` nicht in derselben Regel kombinieren.

- Wenn Sie wollen, dass die Lebenszyklusregel für alle Objekte im Bucket gilt, geben Sie ein leeres Präfix an. In der folgenden Konfiguration gibt die Regel eine `Transition`-Aktion an, die Amazon S3 anweist, Objekte 0 Tage nach ihrer Erstellung zur Speicherklasse S3 Glacier zu überführen. In diesem Fall können Objekte um Mitternacht UTC nach ihrer Erstellung in Amazon S3 Glacier archiviert werden.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Archive all object same-day upon creation</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>S3 Glacier</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

- Sie können null oder mehrere Schlüsselnamenpräfixe und null oder mehr Objekt-Tags in einem Filter angeben. Der folgende Beispiel-Code wendet die Lebenszyklusregel auf eine Untermenge von Objekten mit dem Schüsselpräfix `tax/` an, ebenso wie auf Objekte mit zwei Tags mit spezifischem Schüssel und Wert. Beachten Sie, dass Sie bei der Angabe von mehr als einem Filter eine UND-Verknüpfung verwenden müssen, wie gezeigt (Amazon S3 wendet ein logisches UND an und kombiniert die spezifizierten Filterbedingungen).

```
...
<Filter>
   <And>
      <Prefix>tax/</Prefix>
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
         <Value>value2</Value>
      </Tag>
    </And>
</Filter>
...
```

- Sie können Objekte nur auf Tags basierend filtern. Die folgende Lebenszyklusregel beispielsweise wird auf Objekte angewendet, die die beiden spezifizierten Tags aufweisen (es wird kein Präfix angegeben).

```
...
<Filter>
   <And>
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
         <Value>value2</Value>
      </Tag>
    </And>
</Filter>
```

```
...
```

Important

Wenn Sie mehrere Regeln in einer S3-Lebenszyklus-Konfiguration haben, kann es sein, dass für ein Objekt mehrere Lebenszyklusaktionen zulässig werden. In solchen Fällen folgt Amazon S3 diesen allgemeinen Regeln:

- Das permanente Löschen hat Vorrang vor einem Übergang.
- Der Übergang hat Vorrang vor der Erstellung von Löschmarkierungen.
- Wenn ein Objekt sowohl für einen S3 Glacier- als auch einen S3 Standard-IA (oder S3 One Zone-IA)-Übergang berechtigt ist, wählt Amazon S3 den S3 Glacier-Übergang.

Beispiele finden Sie unter Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklusaktionen und Aktionen von Amazon S3 (p. 159).

Beispiel 2: Deaktivieren einer Lebenszyklusregel

Lebenszyklusregeln können vorübergehend deaktiviert werden. Die folgende Lebenszykluskonfiguration spezifiziert zwei Regeln:

- Regel 1 weist Amazon S3 an, Objekte mit dem Präfix `logs/` bald nach ihrer Erstellung zur Speicherklasse S3 Glacier zu überführen.
- Regel 2 weist Amazon S3 an, Objekte mit dem Präfix `documents/` bald nach ihrer Erstellung zur Speicherklasse S3 Glacier zu überführen.

In der Richtlinie ist die Regel 1 aktiviert und die Regel 2 deaktiviert. Amazon S3 führt keine Aktionen für deaktivierte Regeln aus.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule1</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>S3 Glacier</StorageClass>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule2</ID>
    <Prefix>documents/</Prefix>
    <Status>Disabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>S3 Glacier</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

Beispiel 3: Schichtweise Reduzierung der Speicherklasse über die Lebensdauer des Objekts

In diesem Beispiel nutzen Sie die Lebenszykluskonfiguration, um die Speicherklasse von Objekten über ihre Lebensdauer stufenweise zu reduzieren. Diese schichtweise Reduzierung kann dazu beitragen, Speicherkosten zu reduzieren. Weitere Informationen zu Preisen erhalten Sie unter Amazon S3 – Preise.

Die folgende S3-Lebenszyklus-Konfiguration spezifiziert eine Regel, die auf Objekte mit dem Schlüsselnamenpräfix `logs/` angewendet wird. Die Regel definiert die folgenden Aktionen:

- Zwei Übergangsaktionen:
  - Übergang von Objekten in die S3 Standard-IA-Speicherklasse 30 Tage nach der Erstellung.
  - Übergang von Objekten in die Speicherklasse S3 Glacier 90 Tage nach der Erstellung.
- Eine Ablaufaktion, die Amazon S3 anweist, Objekte ein Jahr nach ihrer Erstellung zu löschen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>example-id</ID>
    <Filter>
       <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>30</Days>
      <StorageClass>STANDARD_IA</StorageClass>
    </Transition>
    <Transition>
      <Days>90</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

> **Note**
>
> Sie können eine Regel verwenden, um alle Lebenszyklusaktionen zu beschreiben, die für dieselbe Objektmenge angewendet werden (identifiziert durch den Filter). Andernfalls können Sie mehrere Regeln hinzufügen, die jeweils einen unterschiedlichen Filter angeben.

Beispiel 4: Festlegen mehrerer Regeln

Sie können mehrere Regeln angeben, wenn Sie unterschiedliche Lebenszyklusaktionen auf unterschiedliche Objekte anwenden wollen. Die folgende Lebenszykluskonfiguration spezifiziert zwei Regeln:

- Regel 1 gilt für Objekte mit dem Schlüsselnamenpräfix `classA/`. Sie weist Amazon S3 an, Objekte ein Jahr nach der Erstellung in die Speicherklasse S3 Glacier zu übertragen, und diese Objekte 10 Jahre nach dem Erstellen ablaufen zu lassen.
- Regel 2 gilt für Objekte mit dem Schlüsselnamenpräfix `classB/`. Amazon S3 wird angewiesen, Objekte 90 Tage nach der Erstellung in die S3 Standard-IA-Speicherklasse zu überführen und sie ein Jahr nach dem zu löschen.

```
<LifecycleConfiguration>
```

```
    <Rule>
        <ID>ClassADocRule</ID>
        <Filter>
           <Prefix>classA/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>365</Days>
           <StorageClass>GLACIER</StorageClass>
        </Transition>
        <Expiration>
             <Days>3650</Days>
        </Expiration>
    </Rule>
    <Rule>
        <ID>ClassBDocRule</ID>
        <Filter>
            <Prefix>classB/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>90</Days>
           <StorageClass>STANDARD_IA</StorageClass>
        </Transition>
        <Expiration>
             <Days>365</Days>
        </Expiration>
    </Rule>
</LifecycleConfiguration>
```

Beispiel 5: Überlappende Filter, widersprüchliche Lebenszyklusaktionen und Aktionen von Amazon S3

Sie könnten eine S3-Lebenszyklus-Konfiguration angeben, in der Sie überlappende Präfixe oder Aktionen spezifizieren.

Im Allgemeinen gibt Amazon S3 Lifecycle Kostenoptimierung Vorrang. Wenn sich z. B. zwei Ablaufrichtlinien überschneiden, wird die Ablaufrichtlinie mit der kürzeren Frist durchgesetzt, sodass die Daten nicht länger als erwartet gespeichert werden.

Wenn sich zwei Übergangsrichtlinien überschneiden, überführt S3-Lebenszyklus Ihre Objekte in die Speicherklasse mit geringeren Kosten. In beiden Fällen versucht S3-Lebenszyklus, den Pfad auszuwählen, der für Sie am kostengünstigsten ist. Eine Ausnahme von dieser allgemeinen Regel ist die S3 Intelligent-Tiering-Speicherklasse. S3 Intelligent-Tiering wird von S3 Lifecycle gegenüber anderen Speicherklassen bevorzugt, abgesehen von den Speicherklassen S3 Glacier und S3 Glacier Deep Archive.

Die folgenden Beispiele zeigen, wie Amazon S3 potenzielle Konflikte auflöst.

Example 1: Überlappende Präfixe (kein Konflikt)

Die folgende Beispielkonfiguration weist zwei Regeln auf, die überlappenden Präfixe spezifizieren, wie folgt:

- Die erste Regel spezifiziert einen leeren Filter, d. h. alle Objekte in dem Bucket werden angesprochen.
- Die zweite Regel spezifiziert ein Schlüsselnamenpräfix `logs/`, d. h. nur eine Untermenge von Objekten.

Regel 1 fordert Amazon S3 auf, alle Objekte ein Jahr nach der Erstellung zu löschen. Regel 2 fordert Amazon S3 auf, eine Teilmenge der Objekte 30 Tage nach der Erstellung in die S3 Standard-IA-Speicherklasse zu überführen.

```
<LifecycleConfiguration>
```

```
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>30</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

Example 2: Widersprüchliche Lebenszyklusaktionen

In dieser Beispielkonfiguration gibt es zwei Regeln, die Amazon S3 anweisen, zwei unterschiedliche Aktionen für dieselbe Objektmenge zur selben Zeit in der Lebensdauer des Objekts auszuführen:

- Beide Regeln geben dasselbe Schlüsselnamenpräfix an, deshalb gelten beide Regeln für dieselbe Objektmenge.
- Beide Regeln spezifizieren dieselben 365 Tage nach der Erstellung des Objekts, wann die Regeln angewendet werden sollen.
- Eine Regel weist Amazon S3 an, Objekte zur S3 Standard-IA-Speicherklasse zu überführen. Eine andere Regel weist Amazon S3 an, die Objekte zur gleichen Zeit ablaufen zu lassen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>365</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

In diesem Fall wollen Sie, dass Objekte ablaufen (entfernt werden), deshalb macht es keinen Sinn, die Speicherklasse zu ändern, und Amazon S3 wählt einfach die Ablaufaktion für diese Objekte aus.

Example 3: Überlappende Präfixe, die zu widersprüchlichen Lebenszyklusaktionen führen

In diesem Beispiel besitzt die Konfiguration zwei Regeln, die überlappende Präfixe angeben, wie folgt:

- Regel 1 legt ein leeres Präfix fest (was für alle Objekte gilt).
- Regel 2 spezifiziert ein Schlüsselnamenpräfix (`logs/`), das eine Untermenge aller Objekte angibt.

Für die Untermenge der Objekte mit dem Schlüsselnamenpräfix `logs/` werden die Lebenszyklusaktionen aus beiden Regeln angewendet. Eine Regel weist Amazon S3 an, Objekte 10 Tage nach ihrer Erstellung zu überführen. Eine andere Regel weist Amazon S3 an, Objekte 365 Tage nach ihrer Erstellung zu überführen.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>10</Days>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>365</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

In diesem Fall entscheidet Amazon S3 den Übergang 10 Tage nach der Erstellung auszuführen.

Example 4: Tag-basiertes Filtern, das zu widersprüchlichen Lebenszyklusaktionen führt

Angenommen, Sie haben die folgende S3-Lebenszyklus-Richtlinie, die zwei Regeln enthält, die beide einen Tag-Filter spezifizieren:

- Regel 1 spezifiziert einen Tag-basierten Filter (`tag1/value1`). Diese Regel weist Amazon S3 an, Objekte 365 Tage nach der Erstellung in die Speicherklasse S3 Glacier zu übertragen.
- Regel 2 spezifiziert einen Tag-basierten Filter (`tag2/value2`). Diese Regel weist Amazon S3 an, Objekte 14 Tage nach der Erstellung ablaufen zu lassen.

Die Lebenszykluskonfiguration wird im Folgenden angezeigt.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Tag>
         <Key>tag1</Key>
         <Value>value1</Value>
      </Tag>
```

```
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>GLACIER<StorageClass>
      <Days>365</Days>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Tag>
         <Key>tag2</Key>
         <Value>value1</Value>
      </Tag>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>14</Days>
    </Expiration>
   </Rule>
</LifecycleConfiguration>
```

Die Richtlinie ist in Ordnung, aber wenn es ein Objekt mit beiden Tags gibt, muss S3 entscheiden, was zu tun ist. Das bedeutet, beide Regeln gelten für eine Objekt, und letztlich weisen Sie Amazon S3 an, widersprüchliche Aktionen auszuführen. In diesem Fall lässt Amazon S3 das Objekt 14 Tage nach der Erstellung ablaufen. Das Objekt wird entfernt, deshalb kommt die Übergangsaktion nicht ins Spiel.

Beispiel 6: Spezifikation einer Lebenszykluskonfigurationsregel für einen Bucket mit Versioning

Angenommen, Sie haben einen Versioning-fähigen Bucket, d. h. Sie haben für jedes Objekt eine aktuelle Version und keine oder mehr nicht aktuelle Versionen. Sie wollen, dass ein Verlauf von einem Jahr beibehalten wird. Anschließend sollen die nicht aktuellen Versionen gelöscht werden. Mehr über S3-Versioning erfahren Sie unter Objekt-Versioning (p. 127).

Außerdem wollen Sie Speicherkosten sparen, indem Sie nicht aktuelle Versionen 30 Tage, nachdem sie nicht aktuell geworden sind, in S3 Glacier verschieben (vorausgesetzt, es handelt sich um kalte Daten, für die Sie keinen Speicherzugriff in Echtzeit benötigen). Darüber hinaus erwarten Sie, dass der häufige Zugriff auf die aktuellen Versionen 90 Tage nach der Erstellung abläuft, Sie könnten also entscheiden, diese Objekte in die S3 Standard-IA-Speicherklasse zu überführen.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Filter>
           <Prefix></Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>90</Days>
           <StorageClass>STANDARD_IA</StorageClass>
        </Transition>
        <NoncurrentVersionTransition>
            <NoncurrentDays>30</NoncurrentDays>
            <StorageClass>S3 Glacier</StorageClass>
        </NoncurrentVersionTransition>
       <NoncurrentVersionExpiration>
            <NoncurrentDays>365</NoncurrentDays>
       </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Beispiel 7: Löschen abgelaufener Löschmarkierungen für Objekte

Ein Bucket mit Versioning enthält eine aktuelle Version und null oder mehr nicht aktuelle Versionen für jedes Objekt. Beachten Sie beim Löschen eines Objekts Folgendes:

- Wenn Sie keine Versions-ID in Ihrer Löschanforderung angeben, fügt Amazon S3 eine Löschmarkierung hinzu, statt das Objekt zu löschen. Das aktuelle Version wird nicht aktuell, und die Löschmarkierung wird zur aktuellen Version.
- Wenn Sie eine Versions-ID in Ihrer Löschanforderung angeben, löscht Amazon S3 die Objektversion permanent (es wird keine Löschmarkierung erstellt).
- Eine Löschmarkierung mit null nicht aktuellen Versionen wird als Löschmarkierung für das abgelaufene Objekt bezeichnet.

Dieses Beispiel zeigt ein Szenario, das Löschmarkierungen für abgelaufene Objekte in Ihrem Bucket erstellen kann, und demonstriert, wie Sie Amazon S3 mit einer S3-Lebenszyklus-Konfiguration anweisen können, die Löschmarkierungen für abgelaufene Objekte zu löschen.

Angenommen, Sie schreiben eine Lebenszyklusrichtlinie, die die `NoncurrentVersionExpiration`-Aktion angibt, um die nicht aktuellen Versionen 30 Tage, nachdem sie nicht aktuell geworden sind, zu löschen, wie nachfolgend gezeigt.

```
<LifecycleConfiguration>
    <Rule>
        ...
        <NoncurrentVersionExpiration>
            <NoncurrentDays>30</NoncurrentDays>
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Die Aktion `NoncurrentVersionExpiration` gilt nicht für die aktuellen Objektversionen. Sie entfernt nur nicht aktuelle Versionen.

Für aktuelle Objektversionen haben Sie die folgenden Optionen, ihre Lebensdauer zu verwalten, abhängig davon, ob die aktuellen Objektversionen einen definieren Lebenszyklus haben:

- Aktuelle Objektversionen folgen einem gut definierten Lebenszyklus.

  In diesem Fall können Sie eine Lebenszyklusrichtlinie mit der `Expiration`-Aktion verwenden, um Amazon S3 anzuweisen, aktuelle Versionen zu entfernen, wie im folgenden Beispiel gezeigt.

  ```
  <LifecycleConfiguration>
      <Rule>
          ...
          <Expiration>
             <Days>60</Days>
          </Expiration>
          <NoncurrentVersionExpiration>
              <NoncurrentDays>30</NoncurrentDays>
          </NoncurrentVersionExpiration>
      </Rule>
  </LifecycleConfiguration>
  ```

  Amazon S3 entfernt aktuelle Versionen 60 Tage, nachdem sie erstellt wurden, indem eine Löschmarkierung für jede der aktuellen Objektversionen hinzugefügt wird. Damit wird die aktuelle Version wird nicht aktuell, und die Löschmarkierung wird zur aktuellen Version. Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514).

Note

Durch diese Regel wird automatisch eine `ExpiredObjectDeleteMarker`-Bereinigung in einem versionierten Bucket durchgeführt, sodass die Notwendigkeit, ein `ExpiredObjectDeleteMarker`-Tag einzufügen, entfällt.

Die `NoncurrentVersionExpiration`-Aktion in derselben Lebenszykluskonfiguration entfernt nicht aktuelle Objekte 30 Tage, nachdem sie nicht aktuell wurden. Somit werden in diesem Beispiel 90 Tage nach der Objekterstellung alle Objektversionen dauerhaft entfernt. Sie erhalten Löschmarkierungen für abgelaufene Obekte, aber Amazon S3 erkennt und entfernt die Löschmarkierungen für abgelaufene Objekte für Sie.

- Aktuelle Objektversionen folgen keinem gut definierten Lebenszyklus.

  In diesem Fall müssen Sie die Objekte möglicherweise manuell entfernen, wenn Sie sie nicht mehr brauchen, und eine Löschmarkierungen mit einer oder mehreren nicht aktuellen Versionen erstellen. Wenn die Lebenszykluskonfiguration mit der `NoncurrentVersionExpiration`-Aktion alle nicht aktuellen Versionen löscht, haben Sie jetzt Löschmarkierungen für abgelaufene Objekte.

  Speziell für dieses Szenario stellt die Amazon S3-Lebenszykluskonfiguration die `Expiration`-Aktion bereit, mit der Sie Amazon S3 auffordern können, die Löschmarkierungen für abgelaufene Objekte zu entfernen.

```
<LifecycleConfiguration>
    <Rule>
       <ID>Rule 1</ID>
        <Filter>
          <Prefix>logs/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Expiration>
           <ExpiredObjectDeleteMarker>true</ExpiredObjectDeleteMarker>
        </Expiration>
        <NoncurrentVersionExpiration>
            <NoncurrentDays>30</NoncurrentDays>
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Wenn Sie das Element `ExpiredObjectDeleteMarker` in der Aktion `Expiration` auf „true“ festlegen, weisen Sie Amazon S3 an, Löschmarkierungen für abgelaufene Objekte zu entfernen.

Note

Bei Angabe der `ExpiredObjectDeleteMarker`-Lebenszyklusaktion kann die Regel keinen Tag-basierten Filter angeben.

Beispiel 8: Lebenszykluskonfigurationsregel für das Abbrechen mehrteiliger Uploads

Sie können die API für mehrteilige Uploads verwenden, um große Objekte in Teilen hochzuladen. Weitere Informationen über mehrteilige Uploads finden Sie unter Überblick Multipart Upload (p. 199).

Mit Hilfe der S3-Lebenszyklus-Konfiguration können Sie Amazon S3 dazu anweisen, unvollständige mehrteilige Uploads abzubrechen (identifiziert durch das Schlüsselnamenpräfix in der Regel), die nicht innerhalb einer bestimmten Anzahl von Tagen nach der Initiierung abgeschlossen wurden. Wenn Amazon S3 einen mehrteiligen Upload abbricht, werden alle diesem mehrteiligen Upload zugeordneten Teile gelöscht. Damit wird sichergestellt, dass Sie keine unvollständigen mehrteiligen Uploads mit Teilen haben, die in Amazon S3 gespeichert sind, sodass Sie keine Speicherkosten für diese Teile zahlen müssen.

Note

Bei Angabe der `AbortIncompleteMultipartUpload`-Lebenszyklusaktion kann die Regel keinen Tag-basierten Filter angeben.

Das folgende Beispiel zeigt eine S3-Lebenszyklus-Konfiguration, die eine Regel mit der Aktion `AbortIncompleteMultipartUpload` spezifiziert. Diese Aktion fordert Amazon S3 auf, unvollständige mehrteilige Uploads sieben Tage nach der Initiierung abzubrechen.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Filter>
           <Prefix>SomeKeyPrefix/</Prefix>
        </Filter>
        <Status>rule-status</Status>
        <AbortIncompleteMultipartUpload>
          <DaysAfterInitiation>7</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

Festlegen der Lebenszykluskonfiguration für einen Bucket

Themen

In diesem Abschnitt wird die programmgesteuerte Einrichtung der S3-Lebenszyklus-Konfiguration für einen Bucket über AWS SDKs, die Amazon S3-Konsole oder die AWS CLI beschrieben. Beachten Sie Folgendes:

- Wenn Sie eine S3-Lebenszyklus-Konfiguration zu einem Bucket hinzufügen, tritt eine gewisse Verzögerung ein, bis eine neue oder aktualisierte Lebenszykluskonfiguration vollständig an alle Amazon S3-Systeme weitergegeben wird. Rechnen Sie mit einer Verzögerung von einigen Minuten, bis die Konfiguration vollständig wirksam ist. Diese Verzögerung kann auch auftreten, wenn Sie eine S3-Lebenszyklus-Konfiguration löschen.
- Wenn Sie eine Lebenszyklusregel deaktivieren oder löschen, stellt Amazon S3 nach einer kurzen Verzögerung die Planung neuer Objekte zur Löschung oder Übertragung ein. Die Planung aller bereits geplanten Objekte wird aufgehoben und sie werden nicht gelöscht oder überführt.
- Wenn Sie einem Bucket eine Lebenszykluskonfiguration hinzufügen, gelten die Konfigurationsregeln für vorhandene Objekte und für Objekte, die Sie später hinzufügen. Zum Beispiel: Wenn Sie heute eine Lebenszykluskonfiguration mit einer Ablaufaktion hinzufügen, die dazu führt, dass Objekte mit einem bestimmten Präfix 30 Tage nach ihrer Erstellung ablaufen, verschiebt Amazon S3 alle bestehenden Objekte, die mehr als 30 Tage alt sind, in die Löschwarteschlange.
- Zwischen dem Zeitpunkt der Erfüllung der Lebenszykluskonfigurationsregeln und der dadurch ausgelösten Aktion kann eine Verzögerung eintreten. Buchhaltungsänderungen werden jedoch sofort nach der Erfüllung der Lebenszykluskonfigurationsregel durchgeführt, auch wenn die Maßnahme noch nicht durchgeführt wurde. Ein Beispiel: Nach dem Ablaufzeitpunkt des Objekts wird die Speicherung

nicht mehr berechnet, selbst wenn das Objekt nicht sofort gelöscht wird. Ein weiteres Beispiel: Sie zahlen Amazon S3 Glacier-Speichergebühren sofort nachdem die Objekt-Übertragungszeit abgelaufen ist, selbst wenn das Objekt nicht sofort zur Speicherklasse S3 Glacier übertragen wird. Lebenszyklusübergänge in die S3 Intelligent-Tiering-Speicherklasse sind eine Ausnahme. Änderungen in der Fakturierung treten erst auf, nachdem das Objekt in die S3 Intelligent-Tiering-Speicherklasse überführt wurde.

Weitere Informationen zur S3-Lebenszyklus-Konfiguration finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Verwaltung des Lebenszyklus eines Objekts mithilfe der Amazon S3-Konsole

Sie können S3-Lebenszyklus-Regeln für einen Bucket mithilfe der Amazon S3-Konsole angeben.

Anleitungen zum Einrichten von S3-Lebenszyklus-Regeln über die AWS Management Console finden Sie unter Wie erstelle ich eine Lebenszyklusrichtlinie für einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Einrichten von Lebenszykluskonfigurationen mithilfe der AWS CLI

Zur Verwaltung der S3-Lebenszyklus-Konfigurationen können Sie die folgenden AWS CLI-Befehle verwenden:

- `put-bucket-lifecycle-configuration`
- `get-bucket-lifecycle-configuration`
- `delete-bucket-lifecycle`

Weitere Informationen zum Einrichten der AWS CLI finden Sie unter Einrichten der AWS-CLI (p. 808).

Die Amazon S3-Lebenszykluskonfiguration ist eine XML-Datei. Wenn Sie jedoch die AWS CLI verwenden, können Sie die XML nicht angeben. Sie müssen stattdessen das JSON-Format angeben. Nachfolgend finden Sie Beispiele für XML-Lebenszykluskonfigurationen und entsprechenden JSON-Code, den Sie in einem AWS CLI-Befehl angeben können:

- Betrachten Sie das folgende Beispiel einer S3-Lebenszyklus-Konfiguration:

```
<LifecycleConfiguration>
    <Rule>
        <ID>ExampleRule</ID>
        <Filter>
           <Prefix>documents/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>365</Days>
           <StorageClass>GLACIER</StorageClass>
        </Transition>
        <Expiration>
             <Days>3650</Days>
        </Expiration>
    </Rule>
</LifecycleConfiguration>
```

Das JSON-Äquivalent wird gezeigt.

```
{
```

```
    "Rules": [
        {
            "Filter": {
                "Prefix": "documents/"
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 365,
                    "StorageClass": "GLACIER"
                }
            ],
            "Expiration": {
                "Days": 3650
            },
            "ID": "ExampleRule"
        }
    ]
}
```

- Betrachten Sie das folgende Beispiel einer S3-Lebenszyklus-Konfiguration:

```
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Rule>
        <ID>id-1</ID>
        <Expiration>
            <Days>1</Days>
        </Expiration>
        <Filter>
            <And>
                <Prefix>myprefix</Prefix>
                <Tag>
                    <Key>mytagkey1</Key>
                    <Value>mytagvalue1</Value>
                </Tag>
                <Tag>
                    <Key>mytagkey2</Key>
                    <Value>mytagvalue2</Value>
                </Tag>
            </And>
        </Filter>
        <Status>Enabled</Status>
    </Rule>
</LifecycleConfiguration>
```

Das JSON-Äquivalent wird gezeigt.

```
{
    "Rules": [
        {
            "ID": "id-1",
            "Filter": {
                "And": {
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
                        }
                    ]
```

```
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 1
            }
        }
    ]
}
```

Sie können den Befehl `put-bucket-lifecycle-configuration` wie folgt testen.

So testen Sie die Konfiguration

1. Speichern Sie die JSON-Lebenszykluskonfiguration in einer Datei (`lifecycle.json`).
2. Führen Sie den folgenden AWS CLI-Befehl aus, um die Lebenszykluskonfiguration auf Ihrem Bucket einzurichten.

```
$ aws s3api put-bucket-lifecycle-configuration  \
--bucket bucketname  \
--lifecycle-configuration file://lifecycle.json
```

3. Um dies zu überprüfen, rufen Sie die S3-Lebenszyklus-Konfiguration über den AWS CLI-Befehl `get-bucket-lifecycle-configuration` wie folgt ab:

```
$ aws s3api get-bucket-lifecycle-configuration  \
--bucket bucketname
```

4. Um die S3-Lebenszyklus-Konfiguration zu löschen, verwenden Sie wie folgt den AWS CLI-Befehl `delete-bucket-lifecycle`.

```
aws s3api delete-bucket-lifecycle \
--bucket bucketname
```

Verwalten der Objektlebenszyklen unter Verwendung des AWS SDK for Java

Sie können mit AWS SDK for Java die S3-Lebenszyklus-Konfiguration eines Buckets verwalten. Weitere Informationen zur Verwaltung einer S3-Lebenszyklus-Konfiguration finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

> **Note**
>
> Wenn Sie einem Bucket eine S3-Lebenszyklus-Konfiguration hinzufügen, ersetzt Amazon S3 die aktuelle Lebenszykluskonfiguration des Buckets, sofern vorhanden. Um eine Lebenszykluskonfiguration zu aktualisieren, rufen Sie sie ab, nehmen die gewünschten Änderungen vor und fügen die geänderte Konfiguration dann dem Bucket hinzu.

Example

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK for Java die Lebenszykluskonfiguration eines Buckets hinzufügen, aktualisieren und löschen. Das Beispiel erledigt Folgendes:

- Fügt eine Lebenszykluskonfiguration zu einem Bucket hinzu.
- Ruft die Lebenszykluskonfiguration und Updates durch Hinzufügen einer weiteren Regel ab.

- Fügt dem Bucket die geänderte Lebenszykluskonfiguration hinzu. Amazon S3 ersetzt die vorhandene Lebenszykluskonfiguration.
- Ruft die Konfiguration erneut ab und überprüft, ob sie die richtige Anzahl von Regeln enthält, indem die Anzahl der Regeln ausgegeben wird.
- Löscht die Lebenszykluskonfiguration und überprüft, ob sie gelöscht wurde, indem versucht wird, sie erneut abzurufen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketLifecycleConfiguration;
import com.amazonaws.services.s3.model.BucketLifecycleConfiguration.Transition;
import com.amazonaws.services.s3.model.StorageClass;
import com.amazonaws.services.s3.model.Tag;
import com.amazonaws.services.s3.model.lifecycle.LifecycleAndOperator;
import com.amazonaws.services.s3.model.lifecycle.LifecycleFilter;
import com.amazonaws.services.s3.model.lifecycle.LifecyclePrefixPredicate;
import com.amazonaws.services.s3.model.lifecycle.LifecycleTagPredicate;

import java.io.IOException;
import java.util.Arrays;

public class LifecycleConfiguration {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        // Create a rule to archive objects with the "glacierobjects/" prefix to Glacier
 immediately.
        BucketLifecycleConfiguration.Rule rule1 = new BucketLifecycleConfiguration.Rule()
                .withId("Archive immediately rule")
                .withFilter(new LifecycleFilter(new
 LifecyclePrefixPredicate("glacierobjects/")))
                .addTransition(new
 Transition().withDays(0).withStorageClass(StorageClass.Glacier))
                .withStatus(BucketLifecycleConfiguration.ENABLED);

        // Create a rule to transition objects to the Standard-Infrequent Access storage
 class
        // after 30 days, then to Glacier after 365 days. Amazon S3 will delete the objects
 after 3650 days.
        // The rule applies to all objects with the tag "archive" set to "true".
        BucketLifecycleConfiguration.Rule rule2 = new BucketLifecycleConfiguration.Rule()
                .withId("Archive and then delete rule")
                .withFilter(new LifecycleFilter(new LifecycleTagPredicate(new
 Tag("archive", "true"))))
                .addTransition(new
 Transition().withDays(30).withStorageClass(StorageClass.StandardInfrequentAccess))
                .addTransition(new
 Transition().withDays(365).withStorageClass(StorageClass.Glacier))
                .withExpirationInDays(3650)
                .withStatus(BucketLifecycleConfiguration.ENABLED);

        // Add the rules to a new BucketLifecycleConfiguration.
```

```
        BucketLifecycleConfiguration configuration = new BucketLifecycleConfiguration()
                .withRules(Arrays.asList(rule1, rule2));

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Save the configuration.
            s3Client.setBucketLifecycleConfiguration(bucketName, configuration);

            // Retrieve the configuration.
            configuration = s3Client.getBucketLifecycleConfiguration(bucketName);

            // Add a new rule with both a prefix predicate and a tag predicate.
            configuration.getRules().add(new
 BucketLifecycleConfiguration.Rule().withId("NewRule")
                    .withFilter(new LifecycleFilter(new LifecycleAndOperator(
                            Arrays.asList(new LifecyclePrefixPredicate("YearlyDocuments/"),
                                    new LifecycleTagPredicate(new Tag("expire_after",
 "ten_years"))))))
                    .withExpirationInDays(3650)
                    .withStatus(BucketLifecycleConfiguration.ENABLED));

            // Save the configuration.
            s3Client.setBucketLifecycleConfiguration(bucketName, configuration);

            // Retrieve the configuration.
            configuration = s3Client.getBucketLifecycleConfiguration(bucketName);

            // Verify that the configuration now has three rules.
            configuration = s3Client.getBucketLifecycleConfiguration(bucketName);
            System.out.println("Expected # of rules = 3; found: " +
 configuration.getRules().size());

            // Delete the configuration.
            s3Client.deleteBucketLifecycleConfiguration(bucketName);

            // Verify that the configuration has been deleted by attempting to retrieve it.
            configuration = s3Client.getBucketLifecycleConfiguration(bucketName);
            String s = (configuration == null) ? "No configuration found." : "Configuration
 found.";
            System.out.println(s);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwaltung des Lebenszyklus von Objekten mittels dem AWS SDK für .NET

Sie können mit AWS SDK für .NET die S3-Lebenszyklus-Konfiguration für einen Bucket verwalten. Weitere Informationen zur Verwaltung einer Lebenszykluskonfiguration finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Note

Wenn Sie eine Lebenszykluskonfiguration hinzufügen, ersetzt Amazon S3 die vorhandene Konfiguration für den angegebenen Bucket. Um eine Konfiguration zu aktualisieren, müssen Sie zuerst die Lebenszykluskonfiguration abrufen, die Änderungen vornehmen und dann die geänderte Lebenszykluskonfiguration dem Bucket hinzufügen.

Example .NET-Code-Beispiel

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK für .NET die Lebenszykluskonfiguration eines Buckets hinzufügen, aktualisieren und löschen. Das Codebeispiel führt die folgenden Aufgaben durch:

- Fügt eine Lebenszykluskonfiguration zu einem Bucket hinzu.
- Ruft die Lebenszykluskonfiguration und Updates durch Hinzufügen einer weiteren Regel ab.
- Fügt dem Bucket die geänderte Lebenszykluskonfiguration hinzu. Amazon S3 ersetzt die vorhandene Lebenszykluskonfiguration.
- Ruft die Konfiguration erneut ab und überprüft sie durch Ausgabe der Anzahl von Regeln in der Konfiguration.
- Löscht die Lebenszykluskonfiguration und überprüft den Löschvorgang.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class LifecycleTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            AddUpdateDeleteLifecycleConfigAsync().Wait();
        }

        private static async Task AddUpdateDeleteLifecycleConfigAsync()
        {
            try
            {
                var lifeCycleConfiguration = new LifecycleConfiguration()
                {
                    Rules = new List<LifecycleRule>
                        {
                            new LifecycleRule
                            {
                                 Id = "Archive immediately rule",
                                 Filter = new LifecycleFilter()
                                 {
                                     LifecycleFilterPredicate = new
 LifecyclePrefixPredicate()
```

```
                                    {
                                        Prefix = "glacierobjects/"
                                    }
                                },
                                Status = LifecycleRuleStatus.Enabled,
                                Transitions = new List<LifecycleTransition>
                                {
                                     new LifecycleTransition
                                     {
                                          Days = 0,
                                          StorageClass = S3StorageClass.Glacier
                                     }
                                 },
                            },
                            new LifecycleRule
                            {
                                 Id = "Archive and then delete rule",
                                  Filter = new LifecycleFilter()
                                 {
                                     LifecycleFilterPredicate = new
LifecyclePrefixPredicate()
                                     {
                                         Prefix = "projectdocs/"
                                     }
                                 },
                                 Status = LifecycleRuleStatus.Enabled,
                                 Transitions = new List<LifecycleTransition>
                                 {
                                      new LifecycleTransition
                                      {
                                           Days = 30,
                                           StorageClass =
S3StorageClass.StandardInfrequentAccess
                                      },
                                      new LifecycleTransition
                                      {
                                        Days = 365,
                                        StorageClass = S3StorageClass.Glacier
                                      }
                                 },
                                 Expiration = new LifecycleRuleExpiration()
                                 {
                                       Days = 3650
                                 }
                            }
                        }
                };

                // Add the configuration to the bucket.
                await AddExampleLifecycleConfigAsync(client, lifeCycleConfiguration);

                // Retrieve an existing configuration.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);

                // Add a new rule.
                lifeCycleConfiguration.Rules.Add(new LifecycleRule
                {
                    Id = "NewRule",
                    Filter = new LifecycleFilter()
                    {
                        LifecycleFilterPredicate = new LifecyclePrefixPredicate()
                        {
                            Prefix = "YearlyDocuments/"
                        }
                    },
                    Expiration = new LifecycleRuleExpiration()
```

```
                    {
                        Days = 3650
                    }
                });

                // Add the configuration to the bucket.
                await AddExampleLifecycleConfigAsync(client, lifeCycleConfiguration);

                // Verify that there are now three rules.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);
                Console.WriteLine("Expected # of rulest=3; found:{0}",
lifeCycleConfiguration.Rules.Count);

                // Delete the configuration.
                await RemoveLifecycleConfigAsync(client);

                // Retrieve a nonexistent configuration.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);

            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing an
object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        static async Task AddExampleLifecycleConfigAsync(IAmazonS3 client,
LifecycleConfiguration configuration)
        {

            PutLifecycleConfigurationRequest request = new PutLifecycleConfigurationRequest
            {
                BucketName = bucketName,
                Configuration = configuration
            };
            var response = await client.PutLifecycleConfigurationAsync(request);
        }

        static async Task<LifecycleConfiguration> RetrieveLifecycleConfigAsync(IAmazonS3
client)
        {
            GetLifecycleConfigurationRequest request = new GetLifecycleConfigurationRequest
            {
                BucketName = bucketName
            };
            var response = await client.GetLifecycleConfigurationAsync(request);
            var configuration = response.Configuration;
            return configuration;
        }

        static async Task RemoveLifecycleConfigAsync(IAmazonS3 client)
        {
            DeleteLifecycleConfigurationRequest request = new
DeleteLifecycleConfigurationRequest
            {
                BucketName = bucketName
            };
            await client.DeleteLifecycleConfigurationAsync(request);
        }
    }
```

```
}
```

Verwaltung des Lebenszyklus von Objekten mittels dem AWS SDK für Ruby

Sie können AWS SDK für Ruby verwenden, um die S3-Lebenszyklus-Konfiguration für einen Bucket mittels der Klasse AWS::S3::BucketLifecycleConfiguration zu verwalten. Weitere Informationen zur Verwendung von AWS SDK für Ruby mit Amazon S3 finden Sie unter Verwenden von AWS SDK für Ruby – Version 3 (p. 813). Für weitere Informationen zur Verwaltung einer Lebenszykluskonfiguration vgl. Verwaltung des Objektlebenszyklus (p. 139).

Verwalten des Lebenszyklus eines Objekts mit der REST-API

Sie können die AWS Management Console verwenden, um eine S3-Lebenszyklus-Konfiguration für Ihren Bucket zu erstellen. Falls in Ihrer Anwendung erforderlich, können Sie auch direkt REST-Anfragen senden. Die folgenden Abschnitte im Amazon Simple Storage Service API Reference beschreiben die REST-API im Zusammenhang mit der S3-Lebenszyklus-Konfiguration.

- PUT Bucket-Lebenszyklus
- GET Bucket-Lebenszyklus
- DELETE Bucket-Lebenszyklus

Cross-Origin Resource Sharing (CORS)

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Mit CORS-Unterstützung können Sie umfassende clientseitige Webanwendungen mit Amazon S3 erstellen und selektiven ursprungsübergreifenden Zugriff auf Ihre Amazon S3-Ressourcen zulassen.

Dieser Abschnitt bietet eine Übersicht über CORS. Die Unterthemen beschreiben, wie Sie CORS über die Amazon S3-Konsole oder programmgesteuert mit der Amazon S3-REST-API und den AWS-SDKs aktivieren können.

Themen

Cross-Origin Resource Sharing: Szenarien in Anwendungsfällen

Es folgen typische Beispielszenarien für den Einsatz von CORS.

Szenario 1

Angenommen, Sie hosten eine Website in einem Amazon S3-Bucket mit dem Namen `website`, wie in Hosten einer statischen Website auf Amazon S3 (p. 602) beschrieben. Ihre Benutzer laden den Website-Endpunkt:

```
http://website.s3-website.us-east-1.amazonaws.com
```

Jetzt können Sie auf den in diesem Bucket gespeicherten Webseiten JavaScript verwenden, um authentifizierte GET- und PUT-Anforderungen für diesen Bucket selbst zu erstellen, indem für den Bucket der Amazon S3-API-Endpunkt `website.s3.us-east-1.amazonaws.com` verwendet wird. Ein Browser würde normalerweise verhindern, dass JavaScript diese Anfragen erstellt, aber mit CORS können Sie Ihren Bucket konfigurieren, um explizit ursprungsübergreifende Anfragen von `website.s3-website.us-east-1.amazonaws.com` zu aktivieren.

Szenario 2

Angenommen, Sie möchten eine Web-Schriftart aus Ihrem S3-Bucket hosten. Auch hier erfordern Browser eine CORS-Prüfung (auch als Preflight-Check bezeichnet) für das Laden von Web-Schriftarten. Sie würden den Bucket, der die Web-Schriftart hostet, deshalb so konfigurieren, dass jeder Ursprung diese Anfragen machen kann.

Wie konfiguriere ich CORS für meinen Bucket?

Um Ihren Bucket so zu konfigurieren, dass er ursprungsübergreifende Anfragen zulässt, erstellen Sie eine CORS-Konfiguration. Dabei handelt es sich um ein XML-Dokument mit Regeln, die die Ursprünge identifizieren, die den Zugriff auf Ihren Bucket zulassen, die Operationen (HTTP-Methoden), die die einzelnen Ursprünge unterstützen, sowie weitere operationsspezifische Informationen.

Sie können der Konfiguration bis zu 100 Regeln hinzufügen. Sie fügen das XML-Dokument als `cors`-Unterressource zum Bucket hinzu – entweder programmgesteuert oder über die Amazon S3-Konsole. Weitere Informationen finden Sie unter Cross-Origin Resource Sharing (CORS) aktivieren (p. 177).

Statt über einen Amazon S3-Website-Endpunkt auf eine Website zuzugreifen, können Sie Ihre eigene Domäne verwenden, z. B. `example1.com`, um Ihren Inhalt bereitzustellen. Weitere Informationen zur Verwendung Ihrer eigenen Domäne finden Sie unter Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten, bei Route 53 registrierten Domäne (p. 631). Die folgende Beispielkonfiguration für `cors` umfasst drei Regeln, die als `CORSRule`-Elemente angegeben sind:

- Die erste Regel gestattet ursprungsübergreifende PUT-, POST- und DELETE-Anfragen vom Ursprung `http://www.example1.com`. Die Regel gestattet auch alle Header in einer Preflight-OPTIONS-Anfrage durch den `Access-Control-Request-Headers`-Header. Als Antwort auf Preflight-OPTIONS-Anfragen gibt Amazon S3 angeforderte Header zurück.
- Die zweite Regel gestattet dieselben ursprungsübergreifenden Anfragen wie die erste Regel, aber sie bezieht sich auf einen anderen Ursprung, `http://www.example2.com`.
- Die dritte Regel gestattet ursprungsübergreifende GET-Anfragen von allen Ursprüngen. Das Platzhalterzeichen `*` bezieht sich auf alle Ursprünge.

```
<CORSConfiguration>
 <CORSRule>
   <AllowedOrigin>http://www.example1.com</AllowedOrigin>

   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>

   <AllowedHeader>*</AllowedHeader>
 </CORSRule>
 <CORSRule>
   <AllowedOrigin>http://www.example2.com</AllowedOrigin>

   <AllowedMethod>PUT</AllowedMethod>
```

```
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>

    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
  </CORSRule>
</CORSConfiguration>
```

Die CORS-Konfiguration unterstützt auch optionale Konfigurationsparameter, wie in der folgenden CORS-Konfiguration gezeigt. In diesem Beispiel gestattet die folgende CORS-Konfiguration ursprungsübergreifende PUT-, POST- und DELETE-Anfragen vom Ursprung `http://www.example.com`.

```
<CORSConfiguration>
 <CORSRule>
   <AllowedOrigin>http://www.example.com</AllowedOrigin>
   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>
   <AllowedHeader>*</AllowedHeader>
  <MaxAgeSeconds>3000</MaxAgeSeconds>
  <ExposeHeader>x-amz-server-side-encryption</
ExposeHeader>
  <ExposeHeader>x-amz-request-id</
ExposeHeader>
  <ExposeHeader>x-amz-id-2</ExposeHeader>
 </CORSRule>
</CORSConfiguration>
```

Das `CORSRule`-Element in der obigen Konfiguration beinhaltet die folgenden optionalen Elemente:

- `MaxAgeSeconds` – Gibt den Zeitraum in Sekunden an (in diesem Beispiel 3000), über den der Browser eine Amazon S3-Antwort auf eine Preflight-OPTIONS-Anforderung für die angegebene Ressource zwischenspeichert. Durch die Zwischenspeicherung der Antwort muss der Browser keine Preflight-Anfragen an Amazon S3 senden, wenn die ursprüngliche Anfrage wiederholt werden soll.
- `ExposeHeader` – Identifiziert die Antwort-Header (in diesem Beispiel `x-amz-server-side-encryption`, `x-amz-request-id` und `x-amz-id-2`), auf die Kunden von ihren Anwendungen aus zugreifen können (z. B. von einem JavaScript-`XMLHttpRequest`-Objekt aus).

AllowedMethod-Element

In der CORS-Konfiguration können Sie die folgenden Werte für das `AllowedMethod`-Element angeben.

- GET
- PUT
- POST
- DELETE
- HEAD

AllowedOrigin-Element

Im `AllowedOrigin`-Element geben Sie die Ursprünge an, über die Sie domänenübergreifende Anfragen erlauben möchten, z. B. `http://www.example.com`. Die Ursprungszeichenfolge darf nur ein `*`-

Platzhalterzeichen enthalten, wie beispielsweise `http://*.example.com`. Optional können Sie `*` als Ursprung angeben, sodass alle Ursprünge ursprungsübergreifende Anfragen senden dürfen. Sie können auch `https` angeben, um nur sichere Ursprünge zuzulassen.

AllowedHeader-Element

Das `AllowedHeader`-Element gibt an, welche Header in einer Preflight-Anfrage durch den `Access-Control-Request-Headers`-Header erlaubt sind. Jeder Header-Name im `Access-Control-Request-Headers`-Header muss mit einem entsprechenden Eintrag in der Regel übereinstimmen. Amazon S3 sendet nur die zulässigen angeforderten Header in einer Antwort. Eine Liste mit Beispielen für Header, die in Anforderungen an Amazon S3 verwendet werden können, finden Sie unter Gängige Anforderungs-Header im Leitfaden Amazon Simple Storage Service API Reference.

Jede AllowedHeader-Zeichenkette in der Regel darf höchstens einmal das Wildcard-Zeichen * enthalten. Beispielsweise aktiviert `<AllowedHeader>x-amz-*</AllowedHeader>` alle für Amazon spezifischen Header.

ExposeHeader-Element

Jedes `ExposeHeader`-Element identifiziert die Antwort-Header, auf die Kunden von ihren Anwendungen aus zugreifen können sollen (z. B. von einem JavaScript `XMLHttpRequest`-Objekt). Eine Liste mit gängigen Amazon S3-Antwort-Headern finden Sie unter Gängige Response-Header im Leitfaden Amazon Simple Storage Service API Reference.

MaxAgeSeconds-Element

Das `MaxAgeSeconds`-Element gibt die Zeit in Sekunden an, wie lange Ihr Browser die Antwort auf eine Preflight-Anfrage zwischenspeichern kann, wie nach der Ressource, der HTTP-Methode und dem Ursprung identifiziert.

Wie wertet Amazon S3 die CORS-Konfiguration für einen Bucket aus?

Wenn Amazon S3 eine Preflight-Anforderung von einem Browser erhält, wertet es die CORS-Konfiguration für den Bucket aus und verwendet die erste `CORSRule`-Regel, die mit der eingehenden Browser-Anforderung übereinstimmt, um eine ursprungsübergreifende Anforderung zuzulassen. Für die Übereinstimmung mit einer Regel müssen die folgenden Bedingungen erfüllt sei:

- Der `Origin`-Header der Anfrage muss mit einem `AllowedOrigin`-Element übereinstimmen.
- Die Anfragemethode (z. B. GET oder PUT) oder der `Access-Control-Request-Method`-Header bei einer Preflight-`OPTIONS`-Anfrage muss eines der `AllowedMethod`-Elemente sein.
- Jeder im `Access-Control-Request-Headers`-Header der Preflight-Anfrage muss mit einem `AllowedHeader`-Element übereinstimmen.

> **Note**
>
> Alle ACLs und Richtlinien gelten weiterhin, wenn Sie CORS für den Bucket aktivieren.

Cross-Origin Resource Sharing (CORS) aktivieren

Sie aktivieren Cross-Origin Resource Sharing (CORS), indem Sie unter Verwendung der AWS Management Console, des REST-API oder der AWS SDKs eine CORS-Konfiguration für Ihren Bucket einstellen.

Themen

Aktivieren von Cross-Origin Resource Sharing (CORS) mit der AWS Management Console

Sie können die AWS Management Console verwenden, um eine CORS-Konfiguration für Ihren Bucket zu erstellen. Weitere Informationen finden Sie unter Wie lasse ich ein domänenübergreifendes Ressourcen-Sharing mit CORS zu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Aktivieren von Cross-Origin Resource Sharing (CORS) mit dem AWS SDK for Java

Cross-Origin Resource Sharing (CORS) unter Verwendung von AWS SDK for Java aktivieren Weitere Informationen über CORS finden Sie unter Cross-Origin Resource Sharing (CORS) (p. 174).

Example

Das folgende Beispiel:

- Erstellt eine CORS-Konfiguration und legt die Konfiguration für einen Bucket fest.
- Ruft die Konfiguration ab und ändert sie durch Hinzufügen einer Regel ab
- Fügt die abgeänderte Konfiguration dem Bucket hinzu
- Löscht die Konfiguration

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketCrossOriginConfiguration;
import com.amazonaws.services.s3.model.CORSRule;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class CORS {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        // Create two CORS rules.
        List<CORSRule.AllowedMethods> rule1AM = new ArrayList<CORSRule.AllowedMethods>();
        rule1AM.add(CORSRule.AllowedMethods.PUT);
```

```
        rule1AM.add(CORSRule.AllowedMethods.POST);
        rule1AM.add(CORSRule.AllowedMethods.DELETE);
        CORSRule rule1 = new CORSRule().withId("CORSRule1").withAllowedMethods(rule1AM)
                .withAllowedOrigins(Arrays.asList("http://*.example.com"));

        List<CORSRule.AllowedMethods> rule2AM = new ArrayList<CORSRule.AllowedMethods>();
        rule2AM.add(CORSRule.AllowedMethods.GET);
        CORSRule rule2 = new CORSRule().withId("CORSRule2").withAllowedMethods(rule2AM)
                .withAllowedOrigins(Arrays.asList("*")).withMaxAgeSeconds(3000)
                .withExposedHeaders(Arrays.asList("x-amz-server-side-encryption"));

        List<CORSRule> rules = new ArrayList<CORSRule>();
        rules.add(rule1);
        rules.add(rule2);

        // Add the rules to a new CORS configuration.
        BucketCrossOriginConfiguration configuration = new
BucketCrossOriginConfiguration();
        configuration.setRules(rules);

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Add the configuration to the bucket.
            s3Client.setBucketCrossOriginConfiguration(bucketName, configuration);

            // Retrieve and display the configuration.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            printCORSConfiguration(configuration);

            // Add another new rule.
            List<CORSRule.AllowedMethods> rule3AM = new
ArrayList<CORSRule.AllowedMethods>();
            rule3AM.add(CORSRule.AllowedMethods.HEAD);
            CORSRule rule3 = new CORSRule().withId("CORSRule3").withAllowedMethods(rule3AM)
                    .withAllowedOrigins(Arrays.asList("http://www.example.com"));

            rules = configuration.getRules();
            rules.add(rule3);
            configuration.setRules(rules);
            s3Client.setBucketCrossOriginConfiguration(bucketName, configuration);

            // Verify that the new rule was added by checking the number of rules in the
configuration.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            System.out.println("Expected # of rules = 3, found " +
configuration.getRules().size());

            // Delete the configuration.
            s3Client.deleteBucketCrossOriginConfiguration(bucketName);
            System.out.println("Removed CORS configuration.");

            // Retrieve and display the configuration to verify that it was
            // successfully deleted.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            printCORSConfiguration(configuration);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
```

```
            e.printStackTrace();
        }
    }

    private static void printCORSConfiguration(BucketCrossOriginConfiguration
 configuration) {
        if (configuration == null) {
            System.out.println("Configuration is null.");
        } else {
            System.out.println("Configuration has " + configuration.getRules().size() + "
 rules\n");

            for (CORSRule rule : configuration.getRules()) {
                System.out.println("Rule ID: " + rule.getId());
                System.out.println("MaxAgeSeconds: " + rule.getMaxAgeSeconds());
                System.out.println("AllowedMethod: " + rule.getAllowedMethods());
                System.out.println("AllowedOrigins: " + rule.getAllowedOrigins());
                System.out.println("AllowedHeaders: " + rule.getAllowedHeaders());
                System.out.println("ExposeHeader: " + rule.getExposedHeaders());
                System.out.println();
            }
        }
    }
}
```

Aktivieren von Cross-Origin Resource Sharing (CORS) mit dem AWS SDK für .NET

Sie können das Cross-Origin Resource Sharing (CORS) für einen Bucket mit AWS SDK für .NET verwalten. Weitere Informationen über CORS finden Sie unter Cross-Origin Resource Sharing (CORS) (p. 174).

Example

Der folgende C#-Code:

- Erstellt eine CORS-Konfiguration und legt die Konfiguration für einen Bucket fest.
- Ruft die Konfiguration ab und ändert sie durch Hinzufügen einer Regel ab
- Fügt die abgeänderte Konfiguration dem Bucket hinzu
- Löscht die Konfiguration

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CORSTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
```

```
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CORSConfigTestAsync().Wait();
        }
        private static async Task CORSConfigTestAsync()
        {
            try
            {
                // Create a new configuration request and add two rules
                CORSConfiguration configuration = new CORSConfiguration
                {
                    Rules = new System.Collections.Generic.List<CORSRule>
                        {
                          new CORSRule
                          {
                            Id = "CORSRule1",
                            AllowedMethods = new List<string> {"PUT", "POST", "DELETE"},
                            AllowedOrigins = new List<string> {"http://*.example.com"}
                          },
                          new CORSRule
                          {
                            Id = "CORSRule2",
                            AllowedMethods = new List<string> {"GET"},
                            AllowedOrigins = new List<string> {"*"},
                            MaxAgeSeconds = 3000,
                            ExposeHeaders = new List<string> {"x-amz-server-side-
encryption"}
                          }
                        }
                };

                // Add the configuration to the bucket.
                await PutCORSConfigurationAsync(configuration);

                // Retrieve an existing configuration.
                configuration = await RetrieveCORSConfigurationAsync();

                // Add a new rule.
                configuration.Rules.Add(new CORSRule
                {
                    Id = "CORSRule3",
                    AllowedMethods = new List<string> { "HEAD" },
                    AllowedOrigins = new List<string> { "http://www.example.com" }
                });

                // Add the configuration to the bucket.
                await PutCORSConfigurationAsync(configuration);

                // Verify that there are now three rules.
                configuration = await RetrieveCORSConfigurationAsync();
                Console.WriteLine();
                Console.WriteLine("Expected # of rulest=3; found:{0}",
 configuration.Rules.Count);
                Console.WriteLine();
                Console.WriteLine("Pause before configuration delete. To continue, click
 Enter...");
                Console.ReadKey();

                // Delete the configuration.
                await DeleteCORSConfigurationAsync();

                // Retrieve a nonexistent configuration.
                configuration = await RetrieveCORSConfigurationAsync();
```

```
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        static async Task PutCORSConfigurationAsync(CORSConfiguration configuration)
        {

            PutCORSConfigurationRequest request = new PutCORSConfigurationRequest
            {
                BucketName = bucketName,
                Configuration = configuration
            };

            var response = await s3Client.PutCORSConfigurationAsync(request);
        }

        static async Task<CORSConfiguration> RetrieveCORSConfigurationAsync()
        {
            GetCORSConfigurationRequest request = new GetCORSConfigurationRequest
            {
                BucketName = bucketName

            };
            var response = await s3Client.GetCORSConfigurationAsync(request);
            var configuration = response.Configuration;
            PrintCORSRules(configuration);
            return configuration;
        }

        static async Task DeleteCORSConfigurationAsync()
        {
            DeleteCORSConfigurationRequest request = new DeleteCORSConfigurationRequest
            {
                BucketName = bucketName
            };
            await s3Client.DeleteCORSConfigurationAsync(request);
        }

        static void PrintCORSRules(CORSConfiguration configuration)
        {
            Console.WriteLine();

            if (configuration == null)
            {
                Console.WriteLine("\nConfiguration is null");
                return;
            }

            Console.WriteLine("Configuration has {0} rules:", configuration.Rules.Count);
            foreach (CORSRule rule in configuration.Rules)
            {
                Console.WriteLine("Rule ID: {0}", rule.Id);
                Console.WriteLine("MaxAgeSeconds: {0}", rule.MaxAgeSeconds);
                Console.WriteLine("AllowedMethod: {0}", string.Join(", ",
rule.AllowedMethods.ToArray()));
                Console.WriteLine("AllowedOrigins: {0}", string.Join(", ",
rule.AllowedOrigins.ToArray()));
```

```
                Console.WriteLine("AllowedHeaders: {0}", string.Join(", ",
 rule.AllowedHeaders.ToArray()));
                Console.WriteLine("ExposeHeader: {0}", string.Join(", ",
 rule.ExposeHeaders.ToArray()));
            }
        }
    }
}
```

Cross-Origin Resource Sharing (CORS) unter Verwendung der REST API aktivieren

Sie können die AWS Management Console verwenden, um eine CORS-Konfiguration für Ihren Bucket zu erstellen. Falls in Ihrer Anwendung erforderlich, können Sie auch direkt REST-Anfragen senden. Die folgenden Abschnitte in der Amazon Simple Storage Service API Reference beschreiben die REST-API-Aktionen im Hinblick auf die CORS-Konfiguration:

- PUT Bucket cors
- GET Bucket cors
- DELETE Bucket cors
- OPTIONS-Objekt

Fehlerbehebung bei CORS-Problemen

Wenn Sie ein unerwartetes Verhalten feststellen, wenn Sie auf Buckets zugreifen, die mit CORS-Konfiguration eingerichtet wurden, führen Sie die folgenden Schritte zur Fehlerbehebung durch:

1. Vergewissern Sie sich, dass die CORS-Konfiguration für den Bucket eingerichtet ist.

   Weitere Informationen finden Sie unter Ändern von Bucket-Berechtigungen im Konsolenbenutzerhandbuch für Amazon Simple Storage Service. Falls die CORS-Konfiguration eingerichtet wurde, zeigt die Konsole den Link Edit CORS Configuration (CORS-Konfiguration bearbeiten) im Bereich Permissions (Berechtigungen) des Buckets Properties (Eigenschaften) an.
2. Erfassen Sie die vollständige Anfrage und antworten Sie mit einem Tool Ihrer Wahl. Wie folgt, muss es für jede von Amazon S3 erhaltene Anfrage eine CORS-Regel geben, die mit den Daten in Ihrer Anfrage übereinstimmt:
   a. Stellen Sie sicher, dass die Anfrage den Origin-Header besitzt.

      Falls der Header fehlt, verarbeitet Amazon S3 die Anfrage nicht als ursprungsübergreifende Anfrage und sendet in der Antwort keine CORS-Antwort-Header.
   b. Stellen Sie sicher, dass der Origin-Header in Ihrer Anfrage mit mindestens einem der `AllowedOrigin`-Elemente in der angegebenen `CORSRule` übereinstimmt.

      Das Schema, der Host und die Port-Werte im Origin-Anfrageheader müssen mit den `AllowedOrigin`-Elementen in der `CORSRule` übereinstimmen. Wenn Sie beispielsweise die `CORSRule` so eingerichtet haben, dass der Ursprung `http://www.example.com` zulässig ist, stimmen die Ursprünge `https://www.example.com` und `http://www.example.com:80` in Ihrer Anfrage nicht mit dem in Ihrer Konfiguration erlaubten Ursprung überein.
   c. Stellen Sie sicher, dass die Methode in Ihrer Anfrage (oder in einer Preflight-Anfrage die in `Access-Control-Request-Method` angegebene Methode) eines der `AllowedMethod`-Elemente in derselben `CORSRule` ist.
   d. Wenn bei einer Preflight-Anfrage die anfrage einen `Access-Control-Request-Headers`-Header enthält, überprüfen Sie, ob die `CORSRule` die `AllowedHeader`-Einträge für jeden Wert im `Access-Control-Request-Headers`-Header enthält.

Operationen für Objekte

Amazon S3 ermöglicht Ihnen, Objekte zu speichern, abzurufen und zu löschen. Sie können ein gesamtes Objekt oder einen Teil davon abrufen. Wenn Sie für Ihren Bucket S3-Versioning aktiviert haben, können Sie eine bestimmte Version des Objekts abrufen. Sie können auch Ihrem Objekt zugeordnete Subressourcen abrufen und gegebenenfalls aktualisieren. Sie können eine Kopie Ihres vorhandenen Objekts anlegen. Abhängig von der Objektgröße gelten die folgenden Aspekte für das Hochladen und Kopieren:

- Hochladen von Objekten— In einer einzelnen Operation können Sie Objekte bis zu einer Größe von 5 GB hochladen. Für Objekte mit mehr als 5 GB müssen Sie die API für mehrteilige Uploads verwenden.

  Um Objekte mit jeweils bis zu 5 TB hochzuladen, verwenden Sie die API für mehrteilige Uploads. Weitere Informationen finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).
- Kopieren von Objekten— Die Kopieroperation erzeugt eine Kopie eines Objekts, das bereits in Amazon S3 gespeichert ist.

  Sie können eine Kopie Ihres Objekts mit einer Größe von bis zu 5 GB in einer einzigen atomaren Operation hochladen. Für das Kopieren von Objekten mit mehr als 5 GB müssen Sie jedoch die API für mehrteilige Uploads verwenden. Weitere Informationen finden Sie unter Objekte kopieren (p. 233).

Sie können die REST-API (siehe Senden von Anforderungen unter Verwendung der REST-API (p. 46)) verwenden, um mit Objekten zu arbeiten, oder eine der folgenden AWS-SDK-Bibliotheken verwenden:

- AWS SDK for Java
- AWS SDK für .NET
- AWS SDK für PHP

Diese Bibliotheken bieten eine hohe Abstraktion, sodass die Arbeit mit Objekten ganz einfach wird. Falls in Ihrer Anwendung erforderlich, können Sie die REST API jedoch auch direkt verwenden.

Objekte abrufen

Themen

Sie können Objekte direkt von Amazon S3 abrufen. Beim Abrufen eines Objekts haben Sie die folgenden Optionen:

- Abrufen eines ganzen Objekts— Eine einzige GET-Operation kann das ganze in Amazon S3 gespeicherte Objekt zurückgeben.
- Abrufen eines Objekts in Teilen —— Mit dem `Range`-HTTP-Header in einer GET-Anforderung können Sie einen spezifischen Bytebereich eines in Amazon S3 gespeicherten Objekts abrufen.

  Sie setzen den Abruf weiterer Objektteile fort, wenn Ihre Anwendung dazu bereit ist. Dieser fortsetzbare Download ist praktisch, wenn Sie nur Teile Ihrer Objektdaten benötigen. Außerdem ist er praktisch, wenn eine schlechte Netzwerkverbindung vorliegt und Sie auf Ausfälle reagieren müssen.

> **Note**
>
> Amazon S3 unterstützt nicht den Abruf mehrere Datenbereiche pro GET-Anforderung.

Wenn Sie ein Objekt abrufen, werden seine Metadaten in den Antwort-Headern zurückgegeben. Es gibt Situationen, in denen Sie bestimmte Antwortheader-Werte überschreiben können, die in einer GET-Antwort zurückgegeben werden. Beispielsweise könnten Sie den Antwort-Header-Wert `Content-Disposition` in Ihrer GET-Anforderung überschreiben. Die REST GET Object API (siehe GET Object) gestattet Ihnen, Abfragezeichenfolgenparameter in Ihrer GET-Anforderung anzugeben, um diese Werte zu überschreiben.

Die AWS SDKs for Java, .NET und PHP stellen zudem die erforderlichen Objekte bereit, mit denen Sie Werte für diese Antwort-Header in Ihrer GET-Anforderung angeben können.

Wenn Sie Objekte abrufen, die mit serverseitiger Verschlüsselung (SSE) verschlüsselt gespeichert sind, müssen Sie geeignete Anforderungs-Header bereitstellen. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung (p. 290).

Verwandte Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Abrufen eines Objekts mit AWS SDK for Java

Wenn Sie ein Objekt über AWS SDK for Java herunterladen, gibt Amazon S3 alle Metadaten des Objekts sowie einen Eingabe-Stream, aus dem Sie den Inhalt des Objekts herauslesen können, zurück.

Zum Abrufen eines Objekts verfahren Sie wie folgt:

- Führen Sie die `AmazonS3Client.getObject()`-Methode aus, indem Sie in der Anforderung den Bucket-Namen und Objektschlüssel bereitstellen.
- Führen Sie eine der `S3Object`-Instance-Methoden aus, um den Eingabe-Stream zu verarbeiten.

> **Note**
>
> Ihre Netzwerkverbindung bleibt offen, bis Sie alle Daten lesen oder den Eingabe-Stream schließen. Wir empfehlen, dass Sie den Inhalt des Streams möglichst schnell lesen.

Es folgen einige Variationen, die Sie möglicherweise verwenden:

- Statt das ganze Objekt zu lesen, können Sie auch nur einen Teil der Objektdaten lesen, indem Sie in der Anforderung den gewünschten Bytebereich angeben.
- Sie können optional die Antwort-Header-Werte überschreiben (siehe Objekte abrufen (p. 184)). Dazu verwenden Sie ein `ResponseHeaderOverrides`-Objekt und legen die entsprechende Anforderungseigenschaft fest. Sie können mit dieser Funktion beispielsweise angeben, dass das Objekt in eine Datei mit einem anderen Dateinamen als dem Objektschlüsselnamen heruntergeladen werden soll.

Das folgende Beispiel ruft ein Objekt auf drei Arten aus einem Amazon S3-Bucket ab: zuerst als vollständiges Objekt, dann als ein Bereich von Bytes aus dem Objekt und schließlich als vollständiges Objekt mit überschriebenen Antwortheader-Werten. Weitere Informationen zum Abrufen von Objekten aus Amazon S3 finden Sie unter GET Object. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GetObjectRequest;
import com.amazonaws.services.s3.model.ResponseHeaderOverrides;
import com.amazonaws.services.s3.model.S3Object;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

public class GetObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String key = "*** Object key ***";

        S3Object fullObject = null, objectPortion = null, headerOverrideObject = null;
        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Get an object and print its contents.
            System.out.println("Downloading an object");
            fullObject = s3Client.getObject(new GetObjectRequest(bucketName, key));
            System.out.println("Content-Type: " +
 fullObject.getObjectMetadata().getContentType());
            System.out.println("Content: ");
            displayTextInputStream(fullObject.getObjectContent());

            // Get a range of bytes from an object and print the bytes.
            GetObjectRequest rangeObjectRequest = new GetObjectRequest(bucketName, key)
                    .withRange(0, 9);
            objectPortion = s3Client.getObject(rangeObjectRequest);
            System.out.println("Printing bytes retrieved.");
            displayTextInputStream(objectPortion.getObjectContent());

            // Get an entire object, overriding the specified response headers, and print
 the object's content.
            ResponseHeaderOverrides headerOverrides = new ResponseHeaderOverrides()
                    .withCacheControl("No-cache")
                    .withContentDisposition("attachment; filename=example.txt");
            GetObjectRequest getObjectRequestHeaderOverride = new
 GetObjectRequest(bucketName, key)
                    .withResponseHeaders(headerOverrides);
            headerOverrideObject = s3Client.getObject(getObjectRequestHeaderOverride);
            displayTextInputStream(headerOverrideObject.getObjectContent());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        } finally {
            // To ensure that the network connection doesn't remain open, close any open
 input streams.
            if (fullObject != null) {
                fullObject.close();
```

```
            }
            if (objectPortion != null) {
                objectPortion.close();
            }
            if (headerOverrideObject != null) {
                headerOverrideObject.close();
            }
        }
    }

    private static void displayTextInputStream(InputStream input) throws IOException {
        // Read the text input stream one line at a time and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line = null;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

Abrufen eines Objekts mit AWS SDK für .NET

Wenn Sie ein Objekt herunterladen, erhalten Sie alle Metadaten des Objekts, sowie einen Stream, aus dem Sie den Inhalt herauslesen können. Sie sollten den Inhalt des Streams so schnell wie möglich lesen, weil die Daten direkt von Amazon S3 gestreamt werden, und Ihre Netzwerkverbindung offen bleibt, bis Sie alle Daten gelesen haben oder den Eingabestream schließen. Sie verfahren zum Abrufen eines Objekts wie folgt:

- Führen Sie die `getObject`-Methode aus, indem Sie in der Anforderung den Bucket-Namen und Objektschlüssel bereitstellen.
- Führen Sie eine der `GetObjectResponse`-Methoden aus, um den Stream zu verarbeiten.

Es folgen einige Variationen, die Sie möglicherweise verwenden:

- Statt das ganze Objekt zu lesen, können Sie auch einen Teil der Objektdaten lesen, indem Sie in der Abfrage den Bytebereich angeben, wie im folgenden C#-Codebeispiel gezeigt:

  Example

  ```
  GetObjectRequest request = new GetObjectRequest
  {
      BucketName = bucketName,
      Key = keyName,
      ByteRange = new ByteRange(0, 10)
  };
  ```

- Wenn Sie ein Objekt abrufen, können Sie optional die Antwort-Header-Werte überschreiben (siehe Objekte abrufen (p. 184)). Dazu verwenden Sie das `ResponseHeaderOverrides`-Objekt und legen die entsprechende Anforderungseigenschaft fest. Das folgende C#-Codebeispiel veranschaulicht, wie dazu vorgegangen wird. Sie können mit dieser Funktion beispielsweise angeben, dass das Objekt in eine Datei mit einem anderen Dateinamen als dem Objektschlüsselnamen heruntergeladen werden soll.

  Example

  ```
  GetObjectRequest request = new GetObjectRequest
  {
      BucketName = bucketName,
  ```

```
    Key = keyName
};

ResponseHeaderOverrides responseHeaders = new ResponseHeaderOverrides();
responseHeaders.CacheControl = "No-cache";
responseHeaders.ContentDisposition = "attachment; filename=testing.txt";

request.ResponseHeaderOverrides = responseHeaders;
```

Example

Das folgende C#-Codebeispiel ruft ein Objekt aus einem Amazon S3-Bucket ab. Aus der Antwort liest das Beispiel die Objektdaten unter Verwendung der `GetObjectResponse.ResponseStream`-Eigenschaft. Das Beispiel zeigt auch, wie Sie die `GetObjectResponse.Metadata`-Sammlung verwenden können, um Objektmetadaten zu lesen. Wenn das von Ihnen abgerufene Objekt die `x-amz-meta-title`-Metadaten enthält, gibt der Code den Metadatenwert aus.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class GetObjectTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            ReadObjectDataAsync().Wait();
        }

        static async Task ReadObjectDataAsync()
        {
            string responseBody = "";
            try
            {
                GetObjectRequest request = new GetObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };
                using (GetObjectResponse response = await client.GetObjectAsync(request))
                using (Stream responseStream = response.ResponseStream)
                using (StreamReader reader = new StreamReader(responseStream))
                {
                    string title = response.Metadata["x-amz-meta-title"]; // Assume you
 have "title" as medata added to the object.
                    string contentType = response.Headers["Content-Type"];
                    Console.WriteLine("Object metadata, Title: {0}", title);
                    Console.WriteLine("Content type: {0}", contentType);
```

```
                    responseBody = reader.ReadToEnd(); // Now you process the response
 body.
                }
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing an
 object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Abrufen eines Objekts mit AWS SDK für PHP

In diesem Thema erfahren Sie, wie Sie mit einer Klasse aus AWS SDK für PHP ein Amazon S3-Objekt abrufen. Sie können ein gesamtes Objekt oder einen Bytebereich des Objekts abrufen. Es wird angenommen, dass Sie bereits die Anweisungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) befolgt haben und AWS SDK für PHP ordnungsgemäß installiert ist.

Wenn Sie ein Objekt abrufen, können Sie optional die Antwort-Header-Werte überschreiben. Dazu fügen Sie die Antwortschlüssel `ResponseContentType`, `ResponseContentLanguage`, `ResponseContentDisposition`, `ResponseCacheControl` und `ResponseExpires` der `getObject()`-Methode hinzu, wie im folgenden PHP-Codebeispiel veranschaulicht:

Example

```
$result = $s3->getObject([
    'Bucket'                     => $bucket,
    'Key'                        => $keyname,
    'ResponseContentType'        => 'text/plain',
    'ResponseContentLanguage'    => 'en-US',
    'ResponseContentDisposition' => 'attachment; filename=testing.txt',
    'ResponseCacheControl'       => 'No-cache',
    'ResponseExpires'            => gmdate(DATE_RFC2822, time() + 3600),
]);
```

Weitere Informationen über das Abrufen von Objekten finden Sie unter Objekte abrufen (p. 184).

Das folgende PHP-Codebeispiel ruft ein Objekt ab und zeigt den Objektinhalt im Browser an. Das Beispiel veranschaulicht, wie Sie die `getObject()`-Methode verwenden. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

try {
```

```
    // Get the object.
    $result = $s3->getObject([
        'Bucket' => $bucket,
        'Key'    => $keyname
    ]);

    // Display the object in the browser.
    header("Content-Type: {$result['ContentType']}");
    echo $result['Body'];
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Verwandte Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Abrufen eines Objekts mit der REST API

Sie können das AWS SDK verwenden, um die Objektschlüssel von einem Bucket abzurufen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Sie können eine GET-Anforderung senden, um Objektschlüssel abzurufen. Weitere Informationen über das Anforderungs- und Antwort-Format finden sie unter Get Object.

Ein Objekt mit anderen teilen

Standardmäßig werden alle Objekte als privat eingestuft. Nur der Objekteigentümer hat die Berechtigung, auf diese Objekte zuzugreifen. Der Objekteigentümer kann die Objekte jedoch optional mit anderen teilen, indem er eine vorsignierte URL erstellt und seine eigenen Sicherheitsanmeldeinformationen verwendet, um eine temporäre Berechtigung für den Download der Objekte zu erteilen.

Wenn Sie eine vorsignierte URL für Ihr Objekt erstellen, müssen Sie Ihre Sicherheitsanmeldeinformationen bereitstellen, einen Bucket-Namen, einen Objektschlüssel, eine HTTP-Methode (GET für den Download des Objekts) und ein Datum und eine Uhrzeit für den Ablauf angeben. Vorsignierte URLs sind nur für die angegebene Dauer gültig.

Jeder, der die vorsignierte URL erhält, kann dann auf das Objekt zugreifen. Wenn Sie beispielsweise ein Video in Ihrem Bucket haben und sowohl der Bucket als auch das Objekt privat sind, können Sie das Video mit anderen teilen, indem Sie eine vorsignierte URL generieren.

Note

- Alle Benutzer mit gültigen Sicherheitsanmeldeinformationen können vorsignierte URLs erstellen. Um erfolgreich auf ein Objekt zugreifen zu können, muss jedoch die vorsignierte URL von jemandem erstellt werden, der die Berechtigung für die Operation besitzt, auf der die vorsignierte URL basiert.
- Die Anmeldeinformationen, die Sie zum Erstellen einer vorsignierten URL verwenden können, beinhalten:
  - IAM-Instance-Profil: Bis zu 6 Stunden gültig.
  - AWS Security Token Service: Bis zu 36 Stunden gültig, wenn mit permanenten Anmeldeinformationen signiert (beispielsweise Anmeldeinformationen des AWS-Konto-Root-Benutzers oder eines IAM-Benutzers).
  - IAM-Benutzer: Bis zu 7 Tage gültig, bei Verwendung von AWS-Signaturversion 4.

    Um eine vorsignierte URL zu erstellen, die bis zu 7 Tage gültig ist, geben Sie zunächst IAM Benutzeranmeldeinformationen (den Zugriffsschlüssel und den geheimen Zugriffsschlüssel)

für das SDK an, das Sie verwenden. Erzeugen Sie dann eine vorsignierte URL mit AWS Signature Version 4.

- Wenn Sie eine vorsignierte URL mit einem temporären Token erstellt haben, verfällt die URL mit Ablauf des Token, auch wenn die URL mit einer späteren Ablaufzeit erstellt wurde.

Sie können eine vorsignierte URL mithilfe von REST API, AWS Command Line Interface und AWS SDK for Java, .NET, Ruby, PHP, Node.js und Python programmgesteuert generieren.

Themen

Generieren einer vorsignierten Objekt-URL mit dem AWS Explorer für Visual Studio

Wenn Sie Visual Studio verwenden, können Sie auch den AWS Explorer für Visual Studio verwenden, um eine vorsignierte URL für ein Objekt zu erstellen, ohne Code schreiben zu müssen. Jeder mit dieser URL kann das Objekt herunterladen. Weitere Informationen finden Sie unter Verwenden von Amazon S3 über AWS Explorer.

Anweisungen zur Installation des AWS-Explorers finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).

Generieren einer vorsignierten Objekt-URL mit AWS SDK for Java

Example

Das folgende Beispiel erstellt eine vorsignierte URL, die Sie für andere bereitstellen können, so dass sie ein Objekt aus einem S3-Bucket abrufen können. Weitere Informationen finden Sie unter Ein Objekt mit anderen teilen (p. 190).

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.HttpMethod;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GeneratePresignedUrlRequest;

import java.io.IOException;
import java.net.URL;

public class GeneratePresignedURL {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String objectKey = "*** Object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
```

```
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Set the presigned URL to expire after one hour.
            java.util.Date expiration = new java.util.Date();
            long expTimeMillis = expiration.getTime();
            expTimeMillis += 1000 * 60 * 60;
            expiration.setTime(expTimeMillis);

            // Generate the presigned URL.
            System.out.println("Generating pre-signed URL.");
            GeneratePresignedUrlRequest generatePresignedUrlRequest =
                    new GeneratePresignedUrlRequest(bucketName, objectKey)
                            .withMethod(HttpMethod.GET)
                            .withExpiration(expiration);
            URL url = s3Client.generatePresignedUrl(generatePresignedUrlRequest);

            System.out.println("Pre-Signed URL: " + url.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Generieren einer vorsignierten Objekt-URL mit AWS SDK für .NET

Example

Das folgende Beispiel generiert eine vorsignierte URL, die Sie für andere bereitstellen können, so dass sie ein Objekt abrufen können. Weitere Informationen finden Sie unter Ein Objekt mit anderen teilen (p. 190).

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;

namespace Amazon.DocSamples.S3
{
    class GenPresignedURLTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string objectKey = "*** object key ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            string urlString = GeneratePreSignedURL();
        }
        static string GeneratePreSignedURL()
        {
```

```
            string urlString = "";
            try
            {
                GetPreSignedUrlRequest request1 = new GetPreSignedUrlRequest
                {
                    BucketName = bucketName,
                    Key = objectKey,
                    Expires = DateTime.Now.AddMinutes(5)
                };
                urlString = s3Client.GetPreSignedURL(request1);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            return urlString;
        }
    }
}
```

Objekte hochladen

Abhängig von der Größe der Daten, die Sich hochladen, bietet Amazon S3 die folgenden Optionen:

- Hochladen von Objekten in einer einzigen Operation —– Mit einer einzigen PUT- Operation können Sie Objekte bis zu einer Größe von 5 GB hochladen.

  Weitere Informationen finden Sie unter Hochladen eines Objekts in einer einzigen Operation (p. 194).
- Hochladen von Objekten in Teilen— Um große Objekte mit jeweils bis zu 5 TB hochzuladen, verwenden Sie die API für mehrteilige Uploads.

  Die API für mehrteilige Uploads ist darauf ausgelegt, die Upload-Leistung für größere Objekte zu verbessern. Sie können diese Objekte in Teilen hochladen. Diese Objektteile können unabhängig, in jeder beliebigen Reihenfolge und parallel hochgeladen werden. Sie können einen mehrteiligen Upload verwenden, um Objekte mit einer Größe von 5 MB bis 5 TB hochzuladen. Weitere Informationen finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).

Wir empfehlen, zuerst das mehrteilige Hochladen wie folgt zu verwenden:

- Wenn Sie große Objekte über ein Netzwerk mit stabiler hoher Bandbreite hochladen, verwenden Sie einen mehrteiligen Upload, um die Nutzung der verfügbaren Bandbreite zu maximieren, indem Sie Objektteile parallel hochladen und von einer Multi-Threading-Leistung profitieren.
- Wenn Sie über ein instabiles Netzwerk hochladen, verwenden Sie den mehrteiligen Upload, um die Stabilität gegenüber Netzwerkfehlern zu verbessern, indem Neustarts der Uploads vermieden werden. Wenn Sie mehrteilige Uploads verwenden, müssen Sie nur die Teile erneut hochladen, die während des Uploads unterbrochen wurden. Sie müssen nicht das gesamte Objekt von Anfang an neu hochladen.

Weitere Informationen über mehrteilige Uploads finden Sie unter Überblick Multipart Upload (p. 199).

Themen

- Hochladen von Objekten unter Verwendung vorsignierter URLs (p. 228)

Beim Hochladen eines Objekts können Sie optional anfordern, dass Amazon S3 Ihr Objekt verschlüsselt, bevor es dieses auf einen Datenträger schreibt, und es wieder entschlüsselt, wenn Sie es herunterladen. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung (p. 290).

Verwandte Themen

Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Hochladen eines Objekts in einer einzigen Operation

Themen

- Hochladen eines Objekts mit AWS SDK for Java (p. 194)
- Hochladen eines Objekts mit AWS SDK für .NET (p. 195)
- Hochladen eines Objekts mit AWS SDK für PHP (p. 196)
- Hochladen eines Objekts mit AWS SDK für Ruby (p. 197)
- Hochladen eines Objekts mit der REST API (p. 198)
- Hochladen eines Objekts mit der CLI (p. 198)

Sie können das AWS SDK verwenden, um Objekte hochzuladen. Das SDK bietet Ihnen Wrapper-Bibliotheken für den einfachen Upload von Daten an. Falls in Ihrer Anwendung erforderlich, können Sie jedoch auch direkt die REST API in Ihrer Anwendung verwenden. Sie können auch die AWS Command Line Interface verwenden.

Hochladen eines Objekts mit AWS SDK for Java

Example

Das folgende Beispiel erstellt zwei Objekte. Das erste Objekt besitzt eine Textzeichenfolge als Daten, und das zweite Objekt ist eine Datei. Das Beispiel erstellt das erste Objekt durch Angabe des Bucket-Namen, des Objektschlüssels und von Textdaten direkt in einem Aufruf an `AmazonS3Client.putObject()`. Das Beispiel erstellt das zweite Objekt mittels einer `PutObjectRequest`, die den Bucket-Namen, den Objektschlüssel und den Dateipfad angibt. Die `PutObjectRequest` gibt zudem den `ContentType`-Header und Titelmetadaten an.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.PutObjectRequest;

import java.io.File;
import java.io.IOException;

public class UploadObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
```

```
        String stringObjKeyName = "*** String object key name ***";
        String fileObjKeyName = "*** File object key name ***";
        String fileName = "*** Path to file to upload ***";

        try {
            //This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .build();

            // Upload a text string as a new object.
            s3Client.putObject(bucketName, stringObjKeyName, "Uploaded String Object");

            // Upload a file as a new object with ContentType and title specified.
            PutObjectRequest request = new PutObjectRequest(bucketName, fileObjKeyName, new
 File(fileName));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("plain/text");
            metadata.addUserMetadata("title", "someTitle");
            request.setMetadata(metadata);
            s3Client.putObject(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Hochladen eines Objekts mit AWS SDK für .NET

Example

Das folgende C#-Codebeispiel erstellt zwei Objekte mit zwei `PutObjectRequest`-Anforderungen:

- Die erste `PutObjectRequest`-Anforderung speichert als Beispielobjektdaten eine Textzeichenfolge. Sie gibt zudem Bucket- und Objektschlüsselnamen an.
- Die zweite `PutObjectRequest`-Anforderung lädt eine Datei durch Angabe des Dateinamens hoch. Diese Anforderung gibt außerdem den `ContentType`-Header und optionale Objektmetadaten (einen Titel) ein.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadObjectTest
    {
        private const string bucketName = "*** bucket name ***";
```

```
        // For simplicity the example creates two objects from the same file.
        // You specify key names for these objects.
        private const string keyName1 = "*** key name for first object created ***";
        private const string keyName2 = "*** key name for second object created ***";
        private const string filePath = @"*** file path ***";
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.EUWest1;

        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            WritingAnObjectAsync().Wait();
        }

        static async Task WritingAnObjectAsync()
        {
            try
            {
                // 1. Put object-specify only key name for the new object.
                var putRequest1 = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName1,
                    ContentBody = "sample text"
                };

                PutObjectResponse response1 = await client.PutObjectAsync(putRequest1);

                // 2. Put the object-set ContentType and add metadata.
                var putRequest2 = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName2,
                    FilePath = filePath,
                    ContentType = "text/plain"
                };
                putRequest2.Metadata.Add("x-amz-meta-title", "someTitle");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an object"
                        , e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Unknown encountered on server. Message:'{0}' when writing an object"
                    , e.Message);
            }
        }
    }
}
```

Hochladen eines Objekts mit AWS SDK für PHP

Dieses Thema führt Sie durch die Verwendung von AWS SDK für PHP-Klassen für das Hochladen eines Objekts mit einer Größe von bis zu 5 GB. Für größere Dateien müssen Sie eine API für einen mehrteiligen Upload verwenden. Weitere Informationen finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).

Dieser Abschnitt setzt voraus, dass Sie den Anweisungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und AWS SDK für PHP ordnungsgemäß installiert ist.

Example Erstellen eines Objekts in einem Amazon S3-Bucket, indem Daten hochgeladen werden

Das folgende PHP- Beispiel erstellt ein Objekt in einem spezifizierten Bucket, indem die Daten mit der `putObject()`-Methode hochgeladen werden. Weitere Informationen über die Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

try {
    // Upload data.
    $result = $s3->putObject([
        'Bucket' => $bucket,
        'Key'    => $keyname,
        'Body'   => 'Hello, world!',
        'ACL'    => 'public-read'
    ]);

    // Print the URL to the object.
    echo $result['ObjectURL'] . PHP_EOL;
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Verwandte Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Hochladen eines Objekts mit AWS SDK für Ruby

The AWS SDK für Ruby - Version 3 bietet zwei Möglichkeiten, ein Objekt zu Amazon S3 hochzuladen. Bei der ersten wird ein verwalteter Datei-Uploader verwendet, mit dem es ganz einfach ist, Dateien beliebiger Größe von einem Datenträger hochzuladen. So verwenden Sie die Methode des verwalteten Datei-Uploaders:

1. Erstellen Sie eine Instance der `Aws::S3::Resource`-Klasse.
2. Verweisen Sie per Bucket-Name und Schlüssel auf das Ziel-Objekt. Objekte befinden sich in einem Bucket und haben eindeutige Schlüssel, die jedes Objekt identifizieren.
3. Rufen Sie`#upload_file` für das Objekt auf.

Example

```
require 'aws-sdk-s3'

s3 = Aws::S3::Resource.new(region:'us-west-2')
obj = s3.bucket('bucket-name').object('key')
obj.upload_file('/path/to/source/file')
```

Die zweite Möglichkeit, mit der AWS SDK für Ruby - Version 3 ein Objekt hochladen kann, ist die `#put`-Methode von `Aws::S3::Object`. Das ist praktisch, wenn das Objekt eine Zeichenfolge oder ein I/O-Objekt ist, bei dem es sich nicht um eine Datei auf einem Datenträger handelt. So verwenden Sie diese Methode:

1. Erstellen Sie eine Instance der `Aws::S3::Resource`-Klasse.
2. Verweisen Sie per Bucket-Name und Schlüssel auf das Ziel-Objekt.
3. Rufen Sie `#put` auf und übergeben Sie die Zeichenfolge oder das I/O-Objekt.

Example

```
require 'aws-sdk-s3'

s3 = Aws::S3::Resource.new(region:'us-west-2')
obj = s3.bucket('bucket-name').object('key')

# I/O object
File.open('/path/to/source.file', 'rb') do |file|
  obj.put(body: file)
end
```

Hochladen eines Objekts mit der REST API

Sie können das AWS SDK verwenden, um ein Objekt hochzuladen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Sie können eine PUT-Anforderung senden, um Daten in einer einzigen Operation hochzuladen. Weitere Informationen finden Sie unter PUT Object.

Hochladen eines Objekts mit der CLI

Sie können das AWS SDK verwenden, um ein Objekt hochzuladen. Falls in Ihrer Anwendung erforderlich, können Sie AWS Command Line Interface-Anforderungen jedoch auch direkt senden. Sie können eine PUT-Anforderung senden, um Daten in einer einzigen Operation hochzuladen. Weitere Informationen finden Sie unter PUT Object im Handbuch AWS CLI Command Reference.

Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads

Mit dem mehrteiligen Upload können Sie ein einzelnes Objekt als Satz aus mehreren Teilen hochladen. Jeder Teil ist ein zusammenhängender Teil der Daten des Objekts. Sie können diese Objektteile unabhängig und in beliebiger Reihenfolge hochladen. Wenn die Übertragung eines Teils fehlschlägt, können Sie das Teil erneut übertragen, ohne dass dies Auswirkungen auf andere Teile hat. Nachdem alle Teile Ihres Objekts hochgeladen sind, baut Amazon S3 diese Teile zusammen und erstellt das Objekt. Wenn Ihre Objektgröße 100 MB erreicht, sollten Sie in der Regel mehrteilige Uploads verwenden, anstatt das Objekt in einem einzigen Vorgang hochzuladen.

Die Nutzung mehrteiliger Uploads bietet die folgenden Vorteile:

- Verbesserter Durchsatz - Sie können die Teile parallel hochladen, um den Durchsatz zu erhöhen.
- Schnelle Wiederherstellung bei Netzwerkproblemen - Die kleinere Teilegröße minimiert die Auswirkungen eines Neustarts eines fehlgeschlagenen Uploads aufgrund eines Netzwerkfehlers.
- Anhalten und Fortsetzen von Objekt-Uploads - Sie können Objektteile über die Zeit hochladen. Sobald Sie einen mehrteiligen Upload initiieren, gibt es kein Ablaufdatum. Sie müssen den Multipart-Upload explizit abschließen oder abbrechen.
- Starten Sie einen Upload, bevor Sie die endgültige Objektgröße kennen. Sie können ein Objekt hochladen, während Sie es noch erstellen.

Themen

Überblick Multipart Upload

Themen

Mit dem Multipart-Upload-API können Sie große Objekte in Teilen hochladen. Sie können dieses API verwenden, um neue große Objekte hochzuladen oder eine Kopie eines vorhandenen Objekts zu erstellen (siehe Operationen für Objekte (p. 184)).

Der mehrteilige Upload ist ein Prozess in drei Schritten: Sie beginnen den Upload, dann laden Sie die Objektteile hoch und wenn alle Teile hochgeladen sind, schließen Sie den mehrteiligen Upload ab. Nach dem vollständigen Eingang der Daten aus dem mehrteiligen Upload erstellt Amazon S3 das Objekt anhand der hochgeladenen Teile. Dann können Sie auf das Objekt wie auf jedes andere Objekt in Ihrem Bucket zugreifen.

Sie können alle laufenden mehrteiligen Uploads auflisten oder eine Liste der Teile anfordern, die Sie für einen bestimmten Multipart-Upload hochgeladen haben. Alle Operationen werden in diesem Abschnitt erklärt.

Initiieren des mehrteiligen Uploads

Wenn Sie eine Anforderung zum Initiieren eines mehrteiligen Uploads senden, gibt Amazon S3 eine Antwort mit einer Upload-ID zurück, als eindeutige Kennung für den Multipart-Upload. Sie müssen diese Upload-ID immer angeben, wenn Sie Teile hochladen, die Teile auflisten, einen Upload durchführen oder ihn abbrechen. Wenn Sie Metadaten bereitstellen möchten, die das hochzuladende Objekt beschreiben, müssen sie in der Anforderung auf Initiierung des mehrteiligen Uploads angegeben werden.

Teile hochladen

Beim Hochladen eines Teils müssen Sie zusätzlich zur Upload-ID eine Teilenummer angeben. Sie können jede Teilenummer zwischen 1 und 10.000 wählen. Die Teilenummer identifiziert eindeutig einen Teil und

seine Position im Objekt, das Sie hochladen. Die von Ihnen gewählte Teilenummer muss nicht fortlaufend sein (möglich sind z. B. 1, 5 und 14). Wenn Sie einen neuen Teil mit derselben Teilenummer hochladen wie bereits einmal zuvor, wird der früher hochgeladene Teil überschrieben. Wenn Sie einen Teil hochladen, gibt Amazon S3 einen ETag-Header in der Antwort zurück. Für jeden Teilupload müssen Sie die Teilenummer und den ETag-Wert notieren. Sie müssen diese Werte in die spätere Anforderung einschließen, um den mehrteiligen Upload abzuschließen.

> **Note**
>
> Nachdem Sie einen mehrteiligen Upload initiiert und einen oder mehrere Teile hochgeladen haben, müssen Sie den Multipart-Upload entweder abschließen oder abbrechen, damit keine Gebühren für die Speicherung der hochgeladenen Teile erhoben werden. Erst nach dem Abschluss oder Abbruch eines mehrteiligen Uploads gibt Amazon S3 den Speicher für die Teile frei und stoppt die Berechnung von Gebühren für die Speicherung der Teile.

Abschließen (oder Abbrechen) des mehrteiligen Uploads

Wenn Sie einen mehrteiligen Upload abschließen, erstellt Amazon S3 ein Objekt, indem die Teile in aufsteigender Reihenfolge auf Grundlage der Teilenummer verkettet werden. Wenn in einer Anforderung für die Initiierung eines mehrteiligen Uploads Objektmetadaten bereitgestellt wurden, verknüpft Amazon S3 diese Metadaten mit dem Objekt. Nach einer erfolgreich ausgeführten Abschlussanforderung sind die Teile nicht mehr vorhanden. Ihre Anforderung für den Abschluss eines mehrteiligen Uploads muss die Upload-ID und eine Liste der Nummern der Teile und der entsprechenden ETag-Werte enthalten. Die Amazon S3-Antwort enthält ein ETag, das die kombinierten Objektdaten auf eindeutige Weise identifiziert. Dieser ETag muss nicht notwendigerweise ein MD5-Hash der Objektdaten sein. Sie können einen mehrteiligen Upload auch abbrechen. Wenn Sie einen mehrteiligen Upload abgebrochen haben, können Sie keinen Teil mit dieser Upload-ID mehr hochladen. Die Speicherplatz aller Teile des abgebrochenen Multipart-Uploads wird dann freigegeben. Wenn Teile-Uploads im Gange waren, können sie auch nach dem Abbruch erfolgreich sein oder fehlschlagen. Um den verbrauchten Speicherplatz aller Teile freizugeben, dürfen Sie einen Multipart-Upload erst dann abbrechen, nachdem alle Teil-Uploads abgeschlossen wurden.

Auflistungen mehrteiliger Uploads

Sie können alle Teile eines bestimmten Multipart-Uploads oder alle laufenden mehrteiligen Uploads auflisten. Die Operation für die Teileauflistung gibt die Teileinformationen zurück, die Sie für einen bestimmten mehrteiligen Upload hochgeladen haben. Für jeden Abruf einer Teileauflistung gibt Amazon S3 die Teileinformationen für einen angegebenen mehrteiligen Upload bis zu maximal 1.000 Teilen zurück. Wenn im Multipart-Upload mehr als 1.000 Teile vorhanden sind, müssen Sie eine Reihe von Anforderungen auf Teileauflistung senden, um alle Teile abzurufen. Beachten Sie, dass die zurückgegebene Teileauflistung keine Teile enthält, die noch nicht vollständig hochgeladen wurden. Bei Verwendung der Operation Mehrteilige Uploads auflisten können Sie eine Liste aller mehrteiligen Uploads in Bearbeitung erhalten. Ein laufender mehrteiliger Upload ist ein Upload, den Sie initiiert haben, der aber noch nicht abgeschlossen ist oder abgebrochen wurde. Jeder Anforderung gibt bis zu 1000 mehrteilige Uploads zurück. Wenn mehr als 1.000 Multipart-Uploads vorhanden sind, müssen Sie zusätzliche Anforderungen senden, um die verbleibenden mehrteiligen Uploads abzurufen. Verwenden Sie die zurückgegebene Liste nur zur Überprüfung. Sie sollten das Ergebnis dieser Auflistung nicht verwenden, wenn Sie eine Anforderung für den Abschluss eines mehrteiligen Uploads senden. Halten Sie sich stattdessen an Ihre eigene Liste der Teilenummern, die Sie beim Hochladen von Teilen angegeben haben, und die diesbezüglichen ETag-Werte, die Amazon S3 zurückgegeben hat.

Gleichzeitige Multipart-Upload-Operationen

In einer verteilten Entwicklungsumgebung ist es für Ihre Anwendung möglich, mehrere Updates gleichzeitig für dasselbe Objekt zu initiieren. Ihre Anwendung kann möglicherweise mehrere Multipart-Uploads mit demselben Objektschlüssel initiieren. Für jeden dieser Uploads kann Ihre Anwendung Teile hochladen und eine Anforderung auf Abschluss des Uploads an Amazon S3 senden, um das Objekt zu erstellen. Wenn die Buckets die Versionierung aktiviert haben, wird beim Abschluss eines Multipart-Uploads immer eine neue Version erstellt. Bei Buckets, für die keine Versionierung aktiviert ist, kann es sein, dass zwischen den

Zeitpunkten der Initiierung bis zum Abschluss eines Multipart-Uploads eine andere Anforderung vorrangig ist.

> **Note**
>
> Es ist möglich, dass eine andere Anforderung zwischen dem Zeitpunkt der Initiierung und jenem des Abschlusses eines Multipart-Uploads stattgefunden hat. Wenn zum Beispiel ein anderer Vorgang einen Schlüssel löscht, nachdem Sie einen mehrteiligen Upload mit diesem Schlüssel initiiert haben, aber bevor Sie ihn abschließen, kann die Antwort für den Abschluss des Multipart-Uploads möglicherweise eine erfolgreiche Objekterstellung anzeigen, ohne dass Sie das Objekt je zu Gesicht bekommen haben.

Multipart-Upload und Preise

Sobald Sie einen mehrteiligen Upload initiieren, speichert Amazon S3 alle Teile, bis Sie den Upload entweder abschließen oder abbrechen. Während seiner gesamten Lebensdauer werden Ihnen der gesamte Speicher, die Bandbreite und die Anforderungen für diesen mehrteiligen Upload und die zugehörigen Teile in Rechnung gestellt. Wenn Sie den Multipart-Upload abbrechen, löscht Amazon S3 alle Upload-Artefakte und alle Teile, die Sie hochgeladen haben, und die Verrechnung wird eingestellt. Weitere Informationen zu Preisen finden Sie unter Amazon S3-Preise.

Abbruch unvollständiger mehrteiliger Uploads unter Verwendung einer Bucket-Lebenszyklusrichtlinie

Nach der Initiierung eines mehrteiligen Uploads beginnen Sie mit dem Hochladen von Teilen. Amazon S3 speichert diese Teile, erstellt das Objekt aus diesen Teilen jedoch erst, nachdem alle Teile hochgeladen wurden und Sie `successful` eine Anforderung für den Abschluss des mehrteiligen Uploads gesendet haben. (Sie sollten überprüfen, ob Ihre Anforderung für den Abschluss des mehrteiligen Uploads erfolgreich war.) Nach dem Empfang der Anforderung auf Abschluss des Multipart-Uploads erzeugt Amazon S3 aus den Teilen ein Objekt.

Wenn die Anforderung für den Abschluss des mehrteiligen Uploads nicht erfolgreich ist, fügt Amazon S3 die Teile nicht zusammen und erstellt daher auch kein Objekt. Die Teile bleiben in Amazon S3 gespeichert und Sie bezahlen für die in Amazon S3 gespeicherten Teile. Als Best Practice empfehlen wir Ihnen, eine Lebenszyklusregel zu konfigurieren, um Ihre Speicherkosten zu minimieren (mit der Aktion `AbortIncompleteMultipartUpload`).

Amazon S3 unterstützt eine Bucket-Lebenszyklusregel, mit der Sie Amazon S3 anweisen können, unvollständige mehrteilige Uploads abzubrechen, die nicht innerhalb einer bestimmten Anzahl von Tagen nach der Initiierung abgeschlossen werden. Wenn ein mehrteiliger Upload nicht innerhalb des Zeitrahmens abgeschlossen ist, wird er für eine Abbruchoperation zugelassen und Amazon S3 bricht den Multipart-Upload ab (und löscht die dem Multipart-Upload zugeordneten Teile).

Das folgende Beispiel zeigt eine Lebenszykluskonfiguration, die eine Regel mit der Aktion `AbortIncompleteMultipartUpload` spezifiziert.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Prefix></Prefix>
        <Status>Enabled</Status>
        <AbortIncompleteMultipartUpload>
          <DaysAfterInitiation>7</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

In dem Beispiel gibt die Regel keinen Wert für das `Prefix`-Element (Objektschlüsselnamen-Präfix) an. Daher gilt er für alle Objekte im Bucket, für die Sie mehrteilige Uploads initiiert haben. Alle mehrteiligen Uploads, die initiiert wurden und nicht innerhalb von sieben Tagen abgeschlossen wurden, kommen für

einen Abbruchvorgang in Frage. Die Abbruchaktion hat keine Auswirkung auf abgeschlossene mehrteilige Uploads.

Weitere Informationen zur Bucket-Lebenszykluskonfiguration finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

> **Note**
>
> Wenn der mehrteilige Upload innerhalb der in der Regel angegebenen Anzahl von Tagen abgeschlossen wird, trifft die Lifecycle-Aktion `AbortIncompleteMultipartUpload` nicht zu (d. h., Amazon S3 führt keine Aktion aus). Diese Aktion gilt auch nicht für Objekte, durch diese Lebenszyklusaktion werden keine Objekte gelöscht.

Der folgende CLI-Befehl `put-bucket-lifecycle-configuration` fügt die Lifecycle-Konfiguration für den angegebenen Bucket hinzu.

```
$ aws s3api put-bucket-lifecycle-configuration  \
        --bucket bucketname  \
        --lifecycle-configuration filename-containing-lifecycle-configuration
```

Um den CLI-Befehl zu testen, machen Sie Folgendes:

1. Einrichten des AWS CLI. Anleitungen finden Sie unter Einrichten der AWS-CLI (p. 808).
2. Speichern Sie die folgende Lebenszykluskonfiguration in einer Datei (lifecycle.json). Die Beispielkonfiguration gibt ein leeres Präfix aus und gilt daher für alle Objekte im Bucket. Sie können ein Präfix angeben, um die Richtlinie auf eine Teilmenge von Objekten zu beschränken.

   ```
   {
       "Rules": [
           {
               "ID": "Test Rule",
               "Status": "Enabled",
               "Filter": {
                   "Prefix": ""
               },
               "AbortIncompleteMultipartUpload": {
                   "DaysAfterInitiation": 7
               }
           }
       ]
   }
   ```
3. Führen Sie den folgenden CLI-Befehl aus, um die Lebenszykluskonfiguration auf Ihrem Bucket einzurichten.

   ```
   aws s3api put-bucket-lifecycle-configuration   \
   --bucket bucketname  \
   --lifecycle-configuration file://lifecycle.json
   ```
4. Rufen Sie zur Prüfung die Lebenszykluskonfiguration wie folgt mit dem CLI-Befehl `get-bucket-lifecycle` ab.

   ```
   aws s3api get-bucket-lifecycle  \
   --bucket bucketname
   ```
5. Verwenden Sie zum Löschen der Lebenszykluskonfiguration den CLI-Befehl `delete-bucket-lifecycle`.

   ```
   aws s3api delete-bucket-lifecycle \
   --bucket bucketname
   ```

Limits mehrteiliger Amazon S3-Uploads

Die folgende Tabelle enthält die Core-Spezifikationen für den mehrteiligen Upload. Weitere Informationen finden Sie unter Überblick Multipart Upload (p. 199).

Item	Spezifikation
Maximale Objektgröße	5 TB
Maximale Anzahl von Teilen pro Upload	10.000
Teilenummern	1 bis 10.000 (inklusive)
Teilegröße	5 MB bis 5 GB; der letzte Teil darf < 5 MB sein.
Maximale Anzahl der zurückgegebenen Teile bei einer Anforderung zum Auflisten der Teile	1000
Maximale Anzahl der zurückgegebenen mehrteiligen Uploads bei einer Anforderung zum Auflisten mehrteiliger Uploads	1000

API-Unterstützung für mehrteilige Uploads

Sie können ein AWS SDK verwenden, um ein Objekt in Teilen hochzuladen. Die folgenden AWS SDK-Bibliotheken unterstützen den mehrteiligen Upload:

- AWS SDK for Java
- AWS SDK für .NET
- AWS SDK für PHP
- AWS SDK für Ruby
- AWS SDK for Python (Boto)
- AWS SDK for JavaScript in Node.js

Diese Bibliotheken bieten eine hohe Abstraktion, wodurch der mehrteilige Upload von Objekten sehr einfach wird. Falls in Ihrer Anwendung erforderlich, können Sie die REST API jedoch auch direkt verwenden. Die folgenden Abschnitte im Amazon Simple Storage Service API Reference beschreiben die REST API für mehrteilige Uploads.

- Multipart-Upload initiieren
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Die folgenden Abschnitte im AWS Command Line Interface beschreiben die REST API für mehrteilige Uploads.

- Multipart-Upload initiieren
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Multipart-Upload-API und Berechtigungen

Eine Person muss über die erforderlichen Berechtigungen verfügen, um die Multipart-Upload-Vorgänge zu verwenden. Sie können Zugriffskontrolllisten (ACLs), die Bucket-Richtlinie oder die Benutzerrichtlinie verwenden, um Einzelberechtigungen für die Ausführung dieser Operationen zu erteilen. Die folgende Tabelle listet die erforderlichen Berechtigungen für verschiedene mehrteilige Uploadoperationen bei der Verwendung von ACLs, einer Bucket-Richtlinie oder einer Benutzerrichtlinie auf.

Action	Erforderliche Berechtigungen
Multipart-Upload initiieren	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Multipart-Upload zu initiieren. Der Bucket-Eigentümer kann anderen Prinzipalen erlauben, die Aktion `s3:PutObject` auszuführen.
Initiator	Containerelement, das identifiziert, wer den Multipart-Upload initiiert hat. Wenn der Initiator ein AWS-Konto ist, gibt dieses Element die gleichen Informationen wie das Eigentümerelement zurück. Wenn der Initiator ein IAM-Benutzer ist, stellt dieses Element den Benutzer-ARN und den Anzeigenamen bereit.
Upload Part	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Teil hochzuladen. Der Bucket-Eigentümer muss dem Initiator erlauben, die Aktion `s3:PutObject` für ein Objekt auszuführen, damit der Initiator einen Teil für dieses Objekt hochladen kann.
Teil hochladen (kopieren)	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Teil hochzuladen. Da Sie einen Teil eines vorhandenen Objekts hochladen, müssen Sie die Berechtigung `s3:GetObject` für das Quellobjekt besitzen. Damit der Initiator einen Teil eines Objekts hochladen kann, muss der Bucket-Besitzer dem Initiator eine Genehmigung für die Ausführung der Aktion `s3:PutObject` für das Objekt erteilen.
Abschließen eines mehrteiligen Uploads	Sie müssen die Aktion `s3:PutObject` für ein Objekt ausführen können, um einen Multipart-Upload abzuschließen. Der Bucket-Eigentümer muss dem Initiator erlauben, die Aktion `s3:PutObject` für ein Objekt auszuführen, damit der Initiator den mehrteiligen Upload für dieses Objekt abschließen kann.
Abort Multipart Upload	Sie müssen die Aktion `s3:AbortMultipartUpload` ausführen können, um einen mehrteiligen Upload abzubrechen. Standardmäßig können der Bucket-Eigentümer und der Initiator des Multipart-Uploads diese Aktion ausführen. Wenn der Initiator ein IAM-Benutzer ist, kann das AWS-Konto dieses Benutzers diesen Multipart-Upload auch abbrechen.

Action	Erforderliche Berechtigungen
	Zusätzlich zu diesen Standardberechtigungen, kann der Bucket-Eigentümer anderen Prinzipalen erlauben, die Aktion `s3:AbortMultipartUpload` auszuführen. Der Bucket-Eigentümer kann jedem Prinzipal verbieten, die Aktion `s3:AbortMultipartUpload` auszuführen.
Teile auflisten	Sie müssen die Aktion `s3:ListMultipartUploadParts` ausführen können, um Teileauflistungen in einem mehrteiligen Upload anzufragen. Standardmäßig kann der Bucket-Eigentümer Teileauflistungen von Multipart-Uploads in seinem Bucket anfordern. Der Initiator des mehrteiligen Uploads hat die Berechtigung, Teileauflistungen des spezifischen Multipart-Uploads aufzulisten. Wenn der Initiator des mehrteiligen Uploads ein IAM-Benutzer ist, besitzt das diesen IAM-Benutzer kontrollierende AWS-Konto ebenfalls die Berechtigung, Teile dieses Uploads aufzulisten. Zusätzlich zu diesen Standardberechtigungen, kann der Bucket-Eigentümer anderen Prinzipalen erlauben, die Aktion `s3:ListMultipartUploadParts` auszuführen. Der Bucket-Eigentümer kann jedem Prinzipal auch verbieten, die Aktion `s3:ListMultipartUploadParts` auszuführen.
List Multipart Uploads	Sie müssen die Aktion `s3:ListBucketMultipartUploads` für einen Bucket ausführen können, um eine Multipart-Uploads-Auflistung für diesen Bucket anzufragen. Zusätzlich zu diesen Standardberechtigungen, kann der Bucket-Eigentümer anderen Prinzipalen erlauben, die Aktion `s3:ListBucketMultipartUploads` für ein Bucket auszuführen.
Berechtigungen für die AWS KMS-Verschlüsselung und -Entschlüsselung	Um einen mehrteiligen Upload mit Verschlüsselung unter Verwendung eines AWS Key Management Service (AWS KMS) Kundenhauptschlüssels (CMK) durchzuführen, muss der Anforderer die Berechtigung für die Aktionen `kms:Encrypt`, `kms:Decrypt`, `kms:ReEncrypt*`, `kms:GenerateDataKey*` und `kms:DescribeKey` für den Schlüssel haben. Diese Berechtigungen sind erforderlich, da Amazon S3 Daten aus den verschlüsselten Teilen der Datei entschlüsseln und lesen muss, bevor es den Multipart-Upload vornehmen kann. Wenn sich Ihr IAM-Benutzer oder Ihre Rolle im selben AWS-Konto wie der AWS KMS-CMK befindet, dann müssen Sie diese Berechtigungen für die Schlüsselrichtlinie haben. Wenn Ihr IAM-Benutzer oder Ihre Rolle zu einem anderen Konto als das CMK gehört, müssen Sie über die Berechtigungen sowohl für die Schlüsselrichtlinie als auch für Ihren IAM-Benutzer oder Ihre Rolle verfügen. >

Weitere Informationen zur Beziehung zwischen ACL-Berechtigungen und Berechtigungen in Zugriffsrichtlinien finden Sie unter Zuordnung der ACL-Berechtigungen und Zugriffsrichtlinienberechtigungen (p. 483). Informationen zu IAM-Benutzern finden Sie unter Arbeiten mit Benutzern und Gruppen.

Verwendung des AWS Java SDK für mehrteilige Uploads (High-Level-API)

Themen

Das AWS SDK for Java stellt zur Vereinfachung mehrteiliger Uploads eine High-Level-API mit dem Namen `TransferManager` bereit (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)). Sie können Daten aus einer Datei oder einem Stream hochladen. Sie können auch erweiterte Optionen auswählen, wie beispielsweise die Teilegröße, die Sie für den mehrteiligen Upload verwenden möchten, oder die Anzahl der gleichzeitigen Threads, die Sie für den Upload der Teile verwenden möchten. Sie können auch optionale Objekteigenschaften, die Speicherklasse oder die ACL festlegen. Sie verwenden die Klassen `PutObjectRequest` und `TransferManagerConfiguration`, um die erweiterten Optionen festzulegen.

Wenn möglich, versucht der `TransferManager` mehrere Threads zu verwenden, um mehrere Teile eines einzigen Uploads geleichzeitig hochzuladen. Bei großen Inhalten und hoher Bandbreite kann dies den Durchsatz erheblich erhöhen.

Zusätzlich zur Datei-Upload-Funktionalität ermöglicht die Klasse `TransferManager` Ihnen, laufende mehrteilige Uploads abzubrechen. Ein Upload wird als laufend betrachtet, nachdem Sie ihn initiiert haben und bis er abgeschlossen ist oder Sie ihn abbrechen. Der `TransferManager` bricht alle laufenden mehrteiligen Uploads für einen angegebenen Bucket ab, die vor einem angegebenen Datum und einer angegebenen Uhrzeit initiiert wurden.

Weitere Informationen zu mehrteiligen Uploads, einschließlich über zusätzliche Funktionalität, die von Low-Level-API-Methoden bereitgestellt wird, finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).

Hochladen einer Datei

Example

Im folgenden Beispiel wird veranschaulicht, wie Sie ein Objekt mithilfe der High-Level-Java-API für mehrteilige Uploads (der `TransferManager`-Klasse) hochladen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;

import java.io.File;

public class HighLevelMultipartUpload {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";
        String filePath = "*** Path for file to upload ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // TransferManager processes all transfers asynchronously,
```

```
            // so this call returns immediately.
            Upload upload = tm.upload(bucketName, keyName, new File(filePath));
            System.out.println("Object upload started");

            // Optionally, wait for the upload to finish before continuing.
            upload.waitForCompletion();
            System.out.println("Object upload complete");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Abbrechen von mehrteiligen Uploads

Example

Im folgenden Beispiel wird eine High-Level-Java-API (die `TransferManager`-Klasse) zum Abbrechen aller laufenden mehrteiligen Uploads verwendet, die vor mehr als einer Woche für einen bestimmten Bucket initiiert wurden. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;

import java.util.Date;

public class HighLevelAbortMultipartUpload {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // sevenDays is the duration of seven days in milliseconds.
            long sevenDays = 1000 * 60 * 60 * 24 * 7;
            Date oneWeekAgo = new Date(System.currentTimeMillis() - sevenDays);
            tm.abortMultipartUploads(bucketName, oneWeekAgo);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
```

```
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client couldn't
            // parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verfolgen des Fortschritts eines mehrteiligen Uploads

Die High-Level-Java-API für mehrteilige Uploads stellt die Listen-Schnittstelle `ProgressListener` bereit, um den Fortschritt beim Hochladen eines Objekts zu Amazon S3 nachzuverfolgen. Fortschrittsereignisse treten periodisch auf und benachrichtigen den Listener, dass Bytes übertragen wurden.

Das folgende Beispiel veranschaulicht, wie Sie ein `ProgressEvent`-Ereignis abonnieren und eine Verarbeitungsroutine schreiben:

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.event.ProgressEvent;
import com.amazonaws.event.ProgressListener;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.PutObjectRequest;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;

import java.io.File;

public class HighLevelTrackMultipartUpload {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";
        String filePath = "*** Path to file to upload ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();
            PutObjectRequest request = new PutObjectRequest(bucketName, keyName, new
 File(filePath));

            // To receive notifications when bytes are transferred, add a
            // ProgressListener to your request.
            request.setGeneralProgressListener(new ProgressListener() {
                public void progressChanged(ProgressEvent progressEvent) {
                    System.out.println("Transferred bytes: " +
 progressEvent.getBytesTransferred());
                }
            });
            // TransferManager processes all transfers asynchronously,
```

```
            // so this call returns immediately.
            Upload upload = tm.upload(request);

            // Optionally, you can wait for the upload to finish before continuing.
            upload.waitForCompletion();
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwendung des AWS Java SDK für mehrteilige Uploads (Low-Level-API)

Themen

Das AWS SDK for Java stellt eine Low-Level-API bereit, die der Amazon S3-REST-API für mehrteilige Uploads sehr ähnlich ist (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)). Verwenden Sie die Low-Level-API, wenn Sie mehrteilige Uploads unterbrechen und fortsetzen müssen, die Teilegrößen während des Uploads ändern müssen oder die Größe der Daten nicht vorab kennen. Wenn sich Ihnen diese Anforderungen nicht stellen, verwenden Sie die High-Level API (siehe Verwendung des AWS Java SDK für mehrteilige Uploads (High-Level-API) (p. 205)).

Hochladen einer Datei

Im folgenden Beispiel wird gezeigt, wie Sie mithilfe der Low-Level-Java-Klassen eine Datei hochladen. Es führt die folgenden Schritte aus:

- Startet einen mehrteiligen Upload mit der Methode `AmazonS3Client.initiateMultipartUpload()` und übergibt ein `InitiateMultipartUploadRequest`-Objekt.
- Speichern Sie die Upload-ID, die von der Methode `AmazonS3Client.initiateMultipartUpload()` zurückgegeben wird. Sie geben diese Upload-ID bei jeder nachfolgenden mehrteiligen Upload-Operation an.
- Lädt die Teile des Objekts hoch. Für jedes Teil rufen Sie die Methode `AmazonS3Client.uploadPart()` auf. Sie stellen die Informationen für das Hochladen von Teilen in einem `UploadPartRequest`-Objekt bereit.
- Speichert für jedes Teil das ETag aus der Antwort der Methode `AmazonS3Client.uploadPart()` in einer Liste. Sie verwenden die ETag-Werte, um den mehrteiligen Upload fertigzustellen.
- Ruft die Methode `AmazonS3Client.completeMultipartUpload()` auf, um den mehrteiligen Upload fertigzustellen.

Example

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class LowLevelMultipartUpload {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String filePath = "*** Path to file to upload ***";

        File file = new File(filePath);
        long contentLength = file.length();
        long partSize = 5 * 1024 * 1024; // Set part size to 5 MB.

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Create a list of ETag objects. You retrieve ETags for each object part
 uploaded,
            // then, after each individual part has been uploaded, pass the list of ETags
 to
            // the request to complete the upload.
            List<PartETag> partETags = new ArrayList<PartETag>();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(bucketName, keyName);
            InitiateMultipartUploadResult initResponse =
 s3Client.initiateMultipartUpload(initRequest);

            // Upload the file parts.
            long filePosition = 0;
            for (int i = 1; filePosition < contentLength; i++) {
                // Because the last part could be less than 5 MB, adjust the part size as
 needed.
                partSize = Math.min(partSize, (contentLength - filePosition));

                // Create the request to upload a part.
                UploadPartRequest uploadRequest = new UploadPartRequest()
                        .withBucketName(bucketName)
                        .withKey(keyName)
                        .withUploadId(initResponse.getUploadId())
                        .withPartNumber(i)
                        .withFileOffset(filePosition)
                        .withFile(file)
                        .withPartSize(partSize);

                // Upload the part and add the response's ETag to our list.
                UploadPartResult uploadResult = s3Client.uploadPart(uploadRequest);
                partETags.add(uploadResult.getPartETag());

                filePosition += partSize;
```

```
            }

            // Complete the multipart upload.
            CompleteMultipartUploadRequest compRequest = new
 CompleteMultipartUploadRequest(bucketName, keyName,
                    initResponse.getUploadId(), partETags);
            s3Client.completeMultipartUpload(compRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Auflisten von mehrteiligen Uploads

Example

Im folgenden Beispiel wird gezeigt, wie Sie mithilfe der Low-Level-API eine Liste laufender mehrteiliger Uploads abrufen:

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListMultipartUploadsRequest;
import com.amazonaws.services.s3.model.MultipartUpload;
import com.amazonaws.services.s3.model.MultipartUploadListing;

import java.util.List;

public class ListMultipartUploads {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Retrieve a list of all in-progress multipart uploads.
            ListMultipartUploadsRequest allMultipartUploadsRequest = new
 ListMultipartUploadsRequest(bucketName);
            MultipartUploadListing multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);
            List<MultipartUpload> uploads = multipartUploadListing.getMultipartUploads();

            // Display information about all in-progress multipart uploads.
            System.out.println(uploads.size() + " multipart upload(s) in progress.");
            for (MultipartUpload u : uploads) {
                System.out.println("Upload in progress: Key = \"" + u.getKey() + "\", id =
 " + u.getUploadId());
```

```
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Abbrechen eines mehrteiligen Uploads

Durch Aufruf der Methode `AmazonS3Client.abortMultipartUpload()` können Sie einen laufenden mehrteiligen Upload abbrechen. Diese Methode löscht alle Teile, die in Amazon S3 hochgeladen wurden, und gibt die Ressourcen frei. Sie stellen die Upload-ID, den Bucket-Namen und den Schlüsselnamen bereit.

Example

Im folgenden Beispiel wird veranschaulicht, wie Sie mithilfe der Low-Level-Java-API mehrteilige Uploads abbrechen.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AbortMultipartUploadRequest;
import com.amazonaws.services.s3.model.ListMultipartUploadsRequest;
import com.amazonaws.services.s3.model.MultipartUpload;
import com.amazonaws.services.s3.model.MultipartUploadListing;

import java.util.List;

public class LowLevelAbortMultipartUpload {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Find all in-progress multipart uploads.
            ListMultipartUploadsRequest allMultipartUploadsRequest = new
 ListMultipartUploadsRequest(bucketName);
            MultipartUploadListing multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);

            List<MultipartUpload> uploads = multipartUploadListing.getMultipartUploads();
            System.out.println("Before deletions, " + uploads.size() + " multipart uploads
 in progress.");

            // Abort each upload.
            for (MultipartUpload u : uploads) {
```

```
                System.out.println("Upload in progress: Key = \"" + u.getKey() + "\", id =
 " + u.getUploadId());
                s3Client.abortMultipartUpload(new AbortMultipartUploadRequest(bucketName,
 u.getKey(), u.getUploadId()));
                System.out.println("Upload deleted: Key = \"" + u.getKey() + "\", id = " +
 u.getUploadId());
            }

            // Verify that all in-progress multipart uploads have been aborted.
            multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);
            uploads = multipartUploadListing.getMultipartUploads();
            System.out.println("After aborting uploads, " + uploads.size() + " multipart
 uploads in progress.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Note

Anstatt mehrteilige Uploads einzeln abzubrechen, können Sie alle laufenden mehrteiligen Uploads abbrechen, die vor einem bestimmten Zeitpunkt initiiert wurden. Diese Bereinigungsoperation ist praktisch, um mehrteilige Uploads abzubrechen, die Sie initiiert haben, die aber nicht abgeschlossen oder abgebrochen wurden. Weitere Informationen finden Sie unter Abbrechen von mehrteiligen Uploads (p. 207).

Verwendung des AWS SDK für .NET für mehrteilige Uploads (High-Level-API)

Themen

Das AWS SDK für .NET stellt eine High-Level-API zur Vereinfachung mehrteiliger Uploads bereit (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)). Sie können Daten aus einer Datei, einem Verzeichnis oder einem Stream hochladen. Weitere Informationen zu mehrteiligen Amazon S3-Uploads finden Sie unter Überblick Multipart Upload (p. 199).

Die Klasse `TransferUtility` bietet Methoden für das Hochladen von Dateien und Verzeichnissen, Verfolgen des Upload-Fortschritts und Abbrechen von mehrteiligen Uploads.

Hochladen einer Datei in einen S3-Bucket unter Verwendung des AWS SDK für .NET (High-Level-API)

Um eine Datei in einen S3-Bucket hochzuladen, verwenden Sie die Klasse `TransferUtility`. Beim Hochladen von Daten aus einer Datei müssen Sie den Schlüsselnamen des Objekts angeben. Andernfalls

verwendet die API den Dateinamen für den Schlüsselnamen. Beim Hochladen von Daten aus einem Stream müssen Sie den Schlüsselnamen des Objekts angeben.

Um erweiterte Upload-Optionen festzulegen, —beispielsweise die Größe des Teil-Uploads, die Anzahl der Threads bei gleichzeitigem Hochladen von Upload-Teilen, Metadaten, die Speicherklasse oder die ACL—, verwenden Sie die Klasse `TransferUtilityUploadRequest`.

Das folgende C#-Beispiel lädt eine Datei in mehreren Teilen in einen Amazon S3-Bucket hoch. Es veranschaulicht, wie Sie zahlreiche `TransferUtility.Upload`-Überladungen zum Hochladen einer Datei verwenden. Jeder nachfolgende Aufruf zum Hochladen ersetzt den vorherigen Upload. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version des AWS SDK für .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadFileMPUHighLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string keyName = "*** provide a name for the uploaded object ***";
        private const string filePath = "*** provide the full path name of the file to
 upload ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            UploadFileAsync().Wait();
        }

        private static async Task UploadFileAsync()
        {
            try
            {
                var fileTransferUtility =
                    new TransferUtility(s3Client);

                // Option 1. Upload a file. The file name is used as the object key name.
                await fileTransferUtility.UploadAsync(filePath, bucketName);
                Console.WriteLine("Upload 1 completed");

                // Option 2. Specify object key name explicitly.
                await fileTransferUtility.UploadAsync(filePath, bucketName, keyName);
                Console.WriteLine("Upload 2 completed");

                // Option 3. Upload data from a type of System.IO.Stream.
                using (var fileToUpload =
                    new FileStream(filePath, FileMode.Open, FileAccess.Read))
                {
                    await fileTransferUtility.UploadAsync(fileToUpload,
                                                bucketName, keyName);
                }
                Console.WriteLine("Upload 3 completed");

                // Option 4. Specify advanced settings.
```

```
                var fileTransferUtilityRequest = new TransferUtilityUploadRequest
                {
                    BucketName = bucketName,
                    FilePath = filePath,
                    StorageClass = S3StorageClass.StandardInfrequentAccess,
                    PartSize = 6291456, // 6 MB.
                    Key = keyName,
                    CannedACL = S3CannedACL.PublicRead
                };
                fileTransferUtilityRequest.Metadata.Add("param1", "Value1");
                fileTransferUtilityRequest.Metadata.Add("param2", "Value2");

                await fileTransferUtility.UploadAsync(fileTransferUtilityRequest);
                Console.WriteLine("Upload 4 completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }

        }
    }
}
```

Weitere Infos

AWS SDK für .NET

Upload eines Verzeichnisses

Mit der Klasse `TransferUtility` können Sie ein gesamtes Verzeichnis hochladen. Standardmäßig lädt die API nur die Dateien im Stammverzeichnis des angegebenen Verzeichnisses hoch. Sie können jedoch festlegen, dass die Dateien in allen Unterverzeichnissen rekursiv hochgeladen werden sollen.

Um Dateien im angegebenen Verzeichnis basierend auf Filterkriterien anzugeben, geben Sie Filterausdrücke an. Wenn Sie z. B. nur die .pdf-Dateien aus einem Verzeichnis hochladen möchten, geben Sie den Filterausdruck `"*.pdf"` ein.

Wenn Sie Dateien aus einem Verzeichnis hochladen, geben Sie keine Schlüsselnamen für die resultierenden Objekte an. Amazon S3 konstruiert die Schlüsselnamen anhand des ursprünglichen Dateipfads. Angenommen, ein Verzeichnis mit dem Namen `c:\myfolder` besitzt die folgende Verzeichnisstruktur:

Example

```
C:\myfolder
      \a.txt
      \b.pdf
      \media\
             An.mp3
```

Wenn Sie dieses Verzeichnis hochladen, verwendet Amazon S3 die folgenden Schlüsselnamen:

Example

```
a.txt
```

```
b.pdf
media/An.mp3
```

Example

Das folgende C#-Beispiel lädt ein Verzeichnis in einen Amazon S3-Bucket hoch. Es veranschaulicht, wie Sie zahlreiche `TransferUtility.UploadDirectory`-Überladungen zum Hochladen des Verzeichnisses verwenden. Jeder nachfolgende Aufruf zum Hochladen ersetzt den vorherigen Upload. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadDirMPUHighLevelAPITest
    {
        private const string existingBucketName = "*** bucket name ***";
        private const string directoryPath = @"*** directory path ***";
        // The example uploads only .txt files.
        private const string wildCard = "*.txt";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            UploadDirAsync().Wait();
        }

        private static async Task UploadDirAsync()
        {
            try
            {
                var directoryTransferUtility =
                    new TransferUtility(s3Client);

                // 1. Upload a directory.
                await directoryTransferUtility.UploadDirectoryAsync(directoryPath,
                    existingBucketName);
                Console.WriteLine("Upload statement 1 completed");

                // 2. Upload only the .txt files from a directory
                //    and search recursively.
                await directoryTransferUtility.UploadDirectoryAsync(
                                                directoryPath,
                                                existingBucketName,
                                                wildCard,
                                                SearchOption.AllDirectories);
                Console.WriteLine("Upload statement 2 completed");

                // 3. The same as Step 2 and some optional configuration.
                //    Search recursively for .txt files to upload.
                var request = new TransferUtilityUploadDirectoryRequest
                {
                    BucketName = existingBucketName,
                    Directory = directoryPath,
                    SearchOption = SearchOption.AllDirectories,
                    SearchPattern = wildCard
```

```
                };

                await directoryTransferUtility.UploadDirectoryAsync(request);
                Console.WriteLine("Upload statement 3 completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an object",
 e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Unknown encountered on server. Message:'{0}' when writing an object",
 e.Message);
            }
        }
    }
}
```

Abbrechen mehrteiliger Uploads zu einem S3-Bucket unter Verwendung des AWS SDK für .NET (High-Level-API)

Um laufende mehrteilige Uploads abzubrechen, verwenden Sie die Klasse `TransferUtility` des AWS SDK for .NET. Sie müssen einen Wert für `DateTime` angeben. Die API bricht dann alle mehrteiligen Uploads ab, die vor dem angegebenen Datum und der angegebenen Uhrzeit initiiert wurden, und entfernt die hochgeladenen Teile. Ein Upload wird als laufend betrachtet, nachdem Sie ihn initiiert haben und bis er abgeschlossen ist oder Sie ihn abbrechen.

Da Ihnen der den hochgeladenen Teile zugeordnete Speicher vollständig in Rechnung gestellt wird, sollten Sie den mehrteiligen Upload abschließen, damit das Objekt erstellt wird, oder ihn abbrechen, um hochgeladene Teile zu entfernen. Weitere Informationen zu mehrteiligen Amazon S3-Uploads finden Sie unter Überblick Multipart Upload (p. 199). Weitere Informationen zu den Preisen erhalten Sie unter Multipart-Upload und Preise (p. 201).

Das folgende C#-Bespiel bricht alle laufenden mehrteiligen Uploads ab, die vor mehr als einer Woche für einen bestimmten Bucket initiiert wurden. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version des AWS SDK für .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class AbortMPUUsingHighLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            AbortMPUAsync().Wait();
        }

        private static async Task AbortMPUAsync()
```

```
        {
            try
            {
                var transferUtility = new TransferUtility(s3Client);

                // Abort all in-progress uploads initiated before the specified date.
                await transferUtility.AbortMultipartUploadsAsync(
                    bucketName, DateTime.Now.AddDays(-7));
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Note

Sie können auch einen bestimmten mehrteiligen Upload abbrechen. Weitere Informationen finden Sie unter Auflisten von mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-level) (p. 222).

Weitere Infos

AWS SDK für .NET

Verfolgen des Fortschritts eines mehrteiligen Uploads in einen S3-Bucket mit AWS SDK für .NET (High-level-API)

Das folgende C#-Beispiel lädt eine Datei unter Verwendung der Klasse `TransferUtility` in einen S3-Bucket hoch und verfolgt den Fortschritt des Uploads. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version des AWS SDK für .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TrackMPUUsingHighLevelAPITest
    {
        private const string bucketName = "*** provide the bucket name ***";
        private const string keyName = "*** provide the name for the uploaded object ***";
        private const string filePath = " *** provide the full path name of the file to
 upload **";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;


        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
```

```
            TrackMPUAsync().Wait();
        }

        private static async Task TrackMPUAsync()
        {
            try
            {
                var fileTransferUtility = new TransferUtility(s3Client);

                // Use TransferUtilityUploadRequest to configure options.
                // In this example we subscribe to an event.
                var uploadRequest =
                    new TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
                        FilePath = filePath,
                        Key = keyName
                    };

                uploadRequest.UploadProgressEvent +=
                    new EventHandler<UploadProgressArgs>
                        (uploadRequest_UploadPartProgressEvent);

                await fileTransferUtility.UploadAsync(uploadRequest);
                Console.WriteLine("Upload completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }

        static void uploadRequest_UploadPartProgressEvent(object sender, UploadProgressArgs
 e)
        {
            // Process event.
            Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
        }
    }
}
```

Weitere Infos

AWS SDK für .NET

Verwendung des AWS SDK für .NET für mehrteilige Uploads (Low-Level-API)

Das AWS SDK für .NET stellt eine Low-Level-API bereit, die der Amazon S3-REST-API für mehrteilige Uploads sehr ähnlich ist (siehe Verwenden der REST API für einen mehrteiligen Upload (p. 228)). Verwenden Sie die Low-Level-API, wenn Sie mehrteilige Uploads unterbrechen und fortsetzen müssen, die Teilegrößen während des Uploads ändern müssen oder die Größe der Daten nicht vorab kennen. Verwenden Sie die High-Level API (siehe Verwendung des AWS SDK für .NET für mehrteilige Uploads (High-Level-API) (p. 213)), wenn Sie diese Anforderungen nicht haben.

Themen

- Hochladen einer Datei in einen S3-Bucket unter Verwendung von AWS SDK für .NET (Low-Level-API) (p. 220)

- Auflisten von mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-level) (p. 222)
- Verfolgen des Fortschritts eines mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-Level) (p. 222)
- Abbrechen von mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-level) (p. 223)

Hochladen einer Datei in einen S3-Bucket unter Verwendung von AWS SDK für .NET (Low-Level-API)

Das folgende C#-Beispiel veranschaulicht, wie Sie eine Datei mit der Low-Level-AWS SDK für .NET-API für mehrteilige Uploads in einen S3-Bucket hochladen. Informationen zu mehrteiligen Amazon S3-Uploads finden Sie unter Überblick Multipart Upload (p. 199).

> **Note**
>
> Wenn Sie mit der AWS SDK für .NET-API große Objekte hochladen, kann ein Timeout auftreten, während Daten in den Anfrage-Stream geschrieben werden. Sie können mit `UploadPartRequest` ein explizites Timeout festlegen.

In dem folgenden C#-Beispiel wird eine Datei mithilfe der Low-Level-API für mehrteilige Uploads in einen S3-Bucket hochgeladen. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version des AWS SDK für .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadFileMPULowLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string keyName = "*** provide a name for the uploaded object ***";
        private const string filePath = "*** provide the full path name of the file to
 upload ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Uploading an object");
            UploadObjectAsync().Wait();
        }

        private static async Task UploadObjectAsync()
        {
            // Create list to store upload part responses.
            List<UploadPartResponse> uploadResponses = new List<UploadPartResponse>();

            // Setup information required to initiate the multipart upload.
            InitiateMultipartUploadRequest initiateRequest = new
 InitiateMultipartUploadRequest
```

```
            {
                BucketName = bucketName,
                Key = keyName
            };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // Upload parts.
            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
                Console.WriteLine("Uploading parts");

                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                        {
                            BucketName = bucketName,
                            Key = keyName,
                            UploadId = initResponse.UploadId,
                            PartNumber = i,
                            PartSize = partSize,
                            FilePosition = filePosition,
                            FilePath = filePath
                        };

                    // Track upload progress.
                    uploadRequest.StreamTransferProgress +=
                        new
EventHandler<StreamTransferProgressArgs>(UploadPartProgressEventCallback);

                    // Upload a part and add the response to our list.
                    uploadResponses.Add(await s3Client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                // Setup to complete the upload.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                    {
                        BucketName = bucketName,
                        Key = keyName,
                        UploadId = initResponse.UploadId
                     };
                completeRequest.AddPartETags(uploadResponses);

                // Complete the upload.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (Exception exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown: { 0}",
exception.Message);

                // Abort the upload.
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
                    BucketName = bucketName,
```

```
                    Key = keyName,
                    UploadId = initResponse.UploadId
                };
                await s3Client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
        public static void UploadPartProgressEventCallback(object sender,
 StreamTransferProgressArgs e)
        {
            // Process event.
            Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
        }
    }
}
```

Weitere Infos

AWS SDK für .NET

Auflisten von mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-level)

Um alle aktuell ausgeführten mehrteiligen Uploads für einen bestimmten Bucket aufzulisten, verwenden Sie die Klasse `ListMultipartUploadsRequest` der Low-Level-API von AWS SDK für .NET für mehrteilige Uploads. Die Methode `AmazonS3Client.ListMultipartUploads` gibt eine Instance der Klasse `ListMultipartUploadsResponse` zurück, die Informationen über die laufenden mehrteiligen Uploads bereitstellt.

Ein laufender mehrteiliger Upload ist ein mehrteiliger Upload, der von der Anfrage für das Initiieren mehrteiliger Uploads initiiert, aber weder abgeschlossen noch abgebrochen wurde. Weitere Informationen zu mehrteiligen Amazon S3-Uploads finden Sie unter Überblick Multipart Upload (p. 199).

Das folgende C#-Beispiel veranschaulicht, wie Sie mit AWS SDK für .NET alle laufenden mehrteiligen Uploads in einem Bucket auflisten. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK für .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest
{
  BucketName = bucketName // Bucket receiving the uploads.
};

ListMultipartUploadsResponse response = await
 AmazonS3Client.ListMultipartUploadsAsync(request);
```

Weitere Infos

AWS SDK für .NET

Verfolgen des Fortschritts eines mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-Level)

Um den Fortschritt eines mehrteiligen Uploads nachzuverfolgen, verwenden Sie das Ereignis `UploadPartRequest.StreamTransferProgress`, das von der Low-Level-API für mehrteilige Uploads von AWS SDK für .NET bereitgestellt wird. Das Ereignis tritt regelmäßig auf. Es gibt Informationen wie die Gesamtanzahl der zu übertragenden Bytes und die Anzahl der übertragenen Bytes zurück.

Das folgende C#-Beispiel veranschaulicht, wie Sie den Fortschritt mehrteiliger Uploads verfolgen. Ein vollständiges C#-Beispiel mit dem folgenden Code finden Sie unter Hochladen einer Datei in einen S3-Bucket unter Verwendung von AWS SDK für .NET (Low-Level-API) (p. 220).

```
UploadPartRequest uploadRequest = new UploadPartRequest
```

```
{
// Provide the request data.
};

uploadRequest.StreamTransferProgress +=
     new EventHandler<StreamTransferProgressArgs>(UploadPartProgressEventCallback);

...
public static void UploadPartProgressEventCallback(object sender,
 StreamTransferProgressArgs e)
{
    // Process the event.
    Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
}
```

Weitere Infos

AWS SDK für .NET

Abbrechen von mehrteiligen Uploads in einen S3-Bucket mit dem AWS SDK für .NET (Low-level)

Durch Aufruf der Methode `AmazonS3Client.AbortMultipartUploadAsync` können Sie einen laufenden mehrteiligen Upload abbrechen. Diese Methode bricht nicht nur den Upload ab, sondern löscht auch alle Teile, die zu Amazon S3 hochgeladen wurden.

Um einen mehrteiligen Upload abzubrechen, stellen Sie die Upload-ID bereit sowie die Bucket- und Schlüsselnamen, die im Upload verwendet werden. Nachdem ein mehrteiliger Upload abgebrochen wurde, können Sie mit dieser Upload-ID keinen zusätzlichen Teile mehr hochladen. Weitere Informationen zu mehrteiligen Amazon S3-Uploads finden Sie unter Überblick Multipart Upload (p. 199).

Das folgende C#-Beispiel veranschaulicht, wie Sie mehrteilige Uploads abbrechen. Ein vollständiges C#-Beispiel mit dem folgenden Code finden Sie unter Hochladen einer Datei in einen S3-Bucket unter Verwendung von AWS SDK für .NET (Low-Level-API) (p. 220).

```
AbortMultipartUploadRequest abortMPURequest = new AbortMultipartUploadRequest
{
    BucketName = existingBucketName,
    Key = keyName,
    UploadId = initResponse.UploadId
};
await AmazonS3Client.AbortMultipartUploadAsync(abortMPURequest);
```

Sie können auch alle laufenden mehrteiligen Uploads abbrechen, die vor einem bestimmten Zeitpunkt initiiert wurden. Diese Bereinigungsoperation ist praktisch, um mehrteilige Uploads abzubrechen, die nicht abgeschlossen oder abgebrochen wurden. Weitere Informationen finden Sie unter Abbrechen mehrteiliger Uploads zu einem S3-Bucket unter Verwendung des AWS SDK für .NET (High-Level-API) (p. 217).

Weitere Infos

AWS SDK für .NET

Verwenden des AWS PHP SDK für mehrteilige Uploads

Sie können große Dateien in mehreren Teilen zu Amazon S3 hochladen. Sie müssen einen mehrteiligen Upload für Dateien mit mehr als 5 GB verwenden. Das AWS SDK für PHP stellt zur Vereinfachung mehrteiliger Uploads die Klasse MultipartUploader bereit.

Die `upload`-Methode der `MultipartUploader`-Klasse ist am besten für einen einfachen mehrteiligen Upload geeignet. Wenn Sie mehrteilige Uploads unterbrechen und fortsetzen müssen, die Teilegrößen während des Uploads ändern müssen oder die Größe der Daten nicht vorab kennen, verwenden Sie

die Low-Level-PHP-API. Weitere Informationen finden Sie unter Verwendung des AWS PHP SDK für mehrteilige Uploads (Low-Level-API) (p. 224).

Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198). Informationen über das Hochladen von Dateien mit weniger als 5 GB finden Sie unter Hochladen eines Objekts mit AWS SDK für PHP (p. 196).

Hochladen einer Datei unter Verwendung des mehrteiligen Uploads (High-Level)

In diesem Thema wird die Verwendung der High-Level-Klasse `Aws\S3\Model\MultipartUpload\UploadBuilder` aus dem AWS SDK für PHP für mehrteilige Datei-Uploads beschrieben. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel lädt eine Datei in einem Amazon S3-Bucket hoch. Das Beispiel veranschaulicht, wie Sie Parameter für das `MultipartUploader`-Objekt festlegen.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\Common\Exception\MultipartUploadException;
use Aws\S3\MultipartUploader;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Prepare the upload parameters.
$uploader = new MultipartUploader($s3, '/path/to/large/file.zip', [
    'bucket' => $bucket,
    'key'    => $keyname
]);

// Perform the upload.
try {
    $result = $uploader->upload();
    echo "Upload complete: {$result['ObjectURL']}" . PHP_EOL;
} catch (MultipartUploadException $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- Mehrteilige Uploads in Amazon S3
- AWS SDK für PHP-Dokumentation

Verwendung des AWS PHP SDK für mehrteilige Uploads (Low-Level-API)

Themen

- Hochladen einer Datei in mehreren Teilen mithilfe der PHP SDK Low-Level-API (p. 225)
- Auflisten von mehrteiligen Uploads mithilfe der AWS SDK für PHP-Low-Level-API (p. 226)
- Abbrechen eines mehrteiligen Uploads (p. 227)

Das AWS SDK für PHP stellt eine Low-Level-API bereit, die der Amazon S3-REST-API für mehrteilige Uploads sehr ähnlich ist (siehe Verwenden der REST API für einen mehrteiligen Upload (p. 228)). Verwenden Sie die Low-Level-API, wenn Sie mehrteilige Uploads unterbrechen und fortsetzen müssen, die Teilegrößen während des Uploads ändern müssen oder die Größe der Daten nicht vorab kennen. Verwenden Sie stets die High-Level-Abstraktionen des AWS SDK für PHP (siehe Verwenden des AWS PHP SDK für mehrteilige Uploads (p. 223)), wenn es diese Anforderungen nicht gibt.

Hochladen einer Datei in mehreren Teilen mithilfe der PHP SDK Low-Level-API

In diesem Thema wird die Verwendung der Low-Level-Methode `uploadPart` aus Version 3 des AWS SDK für PHP für das Hochladen einer Datei in mehreren Teilen gezeigt. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel lädt eine Datei mit der PHP-Low-Level-API für mehrteilige Uploads zu einem Amazon S3-Bucket hoch. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';
$filename = '*** Path to and Name of the File to Upload ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

$result = $s3->createMultipartUpload([
    'Bucket'       => $bucket,
    'Key'          => $keyname,
    'StorageClass' => 'REDUCED_REDUNDANCY',
    'ACL'          => 'public-read',
    'Metadata'     => [
        'param1' => 'value 1',
        'param2' => 'value 2',
        'param3' => 'value 3'
    ]
]);
$uploadId = $result['UploadId'];

// Upload the file in parts.
try {
    $file = fopen($filename, 'r');
    $partNumber = 1;
    while (!feof($file)) {
        $result = $s3->uploadPart([
            'Bucket'     => $bucket,
            'Key'        => $keyname,
            'UploadId'   => $uploadId,
            'PartNumber' => $partNumber,
            'Body'       => fread($file, 5 * 1024 * 1024),
        ]);
        $parts['Parts'][$partNumber] = [
            'PartNumber' => $partNumber,
            'ETag' => $result['ETag'],
        ];
        $partNumber++;

        echo "Uploading part {$partNumber} of {$filename}." . PHP_EOL;
```

```
    }
    fclose($file);
} catch (S3Exception $e) {
    $result = $s3->abortMultipartUpload([
        'Bucket'   => $bucket,
        'Key'      => $keyname,
        'UploadId' => $uploadId
    ]);

    echo "Upload of {$filename} failed." . PHP_EOL;
}

// Complete the multipart upload.
$result = $s3->completeMultipartUpload([
    'Bucket'   => $bucket,
    'Key'      => $keyname,
    'UploadId' => $uploadId,
    'MultipartUpload'    => $parts,
]);
$url = $result['Location'];

echo "Uploaded {$filename} to {$url}." . PHP_EOL;
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- Mehrteilige Uploads in Amazon S3
- AWS SDK für PHP-Dokumentation

Auflisten von mehrteiligen Uploads mithilfe der AWS SDK für PHP-Low-Level-API

Dieses Thema veranschaulicht, wie Sie die Low-Level-API-Klassen aus Version 3 von AWS SDK für PHP verwenden, um alle laufenden mehrteiligen Uploads für einen Bucket aufzulisten. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel zeigt eine Auflistung aller in einem Bucket laufenden mehrteiligen Uploads.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Retrieve a list of the current multipart uploads.
$result = $s3->listMultipartUploads([
    'Bucket' => $bucket
]);

// Write the list of uploads to the page.
print_r($result->toArray());
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class

- Mehrteilige Uploads in Amazon S3
- AWS SDK für PHP-Dokumentation

Abbrechen eines mehrteiligen Uploads

Dieses Thema beschreibt, wie Sie eine AWS SDK für PHP-Klasse aus Version 3 verwenden, um einen laufenden mehrteiligen Upload abzubrechen. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel veranschaulicht, wie Sie einen laufenden mehrteiligen Upload mit der Methode `abortMultipartUpload()` abbrechen. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';
$uploadId = '*** Upload ID of upload to Abort ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Abort the multipart upload.
$s3->abortMultipartUpload([
    'Bucket'   => $bucket,
    'Key'      => $keyname,
    'UploadId' => $uploadId,
]);
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- Mehrteilige Uploads in Amazon S3
- AWS SDK für PHP-Dokumentation

Verwenden von AWS SDK für Ruby für einen mehrteiligen Upload

AWS SDK für Ruby Version 3 unterstützt mehrteilige Amazon S3-Uploads auf zwei Arten. Bei der ersten Option können Sie verwaltete Datei-Uploads verwenden. Weitere Informationen finden Sie unter Hochladen von Dateien zu Amazon S3 im AWS-Entwickler-Blog. Verwaltete Datei-Uploads sind die empfohlene Methode zum Hochladen von Dateien zu einem Bucket. Sie bieten die folgenden Vorteile:

- Verwaltet mehrteilige Uploads von Objekten über 15 MB.
- Dateien werden im Binärmodus korrekt geöffnet, um Codierungsprobleme zu vermeiden.
- Verwendet mehrere Threads zum parallelen Hochladen von Teilen großer Objekte.

Alternativ können Sie die folgenden Multipart-Upload-Client-Operationen direkt verwenden:

- create_multipart_upload – Initiiert einen mehrteiligen Upload und gibt eine Upload-ID zurück.
- upload_part – Lädt einen Teil eines mehrteiligen Uploads hoch.

- upload_part_copy – Lädt einen Teil durch Kopieren von Daten aus einem vorhandenen Objekt hoch, das als Datenquelle dient.
- complete_multipart_upload – Schließt einen mehrteiligen Upload durch das Zusammenfügen zuvor hochgeladener Teile ab.
- abort_multipart_upload – Bricht einen mehrteiligen Upload ab.

Weitere Informationen finden Sie unter Verwenden von AWS SDK für Ruby – Version 3 (p. 813).

Verwenden der REST API für einen mehrteiligen Upload

Die folgenden Abschnitte im Amazon Simple Storage Service API Reference beschreiben die REST API für mehrteilige Uploads.

- Multipart-Upload initiieren
- Upload Part
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Sie können diese APIs verwenden, um Ihre eigenen REST-Anforderungen zu erstellen, oder Sie können AWS Command Line Interface oder eines der von uns bereitgestellten SDKs verwenden. Weitere Hinweise zur Verwendung von mehrteiligen Uploads mit der AWS CLI finden Sie unter Verwendung von AWS Command Line Interface für mehrteilige Uploads (p. 228). Weitere Informationen zu den SDKs finden Sie unter API-Unterstützung für mehrteilige Uploads (p. 203).

Verwendung von AWS Command Line Interface für mehrteilige Uploads

Die folgenden Abschnitte im AWS Command Line Interface beschreiben die Operationen für mehrteilige Uploads.

- Multipart-Upload initiieren
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Sie können auch die REST API verwenden, um Ihre eigenen REST-Anforderungen zu erstellen, oder Sie können eines der von uns bereitgestellten SDKs verwenden. Weitere Informationen zur REST API finden Sie unter Verwenden der REST API für einen mehrteiligen Upload (p. 228). Weitere Informationen zu den SDKs finden Sie unter API-Unterstützung für mehrteilige Uploads (p. 203).

Hochladen von Objekten unter Verwendung vorsignierter URLs

Eine vorsignierte URL bietet Ihnen Zugriff auf das in der URL identifizierte Objekt, wenn der Ersteller der vorsignierten URL Berechtigungen für den Zugriff auf dieses Objekt besitzt. Das bedeutet, dass Sie ein Objekt, für das Sie eine vorsignierte URL zum Hochladen eines Objekts erhalten haben, nur dann hochladen können, wenn der Ersteller der vorsignierten URL die erforderlichen Berechtigungen zum Hochladen dieses Objekts besitzt.

Standardmäßig werden alle Buckets und Objekte als privat eingestuft. Vorsignierte URLs sind nützlich, wenn Ihr Benutzer/Kunde in der Lage sein soll, ein spezifisches Objekt zu Ihrem Bucket hochzuladen, Sie jedoch nicht voraussetzen, dass dieser AWS-Sicherheitsanmeldeinformationen oder -berechtigungen besitzt. Wenn Sie eine vorsignierte URL erstellen, müssen Sie Ihre Sicherheitsanmeldeinformationen bereitstellen und anschließend einen Bucket-Namen, einen Objektschlüssel, eine HTTP-Methode (PUT für das Hochladen von Objekten) sowie ein Datum und eine Uhrzeit für den Ablauf angeben. Vorsignierte URLs sind nur für die angegebene Dauer gültig.

> Note
>
> Alle Benutzer mit gültigen Sicherheitsanmeldeinformationen können vorsignierte URLs erstellen. Um ein Objekt erfolgreich hochladen zu können, muss die vorsignierte URL jedoch von einem Benutzer erstellt werden, der die Berechtigung zur Ausführung der Operation besitzt, auf der die vorsignierte URL basiert.

Sie können eine vorsignierte URL mithilfe von REST API, AWS Command Line Interface und AWS SDK for Java, .NET, Ruby, PHP, Node.js und Python programmgesteuert generieren.

Wenn Sie Microsoft Visual Studio verwenden, können sie auch AWS Explorer verwenden, um eine vorsignierte Objekt-URL zu erstellen, ohne Code schreiben zu müssen. Alle Benutzer, die eine gültige vorsignierte URL erhalten, können anschließend ein Objekt programmgesteuert hochladen. Weitere Informationen finden Sie unter Verwenden von Amazon S3 über AWS Explorer. Anweisungen zur Installation des AWS-Explorers finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).

Themen

Hochladen eines Objekts unter Verwendung einer vorsignierten URL (AWS SDK for Java)

Sie können den AWS SDK for Java zum Generieren einer vorsignierten URL verwenden, mit der Sie oder jeder Benutzer, dem Sie die URL bereitstellen, ein Objekt zu Amazon S3 hochladen können. Wenn Sie das Objekt über die URL hochladen, erstellt Amazon S3 das Objekt in dem angegebenen Bucket. Wenn im Bucket bereits ein Objekt mit demselben Schlüssel vorhanden ist, der in der vorsignierten URL angegeben wird, ersetzt Amazon S3das vorhandene Objekt durch das hochgeladene Objekt. Um dieses Tutorial erfolgreich zu abzuschließen, müssen Sie die folgenden Aufgaben ausführen:

- Geben Sie beim Erstellen der Objekte `GeneratePresignedUrlRequest` und `HttpURLConnection` das HTTP-Verb PUT an.
- Interagieren Sie mit dem Objekt `HttpURLConnection` auf irgendeine Weise, nachdem der Upload abgeschlossen wurde. Im folgenden Beispiel werden diese Aufgaben umgesetzt, indem der HTTP-Antwortcode mit dem Objekt `HttpURLConnection` überprüft wird.

Example

In diesem Beispiel wird eine vorsignierte URL generiert, die zum Hochladen von Beispieldaten als Objekt verwendet wird. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.HttpMethod;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GeneratePresignedUrlRequest;
import com.amazonaws.services.s3.model.S3Object;

import java.io.IOException;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.URL;

public class GeneratePresignedUrlAndUploadObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String objectKey = "*** Object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Set the pre-signed URL to expire after one hour.
            java.util.Date expiration = new java.util.Date();
            long expTimeMillis = expiration.getTime();
            expTimeMillis += 1000 * 60 * 60;
            expiration.setTime(expTimeMillis);

            // Generate the pre-signed URL.
            System.out.println("Generating pre-signed URL.");
            GeneratePresignedUrlRequest generatePresignedUrlRequest = new
 GeneratePresignedUrlRequest(bucketName, objectKey)
                    .withMethod(HttpMethod.PUT)
                    .withExpiration(expiration);
            URL url = s3Client.generatePresignedUrl(generatePresignedUrlRequest);

            // Create the connection and use it to upload the new object using the pre-
signed URL.
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestMethod("PUT");
            OutputStreamWriter out = new OutputStreamWriter(connection.getOutputStream());
            out.write("This text uploaded as an object via presigned URL.");
            out.close();

            // Check the HTTP response code. To complete the upload and make the object
 available,
            // you must interact with the connection object in some way.
            connection.getResponseCode();
            System.out.println("HTTP response code: " + connection.getResponseCode());

            // Check to make sure that the object was uploaded successfully.
            S3Object object = s3Client.getObject(bucketName, objectKey);
            System.out.println("Object " + object.getKey() + " created in bucket " +
 object.getBucketName());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Hochladen eines Objekts unter Verwendung einer vorsignierten URL (AWS SDK für .NET)

Das folgende C#-Beispiel zeigt, wie Sie den AWS SDK für .NET verwenden, um ein Objekt unter Verwendung einer vorsignierten URL zu einem S3-Bucket hochzuladen. Weitere Informationen zu vorsignierten URLs finden Sie unter Hochladen von Objekten unter Verwendung vorsignierter URLs (p. 228).

In diesem Beispiel wird eine vorsignierte URL für ein bestimmtes Objekt generiert und zum Hochladen einer Datei verwendet. Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK für .NET und Anleitungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.IO;
using System.Net;

namespace Amazon.DocSamples.S3
{
    class UploadObjectUsingPresignedURLTest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string objectKey  = "*** provide the name for the uploaded object
 ***";
        private const string filePath   = "*** provide the full path name of the file to
 upload ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            var url = GeneratePreSignedURL();
            UploadObject(url);
        }

        private static void UploadObject(string url)
        {
            HttpWebRequest httpRequest = WebRequest.Create(url) as HttpWebRequest;
            httpRequest.Method = "PUT";
            using (Stream dataStream = httpRequest.GetRequestStream())
            {
                var buffer = new byte[8000];
                using (FileStream fileStream = new FileStream(filePath, FileMode.Open,
 FileAccess.Read))
                {
                    int bytesRead = 0;
                    while ((bytesRead = fileStream.Read(buffer, 0, buffer.Length)) > 0)
                    {
                        dataStream.Write(buffer, 0, bytesRead);
                    }
                }
```

```
            }
            HttpWebResponse response = httpRequest.GetResponse() as HttpWebResponse;
        }

        private static string GeneratePreSignedURL()
        {
            var request = new GetPreSignedUrlRequest
            {
                BucketName = bucketName,
                Key        = objectKey,
                Verb       = HttpVerb.PUT,
                Expires    = DateTime.Now.AddMinutes(5)
            };

           string url = s3Client.GetPreSignedURL(request);
           return url;
        }
    }
}
```

Weitere Infos

AWS SDK für .NET

Hochladen eines Objekts unter Verwendung einer vorsignierten URL (AWS SDK für Ruby)

Die folgenden Aufgaben führen Sie durch die Verwendung eines Ruby-Skripts für das Hochladen eines Objekts unter Verwendung einer vorsignierten URL für SDK für Ruby Version 3.

Hochladen von Objekten – SDK für Ruby – Version 3

1	Erstellen Sie eine Instance der `Aws::S3::Resource`-Klasse.
2	Geben Sie einen Bucket-Namen und einen Objektschlüssel an, indem Sie die `#bucket[]`- und `#object[]`-Methoden Ihrer `Aws::S3::Resource`-Klassen-Instance aufrufen. Generieren Sie eine vorsignierte URL, indem Sie eine Instance der Klasse `URI` erstellen und mit dieser die Methode `.presigned_url` Ihrer Instance der Klasse `Aws::S3::Resource` parsen. Sie müssen `:put` als Argument für `.presigned_url` angeben, und Sie müssen `PUT` für `Net::HTTP::Session#send_request` verwenden, wenn Sie ein Objekt hochladen wollen.
3	Jeder Benutzer, der die vorsignierte URL besitzt, kann ein Objekt hochladen. Der Upload erstellt ein Objekt oder ersetzt ein vorhandenes Objekt mit demselben Schlüssel, der in der vorsignierten URL angegeben ist.

Im folgenden Ruby-Codebeispiel werden die vorherigen Aufgaben für SDK für Ruby – Version 3 veranschaulicht.

Example

```
# Upload an object using a presigned URL for SDK for Ruby - Version 3.

require 'aws-sdk-s3'
require 'net/http'

s3 = Aws::S3::Resource.new(region: 'us-west-2')
```

```
# Replace BucketName with the name of your bucket.
# Replace KeyName with the name of the object you are creating or replacing.
obj = s3.bucket('BucketName').object('KeyName')

url = URI.parse(obj.presigned_url(:put))

# The contents of your object, as a string
body = 'Hello World!'

Net::HTTP.start(url.host) do |http|
  http.send_request('PUT', url.request_uri, body,
                    # Or else Net::HTTP adds a default, unsigned content-type
                    'content-type' => '')
end

# Print the contents of your object to the terminal window
puts obj.get.body.read
```

Objekte kopieren

Die Kopieroperation erzeugt eine Kopie eines Objekts, das bereits in Amazon S3 gespeichert ist. Sie können eine Kopie Ihres Objekts mit einer Größe von bis zu 5 GB in einer einzigen atomaren Operation hochladen. Für das Kopieren von Objekten mit mehr als 5 GB müssen Sie jedoch die API für mehrteilige Uploads verwenden. Mit der `copy`-Operation können Sie:

- Zusätzliche Kopien von Objekten erstellen
- Objekte umbenennen, indem Sie sie kopieren und die Originalobjekte löschen
- Objekte über Amazon S3-Standorte verschieben (z. B. us-west-1 und Europa)
- Objektmetadaten ändern

  Jedes Amazon S3-Objekt hat Metadaten. Dies sind Name/Wert-Paare. Sie können Objekt-Metadaten beim Hochladen festlegen. Nachdem Sie das Objekt hochgeladen haben, können Sie Objekt-Metadaten nicht mehr ändern. Die einzige Methode, wie Sie Objekt-Metadaten ändern können, ist es, eine Kopie des Objekts anzulegen und die Metadaten festzulegen. In der Kopieroperation geben Sie dasselbe Objekt als Quelle und Ziel an.

Jedes Objekt hat Metadaten. Einige davon sind Systemmetadaten, andere sind benutzerdefiniert. Benutzer steuern einige der Systemmetadaten, wie beispielsweise die Speicherklassenkonfiguration, die für das Objekt verwendet wird, und die Konfiguration der serverseitigen Verschlüsselung. Wenn Sie ein Objekt kopieren, werden auch die Systemmetadaten und benutzerdefinierte Metadaten kopiert. Amazon S3 setzt die systemgesteuerten Metadaten zurück. Wenn Sie beispielsweise ein Objekt kopieren, setzt Amazon S3 das Erstellungsdatum des kopierten Objekts zurück. Sie brauchen keine dieser Werte in Ihrer Kopieranforderung festlegen.

Wenn Sie ein Objekt kopieren, wollen Sie möglicherweise einige der Metadatenwerte aktualisieren. Wenn Ihr Quellobjekt beispielsweise für die Verwendung des Standardspeichers konfiguriert ist, könnten Sie entscheiden, für die Objektkopie RRS zu verwenden. Außerdem könnten Sie entscheiden, einige der benutzerdefinierten Metadatenwerte für das Quellobjekt zu aktualisieren. Wenn Sie entschieden haben, beim Kopieren vom Benutzer des Objekts konfigurierbare Metadaten zu aktualisieren (System oder benutzerdefiniert), müssen Sie explizit alle vom Benutzer konfigurierbaren Metadaten angeben, die im Quellobjekt Ihrer Anforderung vorhanden sind, selbst wenn Sie nur einen der Metadatenwerte ändern.

Weitere Informationen zu Objekt-Metadaten erhalten Sie unter Objektschlüssel und Metadaten (p. 117).

Note

- Beim Kopieren von Objekten über verschiedene Standorte fallen Bandbreitengebühren an.

- Wenn das Quellobjekt in `S3 Glacier` oder `S3 Glacier Deep Archive` archiviert ist, müssen Sie zuerst eine temporäre Kopie wiederherstellen, bevor Sie das Objekt in einen anderen Bucket kopieren können. Weitere Informationen über das Archivieren von Objekten finden Sie unter Übergang zu den Speicherklassen S3 Glacier und S3 Glacier Deep Archive (Objektarchivierung) (p. 144).

Beim Kopieren von Objekten können Sie Amazon S3 anweisen, das Zielobjekt mit einem AWS Key Management Service (AWS KMS)-Kundenhauptschlüssel (CMK), einem von Amazon S3 verwalteten Verschlüsselungsschlüssel oder einem vom Kunden bereitgestellten Verschlüsselungsschlüssel zu speichern. Dementsprechend müssen Sie Verschlüsselungsinformationen in Ihrer Anforderung angeben. Wenn es sich bei der Kopierquelle um ein Objekt handelt, das unter Amazon S3 mit serverseitiger Verschlüsselung mit vom Kunden bereitgestelltem Schlüssel gespeichert ist, müssen Sie in Ihrer Anfrage Verschlüsselungsinformationen angeben, damit Amazon S3 das Objekt zum Kopieren entschlüsseln kann. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung (p. 290).

Zum Kopieren von mehreren Amazon S3-Objekten mit einer einzelnen Anforderung können Sie Amazon S3-Stapelvorgänge verwenden. Sie stellen S3 Stapeloperationen eine Liste der Objekte zur Verfügung, für die Operationen aufgeführt werden sollen. S3 Stapeloperationen rufen die entsprechende API auf, um die angegebene Operation auszuführen. Eine einzelne S3 Stapeloperationen-Aufgabe kann die festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen.

S3 Stapeloperationen-Stapeloperationen verfolgen den Fortschritt nach, senden Benachrichtigungen und speichern einen detaillierten Abschlussbericht zu allen Aktionen. So erhalten Sie eine vollständig verwaltete, prüfbare und serverlose Umgebung. Sie können S3 Stapeloperationen über die AWS Management Console, die AWS CLI, die AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter the section called “Die Grundlagen: Aufträge” (p. 553).

Verwandte Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Kopieren eines Objekts in einer einzigen Operation

Die Beispiele in diesem Abschnitt zeigen, wie Objekte mit bis zu 5 GB in einer einzigen Operation kopiert werden können. Für das Kopieren von Objekten mit mehr als 5 GB müssen Sie die API für mehrteilige Uploads verwenden. Weitere Informationen finden Sie unter Kopieren von Objekten unter Verwendung der API für mehrteilige Uploads (p. 239).

Themen

Kopieren eines Objekts mit AWS SDK for Java

Example

Im folgenden Beispiel wird gezeigt, wie Sie ein Objekt mit AWS SDK for Java zu Amazon S3 kopieren. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

import java.io.IOException;

public class CopyObjectSingleOperation {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String sourceKey = "*** Source object key *** ";
        String destinationKey = "*** Destination object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjRequest = new CopyObjectRequest(bucketName, sourceKey,
 bucketName, destinationKey);
            s3Client.copyObject(copyObjRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Kopieren eines Amazon S3-Objekts in einer einzigen Operation unter Verwendung von AWS SDK für .NET

Im folgenden C#-Beispiel wird gezeigt, wie Sie mithilfe des High-Level-AWS SDK für .NET Objekte bis zu einer Größe von 5 GB in einer einzigen Operation kopieren. Für Objekte mit mehr als 5 GB müssen Sie das in Kopieren eines Amazon S3-Objekts mit der AWS SDK für .NET-API für mehrteilige Uploads (p. 241) beschriebene Kopierbeispiel für mehrteilige Uploads verwenden.

In diesem Beispiel wird eine Kopie eines Objekts bis zu einer Größe von 5 GB erstellt. Weitere Informationen zur Kompatibilität des Beispiels mit einer bestimmten Version von AWS SDK for .NET sowie Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels, finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
```

```
{
    class CopyObjectTest
    {
        private const string sourceBucket = "*** provide the name of the bucket with source
 object ***";
        private const string destinationBucket = "*** provide the name of the bucket to
 copy the object to ***";
        private const string objectKey = "*** provide the name of object to copy ***";
        private const string destObjectKey = "*** provide the destination object key name
 ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Copying an object");
            CopyingObjectAsync().Wait();
        }

        private static async Task CopyingObjectAsync()
        {
            try
            {
                CopyObjectRequest request = new CopyObjectRequest
                {
                    SourceBucket = sourceBucket,
                    SourceKey = objectKey,
                    DestinationBucket = destinationBucket,
                    DestinationKey = destObjectKey
                };
                CopyObjectResponse response = await s3Client.CopyObjectAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Weitere Infos

AWS SDK für .NET

Kopieren eines Objekts mit AWS SDK für PHP

Dieses Thema führt Sie durch die Verwendung der AWS SDK für PHP-Klassen aus Version 3 für das Kopieren eines einzelnen Objekts sowie mehrerer Objekte in Amazon S3 aus einem Bucket in einen anderen oder innerhalb desselben Buckets.

Dieser Abschnitt setzt voraus, dass Sie den Anweisungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und AWS SDK für PHP ordnungsgemäß installiert ist.

Die folgenden Aufgaben führen Sie durch die Verwendung von PHP SDK-Klassen für das Kopieren eines Objekts, das bereits in Amazon S3 gespeichert ist.

Die folgenden Aufgaben führen Sie bei der Verwendung von PHP-Klassen, um mehrere Kopien eines Objekts innerhalb von Amazon S3 zu erstellen.

Objekte kopieren

1	Erstellen Sie mit dem Klassenkonstruktor `Aws\S3\S3Client` eine Instance eines Amazon S3-Clients.
2	Um mehrere Kopien eines Objekts anzulegen, führen Sie einen Aufrufstapel für die getCommand()-Methode des Amazon S3-Clients aus, die von der Klasse Aws \CommandInterface übernommen wird. Sie stellen den `CopyObject`-Befehl als erstes Argument, sowie ein Array mit dem Quell-Bucket, dem Quellschlüsselnamen, dem Ziel-Bucket und dem Zielschlüsselnamen als zweites Argument bereit.

Example Objekte kopieren in Amazon S3

Das folgende PHP-Beispiel verdeutlicht die Verwendung der `copyObject()`-Methode zum Kopieren eines einzelnen Objekts in Amazon S3 sowie die Verwendung eines Aufrufstapels von `CopyObject` mit der `getcommand()`-Methode, um mehrere Kopien eines Objekts zu erstellen.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$sourceBucket = '*** Your Source Bucket Name ***';
$sourceKeyname = '*** Your Source Object Key ***';
$targetBucket = '*** Your Target Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Copy an object.
$s3->copyObject([
    'Bucket'     => $targetBucket,
    'Key'        => "{$sourceKeyname}-copy",
    'CopySource' => "{$sourceBucket}/{$sourceKeyname}",
]);

// Perform a batch of CopyObject operations.
$batch = array();
for ($i = 1; $i <= 3; $i++) {
    $batch[] = $s3->getCommand('CopyObject', [
        'Bucket'     => $targetBucket,
        'Key'        => "{targetKeyname}-{$i}",
        'CopySource' => "{$sourceBucket}/{$sourceKeyname}",
    ]);
}
try {
    $results = CommandPool::batch($s3, $batch);
    foreach($results as $result) {
        if ($result instanceof ResultInterface) {
            // Result handling here
        }
        if ($result instanceof AwsException) {
            // AwsException handling here
        }
    }
} catch (\Exception $e) {
    // General error handling here
```

```
}
```

Verwandte Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Kopieren eines Objekts mit AWS SDK für Ruby

Die folgenden Aufgaben führen Sie durch die Verwendung der Ruby-Klassen für das Kopieren eines Objekts in Amazon S3 aus einem Bucket in einen anderen oder für das Kopieren eines Objekts innerhalb desselben Buckets.

Objekte kopieren

1	Verwenden Sie das modularisierte Amazon S3-Gem für Version 3 von AWS SDK für Ruby, fordern Sie „aws-sdk-s3“ an, und geben Sie Ihre AWS-Anmeldeinformationen an. Weitere Informationen zur Bereitstellung Ihrer Anmeldeinformationen finden Sie in Senden von Anfragen unter Verwendung von AWS-Konto- oder IAM-Benutzeranmeldeinformationen (p. 20).
2	Stellen Sie müssen die Anforderungsinformationen bereit, wie beispielsweise den Bucket-Namen, den Quellschlüsselnamen, den Ziel-Bucket-Namen und den Zielschlüssel.

Das folgende Ruby-Codebeispiel demonstriert die oben beschriebenen Aufgaben unter Verwendung der `#copy_object`-Methode zum Kopieren eines Objekts aus einem Bucket in einen anderen.

Example

```
require 'aws-sdk-s3'

source_bucket_name = '*** Provide bucket name ***'
target_bucket_name = '*** Provide bucket name ***'
source_key = '*** Provide source key ***'
target_key = '*** Provide target key ***'

begin
  s3 = Aws::S3::Client.new(region: 'us-west-2')
  s3.copy_object(bucket: target_bucket_name, copy_source: source_bucket_name + '/' +
 source_key, key: target_key)
rescue StandardError => ex
  puts 'Caught exception copying object ' + source_key + ' from bucket ' +
 source_bucket_name + ' to bucket ' + target_bucket_name + ' as ' + target_key + ':'
  puts ex.message
end

puts 'Copied ' +  source_key + ' from bucket ' + source_bucket_name + ' to bucket ' +
 target_bucket_name + ' as ' + target_key
```

Kopieren eines Objekts mit der REST API

Dieses Beispiel beschreibt, wie ein Objekt mit REST kopiert wird. Weitere Informationen über die REST API finden Sie unter PUT Object (Copy).

Dieses Beispiel kopiert das `flotsam`-Objekt aus dem `pacific`-Bucket in das `jetsam`-Objekt des `atlantic`-Buckets, wobei seine Metadaten beibehalten werden.

```
PUT /jetsam HTTP/1.1
Host: atlantic.s3.amazonaws.com
x-amz-copy-source: /pacific/flotsam
Authorization: AWS AKIAIOSFODNN7EXAMPLE:ENoSbxYByFA0UGLZUqJN5EUnLDg=
Date: Wed, 20 Feb 2008 22:12:21 +0000
```

Die Signatur wurde aus den folgenden Informationen generiert.

```
PUT\r\n
\r\n
\r\n
Wed, 20 Feb 2008 22:12:21 +0000\r\n

x-amz-copy-source:/pacific/flotsam\r\n
/atlantic/jetsam
```

Amazon S3 gibt die folgende Antwort zurück, die das ETag des Objekts angibt, und wann es zuletzt geändert wurde.

```
HTTP/1.1 200 OK
x-amz-id-2: Vyaxt7qEbzv34BnSu5hctyyNSlHTYZFMWK4FtzO+iX8JQNyaLdTshL0KxatbaOZt
x-amz-request-id: 6B13C3C5B34AF333
Date: Wed, 20 Feb 2008 22:13:01 +0000

Content-Type: application/xml
Transfer-Encoding: chunked
Connection: close
Server: AmazonS3
<?xml version="1.0" encoding="UTF-8"?>

<CopyObjectResult>
   <LastModified>2008-02-20T22:13:01</LastModified>
   <ETag>"7e9c608af58950deeb370c98608ed097"</ETag>
</CopyObjectResult>
```

Kopieren von Objekten unter Verwendung der API für mehrteilige Uploads

Die Beispiele in diesem Abschnitt zeigen Ihnen, wie Sie Objekte mit mehr als 5 GB mit Hilfe des API für mehrteilige Uploads hochladen. Sie können in einer einzelnen Operation Objekte bis zu einer Größe von 5 GB hochladen. Weitere Informationen finden Sie unter Kopieren eines Objekts in einer einzigen Operation (p. 234).

Themen

Kopieren eines Objekts mit der AWS SDK for Java-API für mehrteilige Uploads

Um ein Amazon S3-Objekt, das größer als 5 GB ist, mit AWS SDK for Java zu kopieren, verwenden Sie die Low-Level-Java-API. Für Objekte, die kleiner als 5 GB sind, verwenden Sie das Kopieren in einer einzelnen Operation, wie in Kopieren eines Objekts mit AWS SDK for Java (p. 234) beschrieben.

Gehen Sie zum Kopieren eines Objekts mit der Low-Level-Java-API wie folgt vor:

- Initiieren eines mehrteiligen Uploads durch Ausführung der Methode `AmazonS3Client.initiateMultipartUpload()`.
- Speichern Sie die Upload-ID aus dem Antwortobjekt, das die Methode `AmazonS3Client.initiateMultipartUpload()` zurückgibt. Sie geben diese Upload-ID bei jeder Teiloperation mehrteiliger Uploads an.
- Kopieren Sie alle Teile. Erstellen Sie für jeden Teil, den Sie kopieren müssen, eine neue Instance der Klasse `CopyPartRequest`. Geben Sie die teilspezifischen Informationen an, darunter Quell- und Zielbucket-Namen, Quell- und Ziel-Objektschlüssel, Upload-ID, Stelle des ersten und des letzten Byte des Teils sowie die Nummer des Teils.
- Speichern Sie die Antworten der `AmazonS3Client.copyPart()`-Methodenaufrufe. Jede Antwort enthält den Wert für `ETag` und die Teilenummer des hochgeladenen Teils. Sie benötigen diese Informationen, um den mehrteiligen Upload abzuschließen.
- Rufen Sie die Methode `AmazonS3Client.completeMultipartUpload()` auf, um die Kopieroperation abzuschließen.

Example

Das folgende Beispiel veranschaulicht, wie Sie mit der Amazon S3-Low-Level-Java-API eine mehrteilige Kopie ausführen können. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class LowLevelMultipartCopy {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String sourceBucketName = "*** Source bucket name ***";
        String sourceObjectKey = "*** Source object key ***";
        String destBucketName = "*** Target bucket name ***";
        String destObjectKey = "*** Target object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(destBucketName, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(sourceBucketName, sourceObjectKey);
            ObjectMetadata metadataResult = s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();
```

```
            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(sourceBucketName)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(destBucketName)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make the
 copied object available.
            CompleteMultipartUploadRequest completeRequest = new
 CompleteMultipartUploadRequest(
                    destBucketName,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

Kopieren eines Amazon S3-Objekts mit der AWS SDK für .NET-API für mehrteilige Uploads

Das folgende C#-Beispiel veranschaulicht, wie Sie mit AWS SDK für .NET ein Amazon S3-Objekt, das größer als 5 GB ist, von einem Quellspeicherort an einen anderen Speicherort kopieren, beispielsweise aus einem Bucket in einen anderen. Um Objekte zu kopieren, die kleiner als 5 GB sind, verwenden Sie das Kopierverfahren in einer einzigen Operation, das unter Kopieren eines Amazon S3-Objekts

in einer einzigen Operation unter Verwendung von AWS SDK für .NET (p. 235) beschrieben wird. Weitere Informationen über mehrteilige Uploads bei Amazon S3 finden Sie unter Überblick Multipart Upload (p. 199).

Dieses Beispiel veranschaulicht, wie Sie ein Amazon S3-Objekt, das größer als 5 GB ist, mit der AWS SDK für .NET-API für mehrteilige Uploads von einem S3-Bucket in einen anderen kopieren. Weitere Informationen zur SDK-Kompatibilität und Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CopyObjectUsingMPUapiTest
    {
        private const string sourceBucket = "*** provide the name of the bucket with source
 object ***";
        private const string targetBucket = "*** provide the name of the bucket to copy the
 object to ***";
        private const string sourceObjectKey = "*** provide the name of object to copy
 ***";
        private const string targetObjectKey = "*** provide the name of the object copy
 ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Copying an object");
            MPUCopyObjectAsync().Wait();
        }
        private static async Task MPUCopyObjectAsync()
        {
            // Create a list to store the upload part responses.
            List<UploadPartResponse> uploadResponses = new List<UploadPartResponse>();
            List<CopyPartResponse> copyResponses = new List<CopyPartResponse>();

            // Setup information required to initiate the multipart upload.
            InitiateMultipartUploadRequest initiateRequest =
                new InitiateMultipartUploadRequest
                {
                    BucketName = targetBucket,
                    Key = targetObjectKey
                };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // Save the upload ID.
            String uploadId = initResponse.UploadId;

            try
            {
                // Get the size of the object.
                GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest
                {
```

```
                    BucketName = sourceBucket,
                    Key = sourceObjectKey
                };

                GetObjectMetadataResponse metadataResponse =
                    await s3Client.GetObjectMetadataAsync(metadataRequest);
                long objectSize = metadataResponse.ContentLength; // Length in bytes.

                // Copy the parts.
                long partSize = 5 * (long)Math.Pow(2, 20); // Part size is 5 MB.

                long bytePosition = 0;
                for (int i = 1; bytePosition < objectSize; i++)
                {
                    CopyPartRequest copyRequest = new CopyPartRequest
                    {
                        DestinationBucket = targetBucket,
                        DestinationKey = targetObjectKey,
                        SourceBucket = sourceBucket,
                        SourceKey = sourceObjectKey,
                        UploadId = uploadId,
                        FirstByte = bytePosition,
                        LastByte = bytePosition + partSize - 1 >= objectSize ? objectSize -
1 : bytePosition + partSize - 1,
                        PartNumber = i
                    };

                    copyResponses.Add(await s3Client.CopyPartAsync(copyRequest));

                    bytePosition += partSize;
                }

                // Set up to complete the copy.
                CompleteMultipartUploadRequest completeRequest =
                new CompleteMultipartUploadRequest
                {
                    BucketName = targetBucket,
                    Key = targetObjectKey,
                    UploadId = initResponse.UploadId
                };
                completeRequest.AddPartETags(copyResponses);

                // Complete the copy.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }
    }
}
```

Weitere Infos

AWS SDK für .NET

Kopieren von Objekten unter Verwendung der REST API für mehrteilige Uploads

Die folgenden Abschnitte in der Amazon Simple Storage Service API Reference beschreiben die REST-API für mehrteilige Uploads. Zum Kopieren eines vorhandenen Objekts verwenden Sie die Upload Part (Copy) API. Sie geben das Quellobjekt an, indem Sie den Anfrage-Header `x-amz-copy-source` in Ihre Anfrage aufnehmen.

- Multipart-Upload initiieren
- Upload Part
- Teil hochladen (kopieren)
- Abschließen eines mehrteiligen Uploads
- Abort Multipart Upload
- Teile auflisten
- List Multipart Uploads

Sie können diese APIs verwenden, um Ihre eigenen REST-Anfragen zu erstellen, oder Sie können eines der von uns bereitgestellten SDKs verwenden. Weitere Informationen zu den SDKs finden Sie unter API-Unterstützung für mehrteilige Uploads (p. 203).

Auflisten von Objektschlüsseln

Schlüssel können nicht nach dem Präfix aufgelistet werden. Durch die Auswahl eines gemeinsamen Präfix für die Namen verwandter Schlüssel und die Markierung dieser Schlüssel mit einem Sonderzeichen, das die Hierarchie begrenzt, können Sie die Listenoperation verwenden, um Schlüssel hierarchisch auszuwählen und zu durchsuchen. In ähnlicher Weise werden Dateien in einem Dateisystem in Ordnern gespeichert.

Amazon S3 stellt eine Listenoperation bereit, die Ihnen ermöglicht, die in einem Bucket enthaltenen Schlüssel aufzulisten. Schlüssel werden nach Bucket und Präfix für die Auflistung ausgewählt. Angenommen, wir haben einen Bucket namens "dictionary", der einen Schlüssel für jedes englische Wort enthält. Sie können einen Aufruf ausführen, um alle Schlüssel in dem Bucket aufzulisten, die mit dem Buchstaben "q" beginnen. Listenergebnisse werden immer in binärer UTF-8-Reihenfolge zurückgegeben.

Sowohl SOAP- als auch REST-Listenoperationen geben ein XML-Dokument zurück, das die Namen übereinstimmender Schlüssel sowie Informationen über das von jedem Schlüssel identifizierte Objekt enthält.

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Gruppen von Schlüsseln, die ein gemeinsames Präfix haben, abgeschlossen mit einem speziellen Trennzeichen, können in einer Auflistung nach diesem gemeinsamen Präfix ausgegeben werden. Auf diese Weise können Anwendungen ihre Schlüssel hierarchisch organisieren und durchsuchen, ähnlich wie bei der Organisation Ihrer Dateien in Verzeichnissen in einem Dateisystem. Um beispielsweise den dictionary-Bucket zu erweitern, um mehr als nur englische Wörter aufzunehmen, könnten Sie Schlüssel bilden, indem Sie jedem Wort ein Präfix für seine Sprache und ein Trennzeichen voranstellen, wie beispielsweise "French/logical". Mit Hilfe dieses Namensschemas und der Funktion zur hierarchischen Auflistung könnten Sie eine Liste nur mit französischen Wörtern abrufen. Sie könnten auch auf oberster Ebene eine Liste verfügbarer Sprachen durchsuchen, ohne alle lexikographischen Schlüssel durchlaufen zu müssen.

Weitere Informationen über diesen Aspekte der Auflistung finden Sie unter Hierarchische Auflistung von Schlüsseln mit einem Präfix und einem Trennzeichen (p. 245).

Effiziente Implementierung von Listen

Die Leistung der Auflistung wird von der Gesamtzahl der Schlüssel in Ihrem Bucket nicht maßgeblich beeinträchtigt, ebenso wenig wie durch das Vorhandensein oder Fehlen der Argumente für Präfix, Markierung, maximale Schlüsselanzahl oder Trennzeichen.

Durchlaufen mehrseitiger Ergebnisse

Buckets können nahezu unbegrenzt viele Schlüssel enthalten, die vollständigen Ergebnisse einer Listenabfrage können deshalb extrem umfangreich sein. Um große Ergebnismengen zu verwalten, unterstützt die Amazon S3-API eine Paginierung, um sie in mehrere Antworten aufzuteilen. Jede Antwort für Listenschlüssel gibt eine Seite mit bis zu 1000 Schlüsseln zurück, mit einer Angabe, ob die Antwort gekürzt wurde. Sie senden immer wieder Listenschlüsselanforderungen, bis Sie alle Schlüssel erhalten haben. AWS SDK Wrapper-Bibliotheken unterstützen dieselbe Paginierung.

Die folgenden Java- und .NET SDK-Beispiele zeigen, wie die Paginierung bei der Auflistung von Schlüsseln in einem Bucket verwendet wird.

- Auflisten von Schlüsseln mit AWS SDK for Java (p. 246)
- Auflisten von Schlüsseln mit AWS SDK für .NET (p. 247)
- Auflisten von Schlüsseln mit AWS SDK für PHP (p. 249)

Verwandte Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Hierarchische Auflistung von Schlüsseln mit einem Präfix und einem Trennzeichen

Die Präfix- und Trennzeichen-Parameter begrenzen die von einer Listenoperation zurückgegebenen Ergebnisse. Das Präfix begrenzt die Ergebnisse auf die Schlüssel, die mit dem angegebenen Präfix beginnen. Das Trennzeichen veranlasst Listenoperationen, alle Schlüssel mit gemeinsamem Präfix in einer Zusammenfassungsliste als Ergebnis bereitzustellen.

Zweck der Präfix- und Trennzeichen-Parameter ist es, Ihnen dabei zu helfen, Ihre Schlüssel hierarchisch zu organisieren und dann zu durchsuchen. Dazu wählen Sie zuerst ein Trennzeichen für Ihren Bucket aus, wie beispielsweise einen Schrägstrich (/), das in Ihren Schlüsselnamen voraussichtlich nicht vorkommt. Anschließend konstruieren Sie Ihre Schlüsselnamen, indem Sie alle Ebenen der Hierarchie verknüpfen und jede Ebene mit dem Trennzeichen abtrennen.

Wenn Sie beispielsweise Informationen über Städte speichern, könnten Sie diese natürlich nach dem Kontinent, dann nach dem Land und dann nach der Provinz oder dem Staat anordnen. Diese Namen enthalten normalerweise keine Interpunktionszeichen, deshalb könnten Sie den Schrägstrich (/) als Trennzeichen verwenden. Die folgenden Beispiele verwenden einen Schrägstrich (/) als Trennzeichen.

- Europa/Frankreich/Nouvelle-Aquitaine/Bordeaux
- North America/Canada/Quebec/Montreal
- North America/USA/Washington/Bellevue
- North America/USA/Washington/Seattle

Wenn Sie auf diese Weise Daten für jede Stadt in der Welt gespeichert haben, ist es relativ mühsam, einen flachen Schlüsselnamensraum zu verwalten. Durch Verwendung von `Prefix` und `Delimiter` für die Listenoperation können Sie die Hierarchie verwenden, die Sie zum Auflisten Ihrer Daten erstellt haben. Um beispielsweise alle Staaten der USA aufzulisten, setzen Sie `Delimiter='/'` und `Prefix='North America/USA/'`. Um alle Provinzen in Kanada aufzulisten, für die Sie Daten haben, setzen sie `Delimiter='/'` und `Prefix='North America/Canada/'`.

Eine Listenanforderung mit Trennzeichen ermöglicht Ihnen, Ihrer Hierarchie auf nur einer Ebene zu durchlaufen, und die (möglicherweise Millionen) Schlüssel auf tieferen Ebenen zu überspringen und zusammenzufassen. Angenommen, Sie haben einen Bucket (`ExampleBucket`) mit den folgenden Schlüsseln.

`sample.jpg`

`photos/2006/January/sample.jpg`

`photos/2006/February/sample2.jpg`

`photos/2006/February/sample3.jpg`

`photos/2006/February/sample4.jpg`

Der Beispiel-Bucket hat nur das `sample.jpg`-Objekt auf der Root-Ebene. Um nur die Objekte auf Root-Ebene im Bucket aufzulisten, senden Sie eine GET-Anforderung mit dem Trennzeichen "/" an den Bucket. In der Antwort gibt Amazon S3; den `sample.jpg`-Objektschlüssel zurück, weil er das Trennzeichen „/“ nicht enthält. Alle anderen Schlüssel enthalten das Trennzeichen. Amazon S3 gruppiert diese Schlüssel und gibt ein einziges `CommonPrefixes`-Element mit dem Präfix-Wert `photos/` zurück, eine Unterzeichenfolge vom Anfang dieser Schlüssel bis zum Auftreten des angegebenen Trennzeichens.

Example

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>ExampleBucket</Name>
  <Prefix></Prefix>
  <Marker></Marker>
  <MaxKeys>1000</MaxKeys>
  <Delimiter>/</Delimiter>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>sample.jpg</Key>
    <LastModified>2011-07-24T19:39:30.000Z</LastModified>
    <ETag>&quot;d1a7fb5eab1c16cb4f7cf341cf188c3d&quot;</ETag>
    <Size>6</Size>
    <Owner>
      <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
      <DisplayName>displayname</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <CommonPrefixes>
    <Prefix>photos/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Auflisten von Schlüsseln mit AWS SDK for Java

Example

Das folgende Beispiel listet die Objektschlüssel in einem Bucket auf. Das Beispiel ruft einen Satz von Objektschlüsseln mit Paginierung ab. Wenn es nach der ersten Seite mehr zurückzugebende Schlüssel gibt, schließt Amazon S3 ein Fortsetzungs-Token in die Antwort ein. Das Beispiel verwendet das Fortsetzungs-Token in der nachfolgenden Anforderung, um den nächsten Satz von Objektschlüsseln abzurufen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsV2Request;
import com.amazonaws.services.s3.model.ListObjectsV2Result;
import com.amazonaws.services.s3.model.S3ObjectSummary;

import java.io.IOException;

public class ListKeys {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            System.out.println("Listing objects");

            // maxKeys is set to 2 to demonstrate the use of
            // ListObjectsV2Result.getNextContinuationToken()
            ListObjectsV2Request req = new
 ListObjectsV2Request().withBucketName(bucketName).withMaxKeys(2);
            ListObjectsV2Result result;

            do {
                result = s3Client.listObjectsV2(req);

                for (S3ObjectSummary objectSummary : result.getObjectSummaries()) {
                    System.out.printf(" - %s (size: %d)\n", objectSummary.getKey(),
 objectSummary.getSize());
                }
                // If there are more than maxKeys keys in the bucket, get a continuation
 token
                // and list the next objects.
                String token = result.getNextContinuationToken();
                System.out.println("Next Continuation Token: " + token);
                req.setContinuationToken(token);
            } while (result.isTruncated());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Auflisten von Schlüsseln mit AWS SDK für .NET

Example

Das folgende C#-Beispiel listet die Objektschlüssel für einen Bucket auf. Im Beispiel wird ein Satz von Objektschlüsseln mit Paginierung abgerufen. Wenn mehr zurückzugebende Schlüssel vorhanden sind,

schließt Amazon S3 ein Fortsetzungs-Token in die Antwort ein. Der Code verwendet das Fortsetzungs-Token in der nachfolgenden Anforderung, um den nächsten Satz von Objektschlüsseln abzurufen.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ListObjectsTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;

        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            ListingObjectsAsync().Wait();
        }

        static async Task ListingObjectsAsync()
        {
            try
            {
                ListObjectsV2Request request = new ListObjectsV2Request
                {
                    BucketName = bucketName,
                    MaxKeys = 10
                };
                ListObjectsV2Response response;
                do
                {
                    response = await client.ListObjectsV2Async(request);

                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }
                    Console.WriteLine("Next Continuation Token: {0}",
 response.NextContinuationToken);
                    request.ContinuationToken = response.NextContinuationToken;
                } while (response.IsTruncated);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("S3 error occurred. Exception: " +
 amazonS3Exception.ToString());
                Console.ReadKey();
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
                Console.ReadKey();
            }
        }
```

```
    }
}
```

Auflisten von Schlüsseln mit AWS SDK für PHP

Dieses Thema führt Sie durch die Verwendung der AWS SDK für PHP-Klassen aus Version 3 für das Auflisten der Objektschlüssel in einem Amazon S3-Bucket.

Dieser Abschnitt setzt voraus, dass Sie den Anweisungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und AWS SDK für PHP ordnungsgemäß installiert ist.

Um die in einem Bucket aufgelisteten Objektschlüssel mit AWS SDK für PHP aufzulisten, müssen Sie zuerst die im Bucket enthaltenen Objekte auflisten und dann die Schlüssel der einzelnen Objekte extrahieren. Für das Auflisten der Objekte in einem Bucket haben Sie die Möglichkeit, die Low Level Aws\S3\S3Client::listObjects()-Methode oder die High-Level Aws\ResultPaginator-Klasse zu verwenden.

Die Low-Level `listObjects()`-Methode wird auf die zugrunde liegende Amazon S3 REST API abgebildet. Jeder `listObjects()`-Anforderung gibt eine Seite mit bis zu 1000 Objekten zurück. Wenn Sie mehr als 1000 Objekte im Bucket haben, wird Ihre Antwort gekürzt und Sie müssen eine weitere `listObjects()`-Anforderung senden, um die nächsten 1000 Objekte abzurufen.

Sie können sich mit dem High-Level `ListObjects`-Paginator die Aufgabe, die in einem Bucket enthaltenen Objekte aufzulisten, etwas einfacher machen. Um mit dem `ListObjects`-Paginator eine Liste mit Objekten zu erstellen, führen Sie die getPaginator()-Methode des Amazon S3-Clients aus, die von der Klasse Aws/AwsClientInterface übernommen wird. Dabei ist der `ListObjects`-Befehl das erste Argument, und ein Array, das die von dem angegebenen Bucket zurückgegebenen Objekte aufnimmt, ist das zweite Argument. Bei Verwendung als `ListObjects`-Paginator gibt die Methode `getPaginator()` alle in dem jeweiligen Bucket enthaltenen Objekte zurück. Es gibt keine Begrenzung auf 1000 Objekte, Sie müssen sich also keine Gedanken darüber machen, ob die Antwort gekürzt ist oder nicht.

Die folgenden Aufgaben führen Sie durch die Verwendung von PHP Amazon S3-Client-Methoden für das Auflisten der Objekte aus einem Bucket, für die Sie die Objektschlüssel auflisten können.

Example Auflisten von Objektschlüsseln

Das folgende PHP-Beispiel demonstriert, wie Sie die Schlüssel aus einem angegebenen Bucket auflisten können. Es zeigt, wie Sie die High-Level `getIterator()`-Methode anwenden, um die Objekte in einem Bucket aufzulisten, und dann, wie der Schlüssel aus jedem Objekt in der Liste extrahiert wird. Es zeigt außerdem, wie Sie die Low-Level `listObjects()`-Methode anwenden, um die Objekte in einem Bucket aufzulisten, und dann, wie der Schlüssel aus jedem Objekt in der zurückgegebenen Liste extrahiert wird. Weitere Informationen über die Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';

// Instantiate the client.
$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Use the high-level iterators (returns ALL of your objects).
```

```
try {
    $results = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    foreach ($results as $result) {
        foreach ($result['Contents'] as $object) {
            echo $object['Key'] . PHP_EOL;
        }
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}

// Use the plain API (returns ONLY up to 1000 of your objects).
try {
    $objects = $s3->listObjects([
        'Bucket' => $bucket
    ]);
    foreach ($objects['Contents']  as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Verwandte Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- Paginatoren
- AWS SDK für PHP-Dokumentation

Auflisten von Schlüsseln mit der REST API

Sie können das AWS SDK verwenden, um die Objektschlüssel in einem Bucket aufzulisten. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Sie können eine GET-Anforderung senden, um alle oder einen Teil der Objekte in einem Bucket zurückzugeben, oder Auswahlkriterien anwenden, um eine Untermenge der Objekte in einem Bucket zurückzugeben. Weitere Informationen dazu erhalten Sie unter GET Bucket (List Objects) Version 2.

Löschen von Objekten

Themen

Sie können ein oder mehrere Objekte direkt aus Amazon S3 löschen. Beim Löschen eines Objekts haben Sie die folgenden Optionen:

- Löschen eines einzelnen Objekts – Amazon S3 stellt die DELETE API bereit, die Sie zum Löschen eines einzelnen Objekts in einer einzelnen HTTP-Anfrage verwenden können.
- Löschen mehrerer Objekte – Amazon S3 stellt außerdem die Multi-Object Delete API bereit, die Sie zum Löschen von bis zu 1000 Objekten in einer einzelnen HTTP-Anfrage verwenden können.

Wenn Sie Objekte aus einem nicht versionsfähigen Bucket löschen, stellen Sie nur den Namen des Objektschlüssels bereit. Wenn Sie Objekte aus einem versionsfähigen Bucket löschen, können Sie optional die Versions-ID des Objekts angeben, um eine spezifische Version des Objekts zu löschen.

Löschen von Objekten aus einem versionsfähigen Bucket

Wenn Ihr Bucket versionsfähig ist, kann es innerhalb des Buckets mehrere Versionen desselben Objekts geben. Bei der Arbeit mit versionsfähigen Buckets unterstützt die API zum Löschen die folgenden Optionen:

- Angeben einer nicht versionierten Löschanfrage – Hier geben Sie ausschließlich den Schlüssel des Objekts und keine Versions-ID ein. In diesem Fall erstellt Amazon S3 eine Löschmarkierung und gibt in der Antwort ihre Versions-ID zurück. Dadurch verschwindet Ihr Objekt aus dem Bucket. Weitere Informationen zum Objekt-Versioning und zum Konzept der Löschmarkierung finden Sie unter Objekt-Versioning (p. 127).
- Angeben einer versionierten Löschanfrage – Hier geben Sie sowohl den Schlüssel als auch eine Versions-ID an. In diesem Fall sind zwei Ergebnisse möglich:
  - Wenn die Versions-ID einer spezifischen Objektversion zugeordnet ist, löscht Amazon S3 die spezifische Version des Objekts.
  - Wenn die Versions-ID der Löschmarkierung dieses Objekts zugeordnet ist, löscht Amazon S3 die Löschmarkierung. Dadurch wird Ihr Objekt wieder in Ihrem Bucket angezeigt.

Löschen von Objekten aus einem MFA-fähigen Bucket

Beachten Sie beim Löschen von Objekten aus einem MFA-fähigen Bucket (Multi-Factor Authentication) Folgendes:

- Wenn Sie ein ungültiges MFA-Token bereitstellen, schlägt die Anfrage immer fehl.
- Wenn Sie einen MFA-fähigen Bucket haben und eine versionsfähige Löschanfrage erstellen (d. h. Sie geben einen Objektschlüssel und eine Versions-ID an), schlägt die Anfrage fehl, wenn Sie kein gültiges MFA-Token bereitstellen. Wenn Sie die Multi-Object Delete-API für einen MFA-fähigen Bucket verwenden und einer der Löschanfrage versionsfähig ist (d. h. Sie geben einen Objektschlüssel und eine Versions-ID an), schlägt die gesamte Anfrage fehl, wenn Sie kein MFA-Token bereitstellen.

In den folgenden Fällen dagegen ist die Anfrage erfolgreich:

- Wenn Sie einen MFA-kompatiblen Bucket besitzen und eine nicht versionierte Löschanfrage ausführen (d. h. kein versioniertes Objekt löschen) und Sie kein MFA-Token angeben, wird die Löschanfrage erfolgreich ausgeführt.
- Wenn Sie eine Löschanfrage für mehrere Objekte ausführen und ausschließlich nicht versionierte Objekte zur Löschung aus einem MFA-kompatiblen Bucket angeben und Sie kein MFA-Token angeben, werden die Objekt erfolgreich gelöscht.

Weitere Informationen zum Löschen mit MFA finden Sie unter MFA Delete (p. 516).

Zugehörige Ressourcen

- Verwenden der AWS SDKs, CLI und Explorer (p. 801)

Löschen eines Objekts pro Anfrage

Themen

- Löschen eines Objekts mit dem AWS SDK for Java (p. 252)
- Löschen eines Objekts mit dem AWS SDK für .NET (p. 254)
- Löschen eines Objekts mit dem AWS SDK für PHP (p. 256)
- Löschen eines Objekts mit der REST API (p. 257)

- Löschen eines Objekts über die AWS Command Line Interface

Um ein einzelnes Objekt pro Anfrage zu löschen, verwenden Sie die `DELETE` API (siehe DELETE Object). Weitere Informationen zum Löschen von Objekten finden Sie unter Löschen von Objekten (p. 250).

Sie können entweder die REST API direkt verwenden oder die Wrapper-Bibliotheken der AWS SDKs, womit Sie Ihre Anwendungsentwicklung vereinfachen können.

Löschen eines Objekts mit dem AWS SDK for Java

Sie können ein Objekt aus einem Bucket löschen. Bei einem S3-Versioning-fähigen Bucket stehen Ihnen die folgenden Optionen zur Verfügung:

- Löschen einer spezifischen Objektversion durch Angabe einer Versions-ID.
- Löschen eines Objekts ohne Angabe einer Version-ID. In diesem Fall fügt S3 dem Objekt eine Löschmarkierung hinzu.

Mehr über S3-Versioning erfahren Sie unter Objekt-Versioning (p. 127).

Example Beispiel 1: Löschen eines Objekts (nicht versionierter Bucket)

Das folgende Beispiel löscht ein Objekt aus einem Bucket. Das Beispiel geht davon aus, dass der Bucket nicht versionsfähig ist und dass das Objekt über keine Versions-IDs verfügt. In der Löschanfrage geben Sie nur den Objektschlüssel und keine Versions-ID an. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;

import java.io.IOException;

public class DeleteObjectNonVersionedBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(bucketName, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
```

```
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Beispiel 2: Löschen eines Objekts (versionierter Bucket)

Das folgende Beispiel löscht ein Objekt aus einem versionierten Bucket. Das Beispiel löscht eine spezifische Objektversion durch Angabe des Namens des Objektschlüssels und der Versions-ID. Das Beispiel erledigt Folgendes:

1. Es fügt dem Bucket ein Beispielobjekt hinzu. Amazon S3 gibt die Versions-ID des neu hinzugefügten Objekts zurück. Das Beispiel verwendet diese Versions-ID in der Löschanfrage.
2. Löscht die Objektversion durch Angabe sowohl des Objektschlüsselnamens als auch einer Versions-ID. Wenn keine anderen Versionen dieses Objekts vorhanden sind, löscht Amazon S3 das gesamte Objekt. Andernfalls löscht Amazon S3 die angegebene Version.

   Note

   Sie können Version-IDs eines Objekts durch Senden einer `ListVersions`-Anfrage anfordern.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.DeleteVersionRequest;
import com.amazonaws.services.s3.model.PutObjectResult;

import java.io.IOException;

public class DeleteObjectVersionEnabledBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Check to ensure that the bucket is versioning-enabled.
            String bucketVersionStatus =
 s3Client.getBucketVersioningConfiguration(bucketName).getStatus();
            if (!bucketVersionStatus.equals(BucketVersioningConfiguration.ENABLED)) {
                System.out.printf("Bucket %s is not versioning-enabled.", bucketName);
            } else {
                // Add an object.
                PutObjectResult putResult = s3Client.putObject(bucketName, keyName, "Sample
 content for deletion example.");
```

```
                System.out.printf("Object %s added to bucket %s\n", keyName, bucketName);

                // Delete the version of the object that we just created.
                System.out.println("Deleting versioned object " + keyName);
                s3Client.deleteVersion(new DeleteVersionRequest(bucketName, keyName,
 putResult.getVersionId()));
                System.out.printf("Object %s, version %s deleted\n", keyName,
 putResult.getVersionId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Löschen eines Objekts mit dem AWS SDK für .NET

Wenn Sie ein Objekt aus einem nicht versionierten Bucket löschen, wird das Objekt entfernt. Bei einem S3-Versioning-fähigen Bucket stehen Ihnen die folgenden Optionen zur Verfügung:

- Löschen einer bestimmten Version eines Objekts durch Angabe einer Versions-ID.
- Löschen eines Objekts ohne Angabe einer Versions-ID. Amazon S3 fügt eine Löschmarkierung hinzu. Weitere Informationen zu Löschmarkierungen finden Sie unter Objekt-Versioning (p. 127).

Das folgende Beispiel veranschaulicht, wie Sie ein Objekt sowohl aus versionierten als auch aus nicht-versionierten Buckets löschen. Mehr über S3-Versioning erfahren Sie unter Objekt-Versioning (p. 127).

Example Löschen eines Objekts aus einem nicht-versionierten Bucket

Das folgende C#-Beispiel löscht ein Objekt aus einem nicht-versionierten Bucket. Das Beispiel geht davon aus, dass die Objekte über keine Versions-IDs verfügen. Sie geben daher keine Versions-IDs an. Sie geben nur den Objektschlüssel ein. Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteObjectNonVersionedBucketTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            DeleteObjectNonVersionedBucketAsync().Wait();
        }
```

```
        private static async Task DeleteObjectNonVersionedBucketAsync()
        {
            try
            {
                var deleteObjectRequest = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };

                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(deleteObjectRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when deleting
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 deleting an object", e.Message);
            }
        }
    }
}
```

Example Löschen eines Objekts aus einem versionierten Bucket

Das folgende C#-Beispiel löscht ein Objekt aus einem versionierten Bucket. Es löscht eine bestimmte Objektversion durch Angabe des Namens des Objektschlüssels und der Versions-ID.

Der Code führt die folgenden Aufgaben durch:

1. Aktiviert das S3-Versioning für einen Bucket, den Sie angeben (wenn Versioning bereits aktiviert ist, hat dies keine Auswirkungen).
2. Fügt dem Bucket ein Beispielobjekt hinzu. In der Antwort gibt Amazon S3 die Versions-ID des neu hinzugefügten Objekts zurück. Das Beispiel verwendet diese Versions-ID in der Löschanfrage.
3. Löscht das Beispielobjekt durch Angabe sowohl des Objektschlüsselnamens als auch einer Versions-ID.

   Note

   Sie können die Version-IDs eines Objekts durch Senden einer `ListVersions`-Anfrage anfordern:

   ```
   var listResponse = client.ListVersions(new ListVersionsRequest { BucketName =
    bucketName, Prefix = keyName });
   ```

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteObjectVersion
```

```
    {
        private const string bucketName = "*** versioning-enabled bucket name ***";
        private const string keyName = "*** Object Key Name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            CreateAndDeleteObjectVersionAsync().Wait();
        }

        private static async Task CreateAndDeleteObjectVersionAsync()
        {
            try
            {
                // Add a sample object.
                string versionID = await PutAnObject(keyName);

                // Delete the object by specifying an object key and a version ID.
                DeleteObjectRequest request = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    VersionId = versionID
                };
                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when deleting
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 deleting an object", e.Message);
            }
        }

        static async Task<string> PutAnObject(string objectKey)
        {
            PutObjectRequest request = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = objectKey,
                ContentBody = "This is the content body!"
            };
            PutObjectResponse response = await client.PutObjectAsync(request);
            return response.VersionId;
        }
    }
}
```

Löschen eines Objekts mit dem AWS SDK für PHP

Dieses Thema führt durch die Verwendung von AWS SDK für PHP-Klassen aus Version 3 zum Löschen eines Objekts aus einem nicht versionierten Bucket. Weitere Informationen über das Löschen eines Objekts aus einem versionsfähigen Bucket finden Sie unter Löschen eines Objekts mit der REST API (p. 257)

Dieser Abschnitt setzt voraus, dass Sie den Anweisungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und AWS SDK für PHP ordnungsgemäß installiert ist.

Das folgende PHP-Beispiel löscht ein Objekt aus einem Bucket. Da dieses Beispiel zeigt, wie Objekte aus nicht versionsgesteuerten Buckets gelöscht werden, gibt es in der Löschanfrage nur den Bucket-Namen und Objektschlüssel (keine Versions-ID) an. Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
<?php

require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Delete an object from the bucket.
$s3->deleteObject([
    'Bucket' => $bucket,
    'Key'    => $keyname
]);
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Löschen eines Objekts mit der REST API

Sie können ein Objekt mit den AWS SDKs löschen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Weitere Informationen finden Sie unter DELETE Object in Amazon Simple Storage Service API Reference.

Löschen mehrerer Objekte pro Anforderung

Themen

Amazon S3 stellt die Multi-Object Delete API bereit (siehe Löschen – Löschen mehrerer Objekte), mit der Sie mehrere Objekte in einer einzelnen Anfrage löschen können. Die API unterstützt zwei Modi für die Antwort: „Verbose“ und „Quiet“. Standardmäßig verwendet die Operation den Modus „Verbose“. Im Modus „Verbose“ enthält die Antwort das Ergebnis der Löschung der einzelnen Schlüssel, die in Ihrer Anfrage angegeben werden. Im Modus „Quiet“ enthält die Antwort nur die Schlüssel, für die es während der Löschoperation einen Fehler gegeben hat. Wenn bei Verwendung des Modus „Quiet“ alle Schlüssel erfolgreich gelöscht wurden, gibt Amazon S3 eine leere Antwort zurück.

Weitere Informationen zum Löschen von Objekten finden Sie unter Löschen von Objekten (p. 250).

Sie können direkt die REST API oder die AWS SDKs verwenden.

Löschen mehrerer Objekte mit dem AWS SDK for Java

AWS SDK for Java stellt zum Löschen mehrerer Objekte die Methode `AmazonS3Client.deleteObjects()` bereit. Sie geben den Schlüsselnamen eines jeden Objekts an, das Sie löschen möchten. Bei einem versionsfähigen Bucket stehen Ihnen die folgenden Optionen zur Verfügung:

- Sie geben ausschließlich den Namen des Objektschlüssels an. Amazon S3 fügt dem Objekt eine Löschmarkierung hinzu.
- Sie geben sowohl den Namen des Schlüssels als auch die Versions-ID des Objekts an, das gelöscht werden soll. Amazon S3 löscht die angegebene Version des Objekts.

Example

Das folgende Beispiel verwendet die Multi-Object Delete API, um Objekte aus einem nicht versionsfähigen Bucket zu löschen. Das Beispiel lädt Beispielobjekte in den Bucket hoch und löscht diese Objekte dann in einer einzigen Anfrage mit der Methode `AmazonS3Client.deleteObjects()`. In der `DeleteObjectsRequest` gibt das Beispiel nur die Objektschlüsselnamen an, da die Objekte keine Versions-IDs besitzen.

Weitere Informationen zum Löschen von Objekten finden Sie unter Löschen von Objekten (p. 250). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;

import java.io.IOException;

public class DeleteObjectNonVersionedBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(bucketName, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example

Das folgende Beispiel verwendet die Multi-Object Delete API, um Objekte aus einem versionsfähigen Bucket zu löschen. Es führt die folgenden Aktionen aus:

1. Erstellt Beispielobjekte und löscht sie dann, indem der Schlüsselname und die Versions-ID eines jeden zu löschenden Objekts angegeben werden. Die Operation löscht nur die angegebenen Objektversionen.
2. Erstellt Beispielobjekte und löscht sie dann, indem nur die Schlüsselnamen angegeben werden. Da das Beispiel keine Versions-IDs angibt, fügt die Operation zu jedem Objekt eine Löschmarkierung hinzu, ohne spezifische Objektversionen zu löschen. Nachdem die Löschmarkierungen hinzugefügt wurden, erscheinen diese Objekte nicht mehr in der AWS Management Console.
3. Entfernen Sie die Löschmarkierungen, indem Sie die Objektschlüssel und Versions-IDs der Löschmarkierungen angeben. Die Operation löscht die Löschmarkierungen. Dies hat zur Folge, dass die Objekte wieder in der AWS Management Console erscheinen.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.DeleteObjectsRequest;
import com.amazonaws.services.s3.model.DeleteObjectsRequest.KeyVersion;
import com.amazonaws.services.s3.model.DeleteObjectsResult;
import com.amazonaws.services.s3.model.DeleteObjectsResult.DeletedObject;
import com.amazonaws.services.s3.model.PutObjectResult;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class DeleteMultipleObjectsVersionEnabledBucket {
    private static AmazonS3 S3_CLIENT;
    private static String VERSIONED_BUCKET_NAME;

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        VERSIONED_BUCKET_NAME = "*** Bucket name ***";

        try {
            S3_CLIENT = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Check to make sure that the bucket is versioning-enabled.
            String bucketVersionStatus =
 S3_CLIENT.getBucketVersioningConfiguration(VERSIONED_BUCKET_NAME).getStatus();
            if (!bucketVersionStatus.equals(BucketVersioningConfiguration.ENABLED)) {
                System.out.printf("Bucket %s is not versioning-enabled.",
 VERSIONED_BUCKET_NAME);
            } else {
                // Upload and delete sample objects, using specific object versions.
                uploadAndDeleteObjectsWithVersions();

                // Upload and delete sample objects without specifying version IDs.
                // Amazon S3 creates a delete marker for each object rather than deleting
                // specific versions.
                DeleteObjectsResult unversionedDeleteResult =
 uploadAndDeleteObjectsWithoutVersions();
```

```
                // Remove the delete markers placed on objects in the non-versioned create/
delete method.
                multiObjectVersionedDeleteRemoveDeleteMarkers(unversionedDeleteResult);
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void uploadAndDeleteObjectsWithVersions() {
        System.out.println("Uploading and deleting objects with versions specified.");

        // Upload three sample objects.
        ArrayList<KeyVersion> keys = new ArrayList<KeyVersion>();
        for (int i = 0; i < 3; i++) {
            String keyName = "delete object without version ID example " + i;
            PutObjectResult putResult = S3_CLIENT.putObject(VERSIONED_BUCKET_NAME, keyName,
                    "Object number " + i + " to be deleted.");
            // Gather the new object keys with version IDs.
            keys.add(new KeyVersion(keyName, putResult.getVersionId()));
        }

        // Delete the specified versions of the sample objects.
        DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest(VERSIONED_BUCKET_NAME)
                .withKeys(keys)
                .withQuiet(false);

        // Verify that the object versions were successfully deleted.
        DeleteObjectsResult delObjRes = S3_CLIENT.deleteObjects(multiObjectDeleteRequest);
        int successfulDeletes = delObjRes.getDeletedObjects().size();
        System.out.println(successfulDeletes + " objects successfully deleted");
    }

    private static DeleteObjectsResult uploadAndDeleteObjectsWithoutVersions() {
        System.out.println("Uploading and deleting objects with no versions specified.");

        // Upload three sample objects.
        ArrayList<KeyVersion> keys = new ArrayList<KeyVersion>();
        for (int i = 0; i < 3; i++) {
            String keyName = "delete object with version ID example " + i;
            S3_CLIENT.putObject(VERSIONED_BUCKET_NAME, keyName, "Object number " + i + " to
 be deleted.");
            // Gather the new object keys without version IDs.
            keys.add(new KeyVersion(keyName));
        }

        // Delete the sample objects without specifying versions.
        DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest(VERSIONED_BUCKET_NAME).withKeys(keys)
                .withQuiet(false);

        // Verify that delete markers were successfully added to the objects.
        DeleteObjectsResult delObjRes = S3_CLIENT.deleteObjects(multiObjectDeleteRequest);
        int successfulDeletes = delObjRes.getDeletedObjects().size();
        System.out.println(successfulDeletes + " objects successfully marked for deletion
 without versions.");
        return delObjRes;
    }
```

```
    private static void multiObjectVersionedDeleteRemoveDeleteMarkers(DeleteObjectsResult
 response) {
        List<KeyVersion> keyList = new ArrayList<KeyVersion>();
        for (DeletedObject deletedObject : response.getDeletedObjects()) {
            // Note that the specified version ID is the version ID for the delete marker.
            keyList.add(new KeyVersion(deletedObject.getKey(),
 deletedObject.getDeleteMarkerVersionId()));
        }
        // Create a request to delete the delete markers.
        DeleteObjectsRequest deleteRequest = new
 DeleteObjectsRequest(VERSIONED_BUCKET_NAME).withKeys(keyList);

        // Delete the delete markers, leaving the objects intact in the bucket.
        DeleteObjectsResult delObjRes = S3_CLIENT.deleteObjects(deleteRequest);
        int successfulDeletes = delObjRes.getDeletedObjects().size();
        System.out.println(successfulDeletes + " delete markers successfully deleted");
    }
}
```

Löschen mehrerer Objekte mit dem AWS SDK für .NET

AWS SDK für .NET stellt eine komfortable Methode für das Löschen mehrerer Objekte bereit: `DeleteObjects`. Sie geben für jedes Objekt, das Sie löschen möchten, den Schlüsselnamen und die Version des Objekts ein. Wenn der Bucket nicht versionsfähig ist, geben Sie für die Version-ID `null` ein. Wenn eine Ausnahme auftritt, stellen Sie anhand der `DeleteObjectsException`-Antwort fest, welche Objekte aus welchem Grund nicht gelöscht wurden.

Example Löschen mehrerer Objekte aus einem nicht-versionsfähigen Bucket

Das folgende C#-Beispiel verwendet die Multi-Object Delete API, um Objekte aus einem nicht versionsfähigen Bucket zu löschen. Das Beispiel lädt die Beispielobjekte in den Bucket hoch und löscht diese Objekte dann in einer einzigen Anfrage mit der Methode `DeleteObjects`. In der `DeleteObjectsRequest` gibt das Beispiel nur die Objektschlüsselnamen an, weil die Versions-IDs null sind.

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteMultipleObjectsNonVersionedBucketTest
    {
        private const string bucketName = "*** versioning-enabled bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            MultiObjectDeleteAsync().Wait();
        }

        static async Task MultiObjectDeleteAsync()
```

```
        {
            // Create sample objects (for subsequent deletion).
            var keysAndVersions = await PutObjectsAsync(3);

            // a. multi-object delete by specifying the key names and version IDs.
            DeleteObjectsRequest multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keysAndVersions // This includes the object keys and null version
IDs.
            };
            // You can add specific object key to the delete request using the .AddKey.
            // multiObjectDeleteRequest.AddKey("TickerReference.csv", null);
            try
            {
                DeleteObjectsResponse response = await
s3Client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException e)
            {
                PrintDeletionErrorStatus(e);
            }
        }

        private static void PrintDeletionErrorStatus(DeleteObjectsException e)
        {
            // var errorResponse = e.ErrorResponse;
            DeleteObjectsResponse errorResponse = e.Response;
            Console.WriteLine("x {0}", errorResponse.DeletedObjects.Count);

            Console.WriteLine("No. of objects successfully deleted = {0}",
errorResponse.DeletedObjects.Count);
            Console.WriteLine("No. of objects failed to delete = {0}",
errorResponse.DeleteErrors.Count);

            Console.WriteLine("Printing error data...");
            foreach (DeleteError deleteError in errorResponse.DeleteErrors)
            {
                Console.WriteLine("Object Key: {0}\t{1}\t{2}", deleteError.Key,
deleteError.Code, deleteError.Message);
            }
        }

        static async Task<List<KeyVersion>> PutObjectsAsync(int number)
        {
            List<KeyVersion> keys = new List<KeyVersion>();
            for (int i = 0; i < number; i++)
            {
                string key = "ExampleObject-" + new System.Random().Next();
                PutObjectRequest request = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = key,
                    ContentBody = "This is the content body!",
                };

                PutObjectResponse response = await s3Client.PutObjectAsync(request);
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = key,
                    // For non-versioned bucket operations, we only need object key.
                    // VersionId = response.VersionId
                };
                keys.Add(keyVersion);
```

```
            }
            return keys;
        }
    }
}
```

Example Löschen mehrerer Objekte für einen versionsfähigen Bucket

Das folgende C#-Beispiel verwendet die Multi-Objekt Delete API aus einem versionsfähigen Bucket. Das Beispiel führt die folgenden Aktionen aus:

1. Erstellt Beispielobjekte und löscht sie, indem der Schlüsselname und die Versions-ID für jedes Objekt angegeben werden. Die Operation löscht spezifische Versionen der Objekte.
2. Erstellt Beispielobjekte und löscht sie, indem nur die Schlüsselnamen angegeben werden. Da das Beispiel keine Versions-IDs angibt, fügt die Operation nur Löschmarkierungen hinzu. Sie löscht keine spezifischen Versionen der Objekte. Nach dem Löschvorgang erscheinen diese Objekte nicht mehr in der Amazon S3-Konsole.
3. Löscht die Löschmarkierungen, indem die Objektschlüssel und Versions-IDs der Löschmarkierungen angegeben werden. Wenn die Operation die Löschmarkierungen löscht, erscheinen die Objekte wieder in der Konsole.

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteMultipleObjVersionedBucketTest
    {
        private const string bucketName = "*** versioning-enabled bucket name ***";
       // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            DeleteMultipleObjectsFromVersionedBucketAsync().Wait();
        }

        private static async Task DeleteMultipleObjectsFromVersionedBucketAsync()
        {

            // Delete objects (specifying object version in the request).
            await DeleteObjectVersionsAsync();

            // Delete objects (without specifying object version in the request).
            var deletedObjects = await DeleteObjectsAsync();

            // Additional exercise - remove the delete markers S3 returned in the preceding
 response.
            // This results in the objects reappearing in the bucket (you can
            // verify the appearance/disappearance of objects in the console).
            await RemoveDeleteMarkersAsync(deletedObjects);
        }
```

```
        private static async Task<List<DeletedObject>> DeleteObjectsAsync()
        {
            // Upload the sample objects.
            var keysAndVersions2 = await PutObjectsAsync(3);

            // Delete objects using only keys. Amazon S3 creates a delete marker and
            // returns its version ID in the response.
            List<DeletedObject> deletedObjects = await
NonVersionedDeleteAsync(keysAndVersions2);
            return deletedObjects;
        }

        private static async Task DeleteObjectVersionsAsync()
        {
            // Upload the sample objects.
            var keysAndVersions1 = await PutObjectsAsync(3);

            // Delete the specific object versions.
            await VersionedDeleteAsync(keysAndVersions1);
        }

        private static void PrintDeletionReport(DeleteObjectsException e)
        {
            var errorResponse = e.Response;
            Console.WriteLine("No. of objects successfully deleted = {0}",
errorResponse.DeletedObjects.Count);
            Console.WriteLine("No. of objects failed to delete = {0}",
errorResponse.DeleteErrors.Count);
            Console.WriteLine("Printing error data...");
            foreach (var deleteError in errorResponse.DeleteErrors)
            {
                Console.WriteLine("Object Key: {0}\t{1}\t{2}", deleteError.Key,
deleteError.Code, deleteError.Message);
            }
        }

        static async Task VersionedDeleteAsync(List<KeyVersion> keys)
        {
            // a. Perform a multi-object delete by specifying the key names and version
IDs.
            var multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keys // This includes the object keys and specific version IDs.
            };
            try
            {
                Console.WriteLine("Executing VersionedDelete...");
                DeleteObjectsResponse response = await
s3Client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException e)
            {
                PrintDeletionReport(e);
            }
        }

        static async Task<List<DeletedObject>> NonVersionedDeleteAsync(List<KeyVersion>
keys)
        {
            // Create a request that includes only the object key names.
            DeleteObjectsRequest multiObjectDeleteRequest = new DeleteObjectsRequest();
            multiObjectDeleteRequest.BucketName = bucketName;
```

```
            foreach (var key in keys)
            {
                multiObjectDeleteRequest.AddKey(key.Key);
            }
            // Execute DeleteObjects - Amazon S3 add delete marker for each object
            // deletion. The objects disappear from your bucket.
            // You can verify that using the Amazon S3 console.
            DeleteObjectsResponse response;
            try
            {
                Console.WriteLine("Executing NonVersionedDelete...");
                response = await s3Client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException e)
            {
                PrintDeletionReport(e);
                throw; // Some deletes failed. Investigate before continuing.
            }
            // This response contains the DeletedObjects list which we use to delete the
delete markers.
            return response.DeletedObjects;
        }

        private static async Task RemoveDeleteMarkersAsync(List<DeletedObject>
deletedObjects)
        {
            var keyVersionList = new List<KeyVersion>();

            foreach (var deletedObject in deletedObjects)
            {
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = deletedObject.Key,
                    VersionId = deletedObject.DeleteMarkerVersionId
                };
                keyVersionList.Add(keyVersion);
            }
            // Create another request to delete the delete markers.
            var multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keyVersionList
            };

            // Now, delete the delete marker to bring your objects back to the bucket.
            try
            {
                Console.WriteLine("Removing the delete markers .....");
                var deleteObjectResponse = await
s3Client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} delete markers",
                                            deleteObjectResponse.DeletedObjects.Count);
            }
            catch (DeleteObjectsException e)
            {
                PrintDeletionReport(e);
            }
        }

        static async Task<List<KeyVersion>> PutObjectsAsync(int number)
        {
            var keys = new List<KeyVersion>();
```

```
            for (var i = 0; i < number; i++)
            {
                string key = "ObjectToDelete-" + new System.Random().Next();
                PutObjectRequest request = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = key,
                    ContentBody = "This is the content body!",

                };

                var response = await s3Client.PutObjectAsync(request);
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = key,
                    VersionId = response.VersionId
                };

                keys.Add(keyVersion);
            }
            return keys;
        }
    }
}
```

Löschen mehrerer Objekte mit dem AWS SDK für PHP

In diesem Abschnitt wird gezeigt, wie Klassen aus Version 3 von AWS SDK für PHP verwendet werden, um mehrere Objekte aus versionierten und nicht versionierten Amazon S3-Buckets zu löschen. Weitere Informationen über Versioning finden Sie unter Verwenden von Versioning (p. 514).

Dieser Abschnitt setzt voraus, dass Sie den Anweisungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und AWS SDK für PHP ordnungsgemäß installiert ist.

Example Löschen mehrerer Objekte aus einem nicht-versionierten Bucket

Das folgende PHP-Beispiel verwendet die Methode `deleteObjects()`, um mehrere Objekte aus einem nicht versionsfähigen Bucket zu löschen.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
<?php

require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// 1. Create a few objects.
for ($i = 1; $i <= 3; $i++) {
    $s3->putObject([
        'Bucket' => $bucket,
        'Key'    => "key{$i}",
        'Body'   => "content {$i}",
    ]);
}
```

```
// 2. List the objects and get the keys.
$keys = $s3->listObjects([
    'Bucket' => $bucket
]) ->getPath('Contents/*/Key');

// 3. Delete the objects.
$s3->deleteObjects([
    'Bucket'  => $bucket,
    'Delete' => [
        'Objects' => array_map(function ($key) {
            return ['Key' => $key];
        }, $keys)
    ],
]);
```

Example Löschen mehrerer Objekte aus einem versionsfähigen Bucket

Das folgende PHP-Beispiel verwendet die Methode `deleteObjects()`, um mehrere Objekte aus einem versionsfähigen Bucket zu löschen.

Weitere Informationen zur Ausführung der PHP-Beispiele in dieser Anleitung finden Sie unter PHP-Beispiele ausführen (p. 812).

```
<?php

require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// 1. Enable object versioning for the bucket.
$s3->putBucketVersioning([
    'Bucket' => $bucket,
    'Status' => 'Enabled',
]);

// 2. Create a few versions of an object.
for ($i = 1; $i <= 3; $i++) {
    $s3->putObject([
        'Bucket' => $bucket,
        'Key'    => $keyname,
        'Body'   => "content {$i}",
    ]);
}

// 3. List the objects versions and get the keys and version IDs.
$versions = $s3->listObjectVersions(['Bucket' => $bucket])
    ->getPath('Versions');

// 4. Delete the object versions.
$s3->deleteObjects([
    'Bucket'  => $bucket,
    'Delete' => [
        'Objects' => array_map(function ($version) {
          return [
              'Key'       => $version['Key'],
```

```
                'VersionId' => $version['VersionId']
            }, $versions),
        ],
]);

echo "The following objects were deleted successfully:". PHP_EOL;
foreach ($result['Deleted'] as $object) {
    echo "Key: {$object['Key']}, VersionId: {$object['VersionId']}" . PHP_EOL;
}

echo PHP_EOL . "The following objects could not be deleted:" . PHP_EOL;
foreach ($result['Errors'] as $object) {
    echo "Key: {$object['Key']}, VersionId: {$object['VersionId']}" . PHP_EOL;
}

// 5. Suspend object versioning for the bucket.
$s3->putBucketVersioning([
    'Bucket' => $bucket,
    'Status' => 'Suspended',
]);
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Löschen mehrerer Objekte mit der REST-API

Sie können die AWS SDKs verwenden, um mehrere Objekte mit der Multi-Object Delete API zu löschen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Weitere Informationen finden Sie unter Löschen mehrerer Objekte in Amazon Simple Storage Service API Reference.

Auswählen von Inhalten aus Objekten

Mit Amazon S3 Select können Sie mit einfachen Structured Query Language (SQL)-Anweisungen die Inhalte von Amazon S3-Objekten filtern, sodass nur die tatsächlich von Ihnen benötigte Teilmenge an Daten abgerufen wird. Wenn Sie die Daten mit Amazon S3 Select filtern, wird die von Amazon S3 übertragene Datenmenge reduziert. Dies reduziert Kosten und Latenz im Zusammenhang mit dem Abruf dieser Daten.

Amazon S3 Select funktioniert für Objekte, die in den Formaten CSV, JSON oder Apache Parquet gespeichert sind. Es funktioniert auch mit Objekten, die mit GZIP oder BZIP2 (nur für CSV- und JSON-Objekte) komprimiert wurden, sowie für serverseitig verschlüsselte Objekte. Als Format für die Ergebnisse geben Sie entweder CSV oder JSON an. Sie können ebenfalls festlegen, wie die Datensätze im Ergebnis getrennt werden.

In der Anforderung werden SQL-Ausdrücke an Amazon S3 übergeben. Amazon S3 Select unterstützt eine Teilmenge von SQL. Weitere Informationen zu den von Amazon S3 Select unterstützten SQL-Elementen finden Sie unter SQL-Referenz für Amazon S3 Select und S3 Glacier Select. (p. 849).

Sie können SQL-Abfragen über AWS SDKs, die SELECT Object Content REST-API, die AWS Command Line Interface (AWS CLI) oder die Amazon S3-Konsole ausführen. Bei der Amazon S3-Konsole ist die zurückgegebene Datenmenge auf 40 MB begrenzt. Sollen mehr Daten abgerufen werden, verwenden Sie die AWS CLI oder die API.

Voraussetzungen und Einschränkungen

Für die Verwendung von Amazon S3 Select müssen folgende Voraussetzungen erfüllt sein:

- Sie haben die Berechtigung `s3:GetObject` für das abzufragende Objekt.
- Falls das abzufragende Objekt mit einem vom Kunden bereitgestellten Verschlüsselungsschlüssel (SSE-C) verschlüsselt ist, müssen Sie `https` verwenden und den Verschlüsselungsschlüssel in der Anforderung angeben.

Für die Verwendung von Amazon S3 Select gelten die folgenden Einschränkungen:

- Die maximale Länge des SQL-Ausdrucks ist 256 KB.
- Die Maximallänge eines Datensatzes in der Eingabe oder im Ergebnis liegt bei 1 MB.
- Amazon S3 Select kann nur mithilfe des JSON-Ausgabeformats verschachtelte Daten übermitteln.

Bei Verwendung von Amazon S3 Select mit Parquet-Objekten gelten zusätzliche Einschränkungen.

- Amazon S3 Select unterstützt ausschließlich die Spaltenkompression mittels GZIP oder Snappy. Amazon S3 Select unterstützt für Parquet-Objekte keine Kompression ganzer Objekte.
- Amazon S3 Select unterstützt keine Parquet-Ausgabe. Sie müssen das Ausgabeformat als CSV oder JSON angeben.
- Die Maximalgröße für unkomprimierte Zeilengruppen beträgt 256 MB.
- Sie müssen die im Schema des Objekts angegebenen Datentypen verwenden.
- Select-Anfragen für ein wiederholtes Feld geben ausschließlich den letzten Wert zurück.

Erstellen einer Anforderung

Beim Erstellen einer Anforderung geben Sie die Details des abzufragenden Objekts mithilfe eines `InputSerialization`-Objekts an. Die Details zur Rückgabe der Ergebnisse geben Sie mit einem `OutputSerialization`-Objekt an. Außerdem binden Sie den SQL-Ausdruck ein, der von Amazon S3 zum Filtern der Anforderung verwendet wird.

Weitere Informationen zum Konstruieren einer Amazon S3 Select-Anforderung Sie unter SELECTObjectContent im Amazon Simple Storage Service API Reference. In den folgenden Abschnitten finden Sie eines der SDK-Codebeispiele.

Anforderungen mittels Scanbereichen

Mit Amazon S3 Select können Sie eine Teilmenge eines Objekts scannen, indem Sie einen abzufragenden Byte-Bereich angeben. Mit dieser Fähigkeit können Sie das gesamte Objekt parallel scannen, indem Sie die Arbeit in separate Amazon S3 Select-Anforderungen für eine Reihe von sich nicht überlappenden Scanbereichen aufteilen. Scanbereiche müssen nicht an Datensatzgrenzen ausgerichtet werden. Eine Scanbereichsanforderung von Amazon S3 Select wird über den angegebenen Byte-Bereich ausgeführt. Ein Datensatz, der im angegebenen Scanbereich beginnt, aber darüber hinaus reicht, wird von der Abfrage bearbeitet. Beispiel: Im Folgenden wird ein Amazon S3-Objekt gezeigt, das eine Reihe von Datensätzen in einem zeilengetrennten CSV-Format enthält:

```
A,B
C,D
D,E
E,F
G,H
I,J
```

Verwenden Sie den Parameter `ScanRange` von Amazon S3 Select und beginnen Sie bei (Byte) 1 und enden Sie bei (Byte) 4. Damit beginnt der Scanbereich bei "," und scannt bis zum Ende des Datensatzes, der bei "C" beginnt. Damit wird das Ergebnis C, D zurückgegeben, da dies das Ende des Datensatzes ist.

Scanbereichsanfragen von Amazon S3 Select unterstützen Parquet-, CSV- (ohne Trennung mit Anführungszeichen) und JSON-Objekte (nur im LINES-Modus) CSV- und JSON-Objekte dürfen nicht komprimiert sein. Wenn bei zeilenbasierten CSV- und JSON-Objekten ein Scanbereich als Teil der Amazon S3 Select-Anforderung angegeben wird, werden alle Datensätze, die innerhalb des Scanbereichs beginnen, verarbeitet. Bei Parquet-Objekten werden alle Zeilengruppen, die innerhalb des angeforderten Scanbereich beginnen, verarbeitet.

Scanbereichsanforderungen von Amazon S3 Select sind zur Verwendung in der CLI, der API oder dem SDK von Amazon S3 verfügbar. Sie können den `ScanRange`-Parameter in der Amazon S3 Select-Anfrage als diese Funktion verwenden. Weitere Informationen finden Sie im Amazon S3 SELECT-Objektinhalt im Amazon Simple Storage Service API Reference.

Fehler

Amazon S3 Select gibt einen Fehlercode und eine zugehörige Fehlermeldung zurück, wenn beim Versuch, eine Abfrage auszuführen, ein Problem auftritt. Eine Liste der Fehlercodes und Beschreibungen finden Sie im Abschnitt List of SELECT Object Content Error Codes auf der Seite Error Responses im Amazon Simple Storage Service API Reference.

Themen

Zugehörige Ressourcen

Auswählen von Inhalten aus Objekten mit dem SDK für Java

Mit Amazon S3 Select können Sie Inhalte eines Objekts mit Java unter Verwendung der Methode `selectObjectContent` auswählen, die bei erfolgreicher Ausführung die Ergebnisse des SQL-Ausdrucks zurückgibt. Der angegebene Bucket und der Objektschlüssel müssen vorhanden sein, andernfalls tritt ein Fehler auf.

Example Beispiel

Der folgende Java-Code gibt den Wert der ersten Spalte für jeden Datensatz zurück, der in einem Objekt gespeichert ist, das im CSV-Format gespeicherte Daten enthält. Im Beispiel müssen auch `Progress`- und `Stats`-Meldungen zurückgegeben werden. Sie müssen einen gültigen Bucket-Namen sowie ein Objekt angeben, das Daten im CSV-Format enthält.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
package com.amazonaws;

import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CSVInput;
import com.amazonaws.services.s3.model.CSVOutput;
import com.amazonaws.services.s3.model.CompressionType;
import com.amazonaws.services.s3.model.ExpressionType;
import com.amazonaws.services.s3.model.InputSerialization;
import com.amazonaws.services.s3.model.OutputSerialization;
```

```
import com.amazonaws.services.s3.model.SelectObjectContentEvent;
import com.amazonaws.services.s3.model.SelectObjectContentEventVisitor;
import com.amazonaws.services.s3.model.SelectObjectContentRequest;
import com.amazonaws.services.s3.model.SelectObjectContentResult;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.concurrent.atomic.AtomicBoolean;

import static com.amazonaws.util.IOUtils.copy;

/**
 * This example shows how to query data from S3Select and consume the response in the form
 of an
 * InputStream of records and write it to a file.
 */

public class RecordInputStreamExample {

    private static final String BUCKET_NAME = "${my-s3-bucket}";
    private static final String CSV_OBJECT_KEY = "${my-csv-object-key}";
    private static final String S3_SELECT_RESULTS_PATH = "${my-s3-select-results-path}";
    private static final String QUERY = "select s._1 from S3Object s";

    public static void main(String[] args) throws Exception {
        final AmazonS3 s3Client = AmazonS3ClientBuilder.defaultClient();

        SelectObjectContentRequest request = generateBaseCSVRequest(BUCKET_NAME,
 CSV_OBJECT_KEY, QUERY);
        final AtomicBoolean isResultComplete = new AtomicBoolean(false);

        try (OutputStream fileOutputStream = new FileOutputStream(new File
 (S3_SELECT_RESULTS_PATH));
             SelectObjectContentResult result = s3Client.selectObjectContent(request)) {
            InputStream resultInputStream = result.getPayload().getRecordsInputStream(
                    new SelectObjectContentEventVisitor() {
                        @Override
                        public void visit(SelectObjectContentEvent.StatsEvent event)
                        {
                            System.out.println(
                                    "Received Stats, Bytes Scanned: " +
 event.getDetails().getBytesScanned()
                                            +  " Bytes Processed: " +
 event.getDetails().getBytesProcessed());
                        }

                        /*
                         * An End Event informs that the request has finished successfully.
                         */
                        @Override
                        public void visit(SelectObjectContentEvent.EndEvent event)
                        {
                            isResultComplete.set(true);
                            System.out.println("Received End Event. Result is complete.");
                        }
                    }
            );

            copy(resultInputStream, fileOutputStream);
        }

        /*
         * The End Event indicates all matching records have been transmitted.
         * If the End Event is not received, the results may be incomplete.
```

```
         */
        if (!isResultComplete.get()) {
            throw new Exception("S3 Select request was incomplete as End Event was not
 received.");
        }
    }

    private static SelectObjectContentRequest generateBaseCSVRequest(String bucket, String
 key, String query) {
        SelectObjectContentRequest request = new SelectObjectContentRequest();
        request.setBucketName(bucket);
        request.setKey(key);
        request.setExpression(query);
        request.setExpressionType(ExpressionType.SQL);

        InputSerialization inputSerialization = new InputSerialization();
        inputSerialization.setCsv(new CSVInput());
        inputSerialization.setCompressionType(CompressionType.NONE);
        request.setInputSerialization(inputSerialization);

        OutputSerialization outputSerialization = new OutputSerialization();
        outputSerialization.setCsv(new CSVOutput());
        request.setOutputSerialization(outputSerialization);

        return request;
    }
}
```

Auswählen von Inhalten aus Objekten mit der REST-API

Mit dem AWS SDK können Sie Inhalte von Objekten auswählen. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Weitere Informationen zum Format von Anfragen und Antworten finden Sie unter SELECT Object Content.

Auswählen von Inhalten aus Objekten mit anderen SDKs

Sie können Inhalte eines Objekts mit Amazon S3 Select unter Verwendung anderer SDKs auswählen. Weitere Informationen finden Sie unter:

- Python: Verwenden von AWS SDK for Python (Boto) (p. 814).

Wiederherstellen archivierter Objekte

Auf Objekte, die Sie in den Speicherklassen S3 Glacier oder S3 Glacier Deep Archive archivieren, kann nicht in Echtzeit zugegriffen werden. Sie müssen zuerst eine Anfrage zur Wiederherstellung initiieren und dann warten, bis eine temporäre Kopie des Objekts für die Dauer (Anzahl an Tagen) bereitgestellt wird, die Sie in der Anfrage angegeben haben. Weitere Informationen zum Vergleich der Speicherklassen S3 Glacier, S3 Glacier Deep Archive, und anderer Amazon S3-Speicherklassen finden Sie unter Amazon S3-Speicherklassen (p. 122).

Amazon S3 stellt eine temporäre Kopie des Objekts nur für die angegebene Dauer wieder her. Anschließend wird die Kopie des wiederhergestellten Objekts gelöscht. Sie können den Ablaufzeitraum einer wiederhergestellte Kopie durch die erneute Ausgabe einer Wiederherstellung ändern. In diesem Fall aktualisiert Amazon S3 den Ablaufzeitraum relativ zum aktuellen Zeitraum.

Amazon S3 berechnet die Ablaufzeit der wiederhergestellten Objektkopie durch Addition der Anzahl der in der Wiederherstellungsanforderung angegeben Zahl von tagen zum aktuellen Zeitpunkt. Anschließend wird die so berechnete Zeit auf den nächsten Tag um Mitternacht (UTC) aufgerundet. Ein Beispiel: Angenommen, ein Objekt wurde am 15. Oktober 2012 um 10:30 Uhr UTC erstellt, und der

Wiederherstellungszeitraum wurde mit drei Tagen angegeben. In diesem Fall läuft die wiederhergestellte Kopie am 19. Oktober 2012 um 00:00 Uhr UTC ab, zu welchem Zeitpunkt Amazon S3 die Objektkopie löscht.

Nachdem Sie eine temporäre Kopie des wiederhergestellten Objekts erhalten haben, bleibt dessen Speicherklasse S3 Glacier oder S3 Glacier Deep Archive. (Eine -HEAD-Objekt- oder die GET-Objekt-API-Operationsanforderung gibt S3 Glacier oder S3 Glacier Deep Archive als Speicherklasse zurück.).

Die für einen Wiederherstellungsauftrag benötigte Zeit hängt von der verwendeten Archivierungsspeicherklasse und vom angegebenen Abrufverfahren ab: `Expedited`, (nur für S3 Glacier verfügbar), `Standard` oder `Bulk`. Sie können über Amazon S3-Benachrichtigungen über den Abschluss der Wiederherstellung benachrichtigt werden. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).

Sie können eine Objektkopie für beliebig viele Tage wiederherstellen. Sie sollten jedoch Objekte nur für die benötigte Dauer wiederherstellen, weil für die Objektkopie Speicherkosten anfallen. Wenn Sie ein Archiv wiederherstellen, bezahlen Sie sowohl das Archiv (zum Tarif S3 Glacier oder S3 Glacier Deep Archive) als auch eine Kopie, die Sie vorübergehend wiederhergestellt haben (Reduced Redundancy Storage, RRS) oder Standard, je nachdem, welcher der kostengünstigere Speicher in der Region ist). Informationen zu Preisen erhalten Sie unter Amazon S3-Preise.

Bei Bedarf können Sie große Segmente der in den Speicherklassen S3 Glacier und S3 Glacier Deep Archive gespeicherten Daten wiederherstellen. So können Sie beispielsweise Daten für eine sekundäre Kopie wiederherstellen. Wenn Sie jedoch sehr viele Daten wiederherstellen müssen, müssen Sie bedenken, dass die Speicherklassen S3 Glacier und S3 Glacier Deep Archive für 35 zufällige Wiederherstellungsanforderungen pro gespeichertem Pepibyte (PiB) pro Tag ausgelegt sind.

Informationen zur Verwendung von Lebenszyklusübergängen zum Verschieben von Objekten zur Speicherklasse S3 Glacier oder S3 Glacier Deep Archive finden Sie unter Übergang zu den Speicherklassen S3 Glacier und S3 Glacier Deep Archive (Objektarchivierung) (p. 144).

Zur Wiederherstellung von mehr als einem Amazon S3-Objekt in einer einzigen Anforderung können Sie S3 Stapeloperationen-Stapelvorgänge nutzen. Sie stellen S3 Stapeloperationen eine Liste der Objekte zur Verfügung, für die Operationen aufgeführt werden sollen. S3 Stapeloperationen rufen die entsprechende API auf, um die angegebene Operation auszuführen. Eine einzelne S3 Stapeloperationen-Aufgabe kann die festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen.

S3 Stapeloperationen-Stapeloperationen verfolgen den Fortschritt nach, senden Benachrichtigungen und speichern einen detaillierten Abschlussbericht zu allen Aktionen. So erhalten Sie eine vollständig verwaltete, prüfbare und serverlose Umgebung. Sie können S3 Stapeloperationen über die AWS Management Console, die AWS CLI, die AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter the section called “Die Grundlagen: Aufträge” (p. 553).

Die folgenden Abschnitte enthalten weitere Informationen zum Wiederherstellen von Objekten.

Themen

Archiv-Abrufoptionen

Nachfolgend finden Sie die verfügbaren Abrufoptionen bei der Wiederherstellung eines archivierten Objekts:

- **`Expedited`** – Beschleunigte Wiederherstellungen ermöglichen einen schnellen Zugriff auf Ihre Daten in der Speicherklasse S3 Glacier, wenn gelegentliche dringende Anfragen für eine Untermenge von Archiven erforderlich sind. Daten, die unter Verwendung von „Expedited (Beschleunigt)“ abgerufen werden, stehen normalerweise innerhalb von 1 bis 5 Minuten zur Verfügung, außer wenn es sich um die größten archivierten Objekte (250 MB+) handelt. Die bereitgestellte Kapazität stellt sicher, dass für Expedited-Abrufe Abrufkapazität verfügbar ist, wenn Sie sie benötigen. Weitere Informationen finden Sie unter Bereitgestellte Kapazität (p. 274). Beschleunigte Wiederherstellung und bereitgestellte Kapazität sind nicht für Objekte in der Speicherklasse S3 Glacier Deep Archive verfügbar.
- **`Standard`** - Standard-Abrufe ermöglichen Ihnen den Zugriff auf Ihre archivierten Objekte innerhalb einiger Stunden. Dies ist die Standardoption für die S3 Glacier- und S3 Glacier Deep Archive-Abrufanforderungen, in denen keine Abrufoption angegeben ist. Standardabrufe benötigen typischerweise 3–5 Stunden für Objekte in der Speicherklasse S3 Glacier. Sie benötigen typischerweise 12 Stunden für Objekte in der Speicherklasse S3 Glacier Deep Archive.
- **`Bulk`** – Massenabrufe sind die kostengünstigste Amazon S3 Glacier-Abrufoption, die Ihnen einen kostengünstigen Abruf großer Mengen von Daten (sogar im Petabyte-Umfang) pro Tag ermöglichen. Massenabrufe benötigen typischerweise 5–12 Stunden für Objekte in der Speicherklasse S3 Glacier. Sie benötigen typischerweise 48 Stunden für Objekte in der Speicherklasse S3 Glacier Deep Archive.

Die folgende Tabelle fasst die Archivabrufoptionen zusammen.

Um einen `Expedited`-, `Standard`- oder `Bulk`-Abruf auszuführen, legen Sie das Anfrageelement `Tier` in der POST Object restore-REST-API-Anfrage auf die gewünschte Option oder die entsprechende Option in der AWS CLI oder den AWS SDKs fest. Wenn Sie bereitgestellte Durchsatzkapazität gekauft haben, werden alle Expedited-Abrufe automatisch über Ihre bereitgestellte Kapazität erledigt.

Sie können archivierte Objekte programmgesteuert oder über die Amazon S3-Konsole wiederherstellen. Amazon S3 verarbeitet jeweils nur eine Wiederherstellungsanfrage pro Objekt. Sie können sowohl die Konsole als auch die Amazon S3-API verwenden, um den Wiederherstellungsstatus zu überprüfen und festzustellen, wann Amazon S3 die wiederhergestellte Kopie löscht.

Bereitgestellte Kapazität

Die bereitgestellte Kapazität stellt sicher, dass für Expedited-Abrufe Abrufkapazität verfügbar ist, wenn Sie sie benötigen. Jede Kapazitätseinheit stellt sicher, dass alle fünf Minuten mindestens drei Expedited-Abrufe ausgeführt werden können, und bietet bis zu 5 MB/s Abrufdurchsatz.

Sie sollten bereitgestellte Abrufkapazität kaufen, wenn Ihr Workload einen sehr zuverlässigen und vorhersehbaren Zugriff auf eine Untermenge Ihrer Daten innerhalb von Minuten erforderlich macht. Ohne bereitgestellte Kapazität werden beschleunigte Abrufe in Zeiten hoher Nachfrage möglicherweise nicht akzeptiert. Wenn Sie unbedingt Zugriff auf beschleunigte Abrufe benötigen, sollten Sie eine bereitgestellte Abrufkapazität kaufen.

Sie können bereitgestellte Kapazität über die Amazon S3-Konsole, die Amazon S3 Glacier-Konsole, die Purchase Provisioned Capacity-REST-API, die AWS SDKs oder die AWS CLI kaufen. Weitere Informationen zu den Preisen für die bereitgestellte Kapazität finden Sie in der Amazon S3-Preisliste.

Beschleunigte Abrufe mit bereitgestellter Kapazität unterliegen dennoch Anforderungs- und Abrufgebühren und sind für die Speicherklasse S3 Glacier Deep Archive nicht verfügbar.

Upgraden der Geschwindigkeit einer Wiederherstellung in Bearbeitung

Mittels eines Upgrades der Amazon S3-Wiederherstellungsgeschwindigkeit können Sie die Wiederherstellungsgeschwindigkeit beschleunigen, während die Wiederherstellung in Bearbeitung ist. Ein Upgrade der Wiederherstellungsgeschwindigkeit überschreibt eine Wiederherstellung in Bearbeitung mit einer schnelleren Wiederherstellungsstufe. Sie können die Geschwindigkeit einer Wiederherstellung in Bearbeitung nicht reduzieren.

Durch die Ausgabe einer weiteren Wiederherstellungsanfrage für dasselbe Objekt können Sie die Geschwindigkeit einer Wiederherstellung in Bearbeitung beschleunigen. Dabei legen Sie in der POST Object restore-REST-API oder entsprechend in der AWS CLI oder den AWS SDKs ein neues `Tier`-Anforderungselement fest. Bei der Ausgabe einer Anfrage für ein Upgrade der Wiederherstellungsstufe müssen Sie eine Stufe wählen, die schneller als die von der Wiederherstellung in Bearbeitung verwendete Stufe ist. Sie dürfen keine anderen Parameter ändern, beispielsweise das Anforderungselement `Days`.

Sie können über Amazon S3-Benachrichtigungen über den Abschluss der Wiederherstellung benachrichtigt werden. Wiederherstellungen werden zum Preis der aktualisierten Ebene berechnet. Informationen zu Wiederherstellungspreisen finden Sie unter Amazon S3-Preise.

Wiederherstellung eines archivierten Objekts mit der Amazon S3-Konsole

Sie können die Amazon S3-Konsole verwenden, um eine Kopie eines Objekts wiederherzustellen, das zu Amazon S3 Glacier archiviert wurde. Anweisungen zum Wiederherstellen eines Archivs unter Verwendung der AWS Management Console finden Sie unter Wie stelle ich ein S3-Objekt wieder her, das in Amazon S3 Glacier archiviert wurde? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Bei einer Wiederherstellung eines Archivs zahlen Sie sowohl für das Archiv, als auch für die temporär wiederhergestellte Kopie. Informationen zu Preisen finden Sie unter Amazon S3-Preise.

Wiederherstellung eines archivierten Objekts mit AWS SDK for Java

Example

Das folgende Beispiel zeigt die Wiederherstellung eines zu Amazon S3 Glacier archivierten Objekts unter Verwendung des AWS SDK for Java. Das Beispiel initiiert eine Wiederherstellungsanfrage für das angegebene archivierte Objekt und überprüft seinen Wiederherstellungsstatus. Weitere Informationen zum Wiederherstellen archivierter Objekte finden Sie unter Wiederherstellen archivierter Objekte (p. 272). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.RestoreObjectRequest;

import java.io.IOException;

public class RestoreArchivedObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Create and submit a request to restore an object from Glacier for two days.
```

```
            RestoreObjectRequest requestRestore = new RestoreObjectRequest(bucketName,
 keyName, 2);
            s3Client.restoreObjectV2(requestRestore);

            // Check the restoration status of the object.
            ObjectMetadata response = s3Client.getObjectMetadata(bucketName, keyName);
            Boolean restoreFlag = response.getOngoingRestore();
            System.out.format("Restoration status: %s.\n",
                    restoreFlag ? "in progress" : "not in progress (finished or failed)");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Wiederherstellung eines archivierten Objekts mit AWS SDK für .NET

Example

Im folgenden C#-Beispiel wird eine Anfrage zur Wiederherstellung eines archivierten Objekts für 2 Tage initiiert. Amazon S3 bewahrt den Wiederherstellungsstatus in den Objektmetadaten. Nachdem die Anfrage initiiert wurde, ruft das Beispiel die Objekt-Metadaten ab und überprüft den Wert der Eigenschaft `RestoreInProgress`. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class RestoreArchivedObjectTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string objectKey = "** archived object key name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            RestoreObjectAsync(client, bucketName, objectKey).Wait();
        }

        static async Task RestoreObjectAsync(IAmazonS3 client, string bucketName, string
 objectKey)
        {
            try
            {
                var restoreRequest = new RestoreObjectRequest
                {
```

```
                    BucketName = bucketName,
                    Key = objectKey,
                    Days = 2
                };
                RestoreObjectResponse response = await
 client.RestoreObjectAsync(restoreRequest);

                // Check the status of the restoration.
                await CheckRestorationStatusAsync(client, bucketName, objectKey);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }

        static async Task CheckRestorationStatusAsync(IAmazonS3 client, string bucketName,
 string objectKey)
        {
            GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest
            {
                BucketName = bucketName,
                Key = objectKey
            };
            GetObjectMetadataResponse response = await
 client.GetObjectMetadataAsync(metadataRequest);
            Console.WriteLine("restoration status: {0}", response.RestoreInProgress ? "in-
progress" : "finished or failed");
        }
    }
}
```

Wiederherstellung eines archivierten Objekts mit der REST API

Amazon S3 stellt eine API für Sie bereit, um eine Wiederherstellung des Archivs zu initiieren. Weitere Informationen finden Sie unter POST Object restore im Amazon Simple Storage Service API Reference.

Abfrage von archivierten Objekten

Sie können mit dem Select-Typ von POST Object restore Filteroperationen mit einfachen Structured Query Language (SQL)-Anweisungen direkt für die Daten ausführen, die von Amazon S3 in S3 Glacier archiviert wurden. Wenn Sie eine SQL-Abfrage für ein Archivobjekt bereitstellen, führt Select die Abfrage durch und schreibt die Ausgabeergebnisse in einen S3-Bucket. Sie können Abfragen und benutzerdefinierte Analysen für in S3 Glacier gespeicherte Daten ausführen, ohne Ihr gesamtes Objekt in Amazon S3 wiederherstellen zu müssen.

Wenn Sie select-Abfragen ausführen, stellt S3 Glacier drei Datenzugriffsstufen bereit: expedited, standard und bulk. Für diese Kontingente stehen jeweils eigene Datenzugriffszeiten und -kosten zur Verfügung. Wählen Sie ein passendes Kontingent aus, je nachdem, wie schnell Daten verfügbar sein sollen. Weitere Informationen finden Sie unter Datenzugriffsstufen (p. 280).

Sie können den select-Wiederherstellungstyp mit den AWS SDKs, der S3 Glacier-REST-API und der AWS Command Line Interface (AWS CLI) verwenden.

Themen

Voraussetzungen und Beschränkungen für Select

Für die Verwendung von Select müssen folgende Voraussetzungen erfüllt sein:

- Archivobjekte, die mit Select abgerufen werden, müssen als unkomprimierte CSV-Datei vorliegen.
- Es muss ein S3-Bucket für die Ausgabe vorhanden sein. Das AWS-Konto, mit dem Sie einen S3 Glacier-Select Job starten, muss über Schreibrechte für den S3-Bucket verfügen. Der Amazon S3-Bucket muss sich in derselben AWS-Region befinden wie der Bucket, der das abzurufende Archivobjekt enthält.
- Das abfragende AWS-Konto muss über Berechtigungen verfügen, um die `s3:RestoreObject`- und `s3:GetObject`-Aktionen durchzuführen. Weitere Informationen zu diesen Berechtigungen finden Sie unter Beispiel – Bucket-Subressourcen-Operationen (p. 381).
- Das Archiv darf nicht mit SSE-C oder clientseitiger Verschlüsselung verschlüsselt werden.

Für die Verwendung von Select gelten die folgenden Beschränkungen:

- Es gibt keine Beschränkungen für die Anzahl der Datensetze, die von Select verarbeitet werden können. Ein Ein- oder Ausgabedatensatz darf höchstens 1 MB groß sein, andernfalls schlägt die Abfrage fehl. Es gilt eine Beschränkung von 1.048.576 Spalten pro Datensatz.
- Es gibt keine Beschränkung der Größe der Endergebnisse. Die Ergebnisse werden jedoch in mehrere Teile aufgeteilt.
- Ein SQL-Ausdruck ist auf 128 KB beschränkt.

Wie kann ich Daten mit Select abfragen?

Sie können mit Select SQL-Befehle nutzen, um S3 Glacier-Archivobjekte abzurufen, die im unverschlüsselten unkomprimierten CSV-Format vorliegen. Mit dieser Einschränkung können Sie einfache Abfrageoperationen für textbasierte Daten in S3 Glacier ausführen. Sie können beispielsweise nach einem bestimmten Namen oder einer bestimmten ID innerhalb einer Menge von archivierten Textdateien suchen.

Um S3 Glacier-Daten abzufragen, erstellen Sie unter Verwendung der Operation POST Object restore eine select-Anfrage. Wenn Sie eine Select-Abfrage durchführen, geben Sie den SQL-Ausdruck, das abzufragende Archiv sowie den Speicherort für die Ergebnisse an.

Mit dem folgenden Beispielausdruck werden alle Datensätze aus dem archivierten Objekt zurückgegeben, die in POST Object restore angegeben sind.

```
SELECT * FROM object
```

S3 Glacier Select unterstützt eine Untermenge der ANSI SQL-Sprache. Es werden gängige SQL-Filterklauseln wie `SELECT`, `FROM` und `WHERE` unterstützt. Es werden keine `SUM`, `COUNT`, `GROUP BY`, `JOINS`, `DISTINCT`, `UNION`, `ORDER BY` und `LIMIT` unterstützt. Weitere Informationen zur Unterstützung von SQL finden Sie unter SQL-Referenz für Amazon S3 Select und S3 Glacier Select im Entwicklerhandbuch für Amazon Simple Storage Service.

Select-Ausgabe

Wenn Sie eine Select-Abfrage initiieren, legen Sie einen Ausgabespeicherort für die Ergebnisse der Select-Abfrage fest. Dieser Speicherort muss ein Amazon S3-Bucket in derselben AWS-Region sein wie der

Bucket, der das abzurufende Archivobjekt enthält. Das AWS-Konto, von dem aus der Auftrag initiiert wird, muss über Schreibberechtigungen für den S3-Bucket verfügen.

Sie können die Amazon S3-Speicherklasse und -Verschlüsselung für in Amazon S3 gespeicherte Ausgabeobjekte festlegen. Wählen Sie die Unterstützung der AWS Key Management Service (SSE-KMS) und Amazon S3 (SSE-S3) Verschlüsselung aus. Die SSE-C- und clientseitige Verschlüsselung wird von Select nicht unterstützt. Weitere Informationen zu Amazon S3-Speicherklassen und zur S3-Verschlüsselung finden Sie unter Amazon S3-Speicherklassen (p. 122) und Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290).

S3 Glacier Select-Ergebnisse werden im S3-Bucket unter Verwendung des Präfixes gespeichert, das in dem in POST Object restore angegebenen Ausgabespeicherort bereitgestellt wurde. Anhand dieser Informationen erstellt Select ein eindeutiges Präfix mit Verweis auf die Auftrags-ID. (Präfixe werden verwendet, um Amazon S3-Objekte anhand eines identischen Objektnamensanfangs zu gruppieren.) Unter diesem eindeutigen Präfix werden zwei neue Präfixe erstellt, `results` für Ergebnisse und `errors` für Protokolle und Fehler. Nach Abschluss des Auftrags wird ein Ergebnismanifest mit dem Speicherort aller Ergebnisse geschrieben.

Außerdem wird die Platzhalterdatei `job.txt` in das Ausgabeverzeichnis geschrieben. Nach dem Schreiben wird nie aktualisiert. und dient folgenden Zwecken:

- Synchrone Validierung der Schreibberechtigung und der meisten SQL-Syntaxfehler
- Bereitstellung einer statischen Ausgabe über Ihre Select-Anfrage, auf die Sie jederzeit einfach verweisen können.

Angenommen, Sie nehmen eine Select-Abfrage mit dem Ausgabespeicherort für die Ergebnisse `s3://example-bucket/my-prefix` vor und die Auftragsantwort gibt die Auftrags-ID als `examplekne1209ualkdjh812elkassdu9012e` zurück. Nach Abschluss des Auswahlauftrags werden in Ihrem Bucket die folgenden Amazon S3-Objekte angezeigt:

```
s3://example-bucket/my-prefix/examplekne1209ualkdjh812elkassdu9012e/job.txt
s3://example-bucket/my-prefix/examplekne1209ualkdjh812elkassdu9012e/results/abc
s3://example-bucket/my-prefix/examplekne1209ualkdjh812elkassdu9012e/results/def
s3://example-bucket/my-prefix/examplekne1209ualkdjh812elkassdu9012e/results/ghi
s3://example-bucket/my-prefix/examplekne1209ualkdjh812elkassdu9012e/result_manifest.txt
```

Die Auswahlabfrageergebnisse werden in mehrere Teile aufgeteilt. In dem Beispiel verwendet Select das Präfix, das beim Festlegen des Ausgabespeicherorts eingerichtet wurde, und hängt die Auftrags-ID an das Präfix `results` an. Dann wird das Ergebnis in drei Teilen geschrieben, wobei die Objektnamen in `abc`, `def` und `ghi` enden. Das Ergebnismanifest enthält alle drei Dateien für einen programmgesteuerten Abruf. Wenn der Auftrag mit einem Fehler fehlschlägt, wird eine Datei mit dem Fehlerpräfix angezeigt und es wird eine Datei `error_manifest.txt` erstellt.

Wenn die Datei `result_manifest.txt` vorhanden ist und die Datei `error_manifest.txt` fehlt, wurde der Auftrag auf jeden Fall erfolgreich abgeschlossen. Für die Anordnung der Ergebnisse gibt es jedoch keine Garantie.

> Note
>
> Die Länge eines Amazon S3-Objektnamens, auch als Schlüssel bezeichnet, darf 1.024 Bytes nicht überschreiten. S3 Glacier select reserviert 128 Bytes für Präfixe. Die Länge des Amazon S3-Speicherpfads darf 512 Bytes nicht überschreiten. Bei einer Anforderung mit einer Länge über 512 Bytes wird eine Ausnahme zurückgegeben und die Anforderung wird nicht akzeptiert.

Fehlerbehandlung

Select benachrichtigt Sie über zwei Arten von Fehlern. Der erste Satz von Fehlern wird synchron an Sie gesendet, wenn Sie die Abfrage in POST Object restore absenden. Diese Fehler werden im Rahmen der

HTTP-Antwort gesendet. Der zweite Fehlertyp kann auftreten, nachdem die Abfrage erfolgreich akzeptiert wurde, wenn dann bei der Ausführung der Abfrage ein Fehler auftritt. In diesem Fall werden die Fehler in den unter dem `errors`-Präfix angegebenen Ausgabespeicherort geschrieben.

Select stoppt die Ausführung der Abfrage, wenn ein Fehler auftritt. Um die Abfrage erfolgreich ausführen zu können, müssen Sie alle Fehler beheben. Welche Datensätze zu einem Fehler geführt haben, können Sie den Protokollen entnehmen.

Da Abfragen parallel auf mehreren Rechenknoten ausgeführt werden, werden die Fehler nicht in Reihenfolge angezeigt. Wenn eine Abfrage beispielsweise mit einem Fehler in Zeile 6.234 scheitert, bedeutet das nicht, dass alle Zeilen vor Zeile 6.234 erfolgreich verarbeitet wurden. Beim nächsten Durchlauf der Abfrage kann der Fehler an anderer Stelle angezeigt werden.

Datenzugriffsstufen

Sie können bei der Abfrage eines archivierten Objekts eine der folgenden Datenzugriffsstufen verwenden:

- **`Expedited`** – Ermöglicht Ihnen den schnellen Zugriff auf Ihre Daten, wenn gelegentliche dringende Zugriffe für eine Teilmenge der Archive erforderlich sind. Daten, auf die unter Verwendung von `Expedited`-Abfragen zugegriffen wird, stehen normalerweise innerhalb von 1 bis 5 Minuten zur Verfügung, außer wenn es sich um das größte archivierte Objekt (über 250 MB) handelt. Es gibt zwei Arten von `Expedited`-Datenzugriffen: „On-Demand” (bedarfsweise) und „Provisioned” (bereitgestellt). On-Demand-Anforderungen sind vergleichbar mit EC2 On-Demand-Instances und stehen fast jederzeit zur Verfügung. Provisioned-Anforderungen stehen garantiert zur Verfügung, wenn Sie sie benötigen. Weitere Informationen finden Sie unter Bereitgestellte Kapazität (p. 280).
- **`Standard`** – Lässt den Zugriff auf Ihre archivierten Objekte innerhalb einiger Stunden zu. Standard-Abrufe werden in der Regel innerhalb von 3 bis 5 Stunden beendet. Dies ist die Standardstufe.
- **`Bulk`** – Die kostengünstigste Datenzugriffsoption in S3 Glacier, die Ihnen den Abruf großer Mengen von Daten (sogar Petabytes) pro Tag ermöglicht. Ein `Bulk`-Zugriff wird in der Regel innerhalb von 5 bis 12 Stunden abgeschlossen.

Um eine `Expedited`-, `Standard`- oder `Bulk`-Abfrage auszuführen, legen Sie das Anfrageelement `Tier` in der POST Object restore-REST-API-Anfrage auf die gewünschte Option oder die entsprechende Option in der AWS CLI oder den AWS SDKs fest. Für einen `Expedited`-Zugriff müssen Sie nicht angeben, ob ein „Expedited”-Abruf „On-Demand” oder „Provisioned” ist. Wenn Sie bereitgestellte Durchsatzkapazität gekauft haben, werden alle `Expedited`-Abrufe automatisch über Ihre bereitgestellte Kapazität erledigt. Weitere Informationen zu Stufenpreisen finden Sie unter S3 Glacier-Preise.

Bereitgestellte Kapazität

Bereitgestellte Kapazität garantiert, dass Ihre Abrufkapazität für Expedited-Abrufe verfügbar ist, wenn Sie sie benötigen. Jede Kapazitätseinheit stellt sicher, dass alle fünf Minuten mindestens drei Expedited-Abrufe durchgeführt werden können, und bietet bis zu 150 MB/s Abrufdurchsatz.

Sie sollten bereitgestellte Abrufkapazität kaufen, wenn Ihre Workload einen sehr zuverlässigen und vorhersehbaren Zugriff auf eine Untermenge Ihrer Daten innerhalb von Minuten erforderlich macht. Ohne bereitgestellte Kapazität werden `Expedited`-Abrufe akzeptiert, außer in seltenen Situationen unüblich hoher Nachfrage. Wenn Sie unbedingt Zugriff auf `Expedited`-Abrufe benötigen, müssen Sie auf alle Fälle eine bereitgestellte Abrufkapazität kaufen. Sie können bereitgestellte Kapazität über die Amazon S3-Konsole, die S3 Glacier-Konsole, die Purchase Provisioned Capacity-REST-API, die AWS SDKs oder die AWS CLI kaufen. Informationen zu den Preisen für bereitgestellte Kapazitäten finden Sie in der S3 Glacier-Preisliste.

Weitere Infos

- POST Object restore im Amazon Simple Storage Service API Reference

- SQL-Referenz für Amazon S3 Select und S3 Glacier Select im Entwicklerhandbuch für Amazon Simple Storage Service

Amazon S3-Analyse – Speicherklassenanalyse

Mit dem Tool für die Speicherklassenanalyse von Amazon S3 können Sie Speicherzugriffsmuster analysieren, um zu entscheiden, ob Sie die richtigen Daten in die richtige Speicherklasse einordnen. Diese neue analytische Funktion von Amazon S3 beobachtet Datenzugriffsmuster, anhand derer Sie entscheiden können, wann Sie STANDARD-Speicher mit weniger häufigem Zugriff in die Speicherklasse STANDARD_IA (IA für "infrequent access" (seltener Zugriff)) überführen sollen. Weitere Informationen über Speicherklassen finden Sie unter Amazon S3-Speicherklassen (p. 122).

Wenn die Speicherklassenanalyse die seltenen Datenzugriffsmuster für eine gefilterte Datenmenge im Laufe der Zeit beobachtet, können Sie Ihre Lebenszyklusrichtlinien unter Verwendung der Analyseergebnisse verbessern. Sie können die Speicherklassenanalyse so konfigurieren, dass alle Objekte in einem Bucket analysiert werden. Sie können aber auch Filter konfigurieren, um Objekte nach einem gemeinsamen Präfix zu gruppieren (d. h. für Objekte, deren Namen mit einer gemeinsamen Zeichenkette beginnen), nach Objekt-Tags oder sowohl nach Präfix und Tags. Wahrscheinlich werden Sie feststellen, dass die Filterung nach Objektgruppen die beste Methode darstellt, von der Speicherklassenanalyse zu profitieren.

> Important
>
> Die Speicherklassenanalyse gibt keine Empfehlungen für Übergänge in die Speicherklasse ONEZONE_IA oder S3 Glacier.

Sie können mehrere Speicherklassenanalysefilter pro Bucket einrichten, bis zu 1000, und erhalten für jeden Filter eine separate Analyse. Durch Verwendung mehrerer Filterkonfigurationen können Sie spezifische Objektgruppen analysieren, um Ihre Lebenszyklusrichtlinien zu verbessern, die Objekte in STANDARD_IA überführen.

Die Speicherklassenanalyse zeigt Schaubilder zur Speichernutzung in der Amazon S3-Konsole, die täglich aktualisiert werden. Die Speichernutzungsdaten können auch täglich in eine Datei in einem S3-Bucket exportiert werden. Sie können die exportierte Nutzungsberichtdatei in eine Tabellenkalkulation exportieren oder sie mit Business Intelligence-Anwendungen Ihrer Wahl bearbeiten, wie beispielsweise Amazon QuickSight.

Mit der Speicherklassenanalyse sind Kosten verbunden. Preisinformationen finden Sie unter Verwaltung und Replikation Amazon S3-Preise.

Themen

Wie richte ich die Speicherklassenanalyse ein?

Sie können die Speicherklassenanalyse Einrichten, indem Sie konfigurieren, welche Objektdaten analysiert werden sollen. Sie können die Speicherklassenanalsyse für die folgenden Aufgaben konfigurieren:

- Analyse des gesamten Inhalts eines Buckets.

  Sie erhalten eine Analyse für alle Objekte im Bucket.

- Analyse von nach Präfix und Tags gruppierten Objekten.

  Sie können Filter konfigurieren, um Objekte nach einem gemeinsamen Präfix oder nach Objekt-Tags zu gruppieren, oder nach einer Kombination aus Präfix und Tags. Sie erhalten für jeden konfigurierten Filter eine separate Analyse. Sie können maximal 1000 Filterkonfigurationen pro Bucket verwenden.
- Export von Analysedaten.

  Wenn Sie die Speicherklassenanalyse für einen Bucket oder Filter konfigurieren, können Sie die Analysedaten täglich in eine Datei exportieren lassen. Die Analyse für den Tag wird der Datei hinzugefügt, um ein Verlaufsanalyseprotokoll für den konfigurierten Filter zu erstellen. Die Datei wird täglich an dem Ziel Ihrer Wahl aktualisiert. Wenn Sie Dateien zum Export auswählen, geben Sie einen Ziel-Bucket und ein optionales Ziel-Präfix an, wohin die Datei geschrieben werden soll.

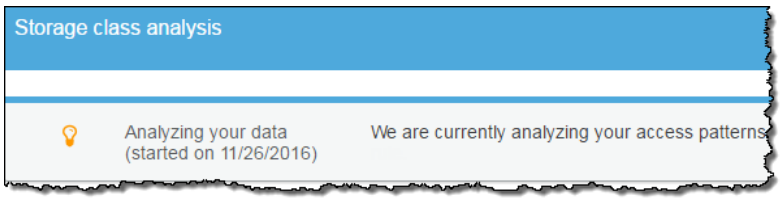
Sie können die Amazon S3-Konsole, die REST-API, die AWS CLI oder die AWS-SDKs verwenden, um die Speicherklassenanalyse zu konfigurieren.

- Weitere Informationen über die Konfiguration der Speicherklassenanalyse in der Amazon S3-Konsole finden Sie unter Wie konfiguriere ich die Speicherklassenanalyse?
- Für die Nutzung der Amazon S3-API verwenden Sie die PutBucketAnalyticsConfiguration-REST-API oder das Äquivalent aus der AWS CLI oder die AWS-SDKs.

Wie verwende ich die Speicherklassenanalyse?

Sie verwenden die Speicherklassenanalyse, um Datenzugriffsmuster im Laufe der Zeit zu analysieren, um Informationen zu erhalten, die Ihnen beim Lebenszyklusmanagement Ihres STANDARD_IA-Speichers helfen. Nachdem Sie einen Filter konfiguriert haben, sehen Sie die Datenanalyse basierend auf dem Filter in der Amazon S3-Konsole über 24 bis 48 Stunden. Die Speicherklassenanalyse beobachtet jedoch die Zugriffsmuster einer gefilterten Datenmenge für 30 Tage oder Länger, um Informationen für eine Analyse zu sammeln, bevor ein Ergebnis zurückgegeben wird. Die Analyse wird nach dem ersten Ergebnis fortgesetzt und aktualisiert das Ergebnis, wenn sich die Zugriffsmuster ändern.

Wenn Sie einen Filter zum ersten Mal konfigurieren, zeigt die Amazon S3-Konsole eine Meldung ähnlich der folgenden angezeigt:



Die Speicherklassenanalyse beobachtet die Zugriffsmuster einer gefilterten Objekt-Datenmenge für 30 Tage oder Länger, um ausreichend viele Informationen für eine Analyse zu sammeln. Nachdem die Speicherklassenanalyse genügend Informationen gesammelt hat, sehen Sie etwa die folgende Meldung in der Amazon S3-Konsole.

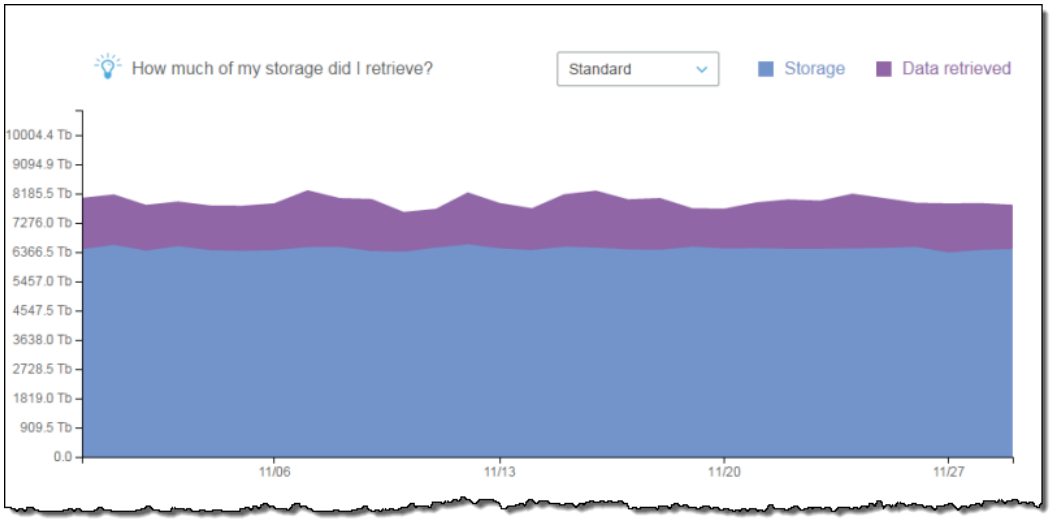
Bei der Durchführung der Analyse für Objekte mit seltenem Zugriff betrachtet die Speicherklassenanalyse die gefilterte gruppierte Objektmenge basierend auf dem Alter, wie lange sie in Amazon S3 hochgeladen

wurden. Die Speicherklassenanalyse stellt fest, ob für die Altersgruppe selten ein Zugriff stattfindet, indem sie die folgenden Faktoren für die gefilterte Datenmenge betrachtet.

- Objekte in der Speicherklasse STANDARD, die größer als 128 KB sind.
- Wie viel durchschnittlichen Gesamtspeicher Sie pro Altersgruppe haben.
- Die durchschnittliche Anzahl an Bytes, die pro Altersgruppe nach außen übertragen werden (nicht die Frequenz).
- Analyseexportdaten enthalten nur Anfragen mit Daten, die für die Speicherklassenanalyse relevant sind. Dies kann Differenzen in der Anzahl der Anfragen sowie im Hinblick auf die gesamten Bytes für Upload und Anfrage im Vergleich zu den Angaben in der Speichermetrik oder in den Ergebnissen Ihrer eigenen internen Systeme verursachen.
- Fehlgeschlagene GET- und PUT-Anfragen werden bei der Analyse nicht berücksichtigt. In den Speichermetriken sehen Sie jedoch fehlgeschlagene Anfragen.

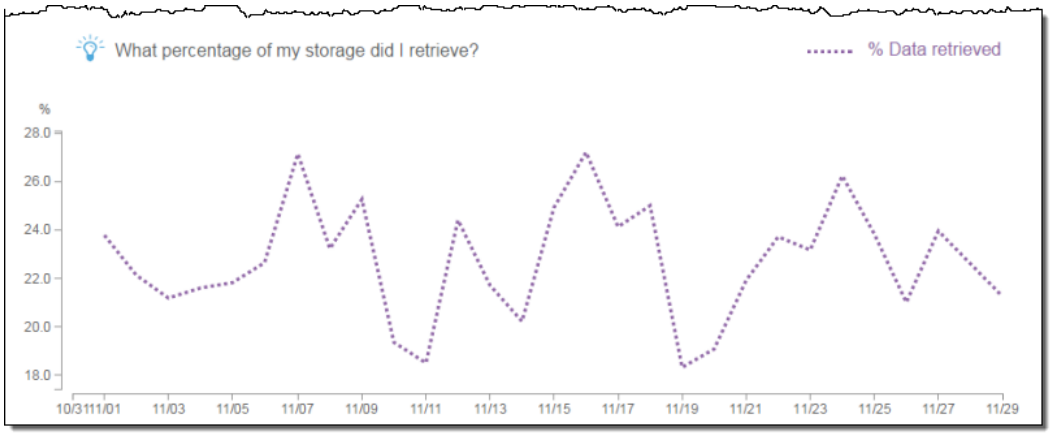
Wie viel von meinem Speicher habe ich abgerufen?

Die Amazon S3-Konsole zeigt in einer Grafik an, wie viel von dem Speicher in der gefilterten Datenmenge für den Beobachtungszeitraum abgerufen wurde, wie im folgenden Beispiel gezeigt.



Welchen Prozentsatz meines Speichers habe ich abgerufen?

Die Amazon S3-Konsole zeigt in einer Grafik an, welcher Prozentsatz von dem Speicher in der gefilterten Datenmenge für den Beobachtungszeitraum abgerufen wurde, wie im folgenden Beispiel gezeigt.



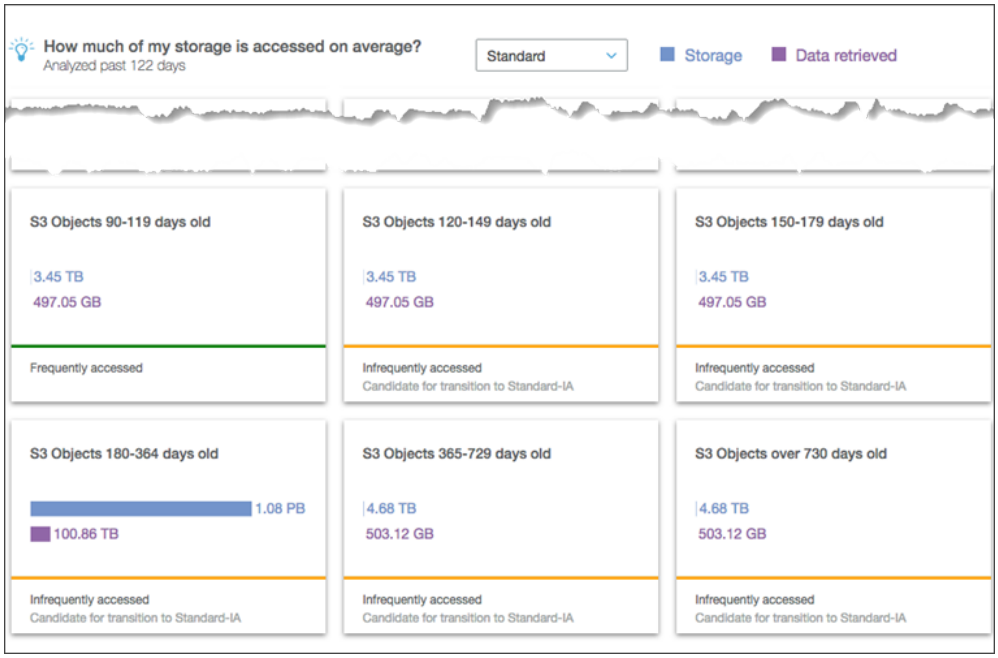
Wie in diesem Thema bereits dargelegt, betrachtet die Speicherklassenanalyse bei der Analyse für Objekte mit seltenem Zugriff die gefilterte Objektemenge, gruppiert nach dem Alter, seit die Objekte in Amazon S3 hochgeladen wurden. Die Speicherklassenanalyse verwendet die folgenden vordefinierten Objektaltersgruppen:

- Amazon S3-Objekte, die weniger als 15 Tage alt sind
- Amazon S3-Objekte, 15 bis 29 Tage alt
- Amazon S3-Objekte, 30 bis 44 Tage alt
- Amazon S3-Objekte, 45 bis 59 Tage alt
- Amazon S3-Objekte, 60 bis 74 Tage alt
- Amazon S3-Objekte, 75 bis 89 Tage alt
- Amazon S3-Objekte, 90 bis 119 Tage alt
- Amazon S3-Objekte, 120 bis 149 Tage alt
- Amazon S3-Objekte, 150 bis 179 Tage alt
- Amazon S3-Objekte, 180 bis 364 Tage alt
- Amazon S3-Objekte, 365 bis 729 Tage alt
- Amazon S3-Objekte, 730 Tage alt und älter

In der Regel benötigt sie über 30 Tage beobachtete Zugriffsmuster, um ausreichend viele Informationen für ein Analyseergebnis zu sammeln. Abhängig von dem spezifischen Zugriffsmuster Ihrer Daten könnte dies länger als 30 Tage dauern. Nachdem Sie jedoch einen Filter konfiguriert haben, sehen Sie die Datenanalyse basierend auf dem Filter in der Amazon S3-Konsole über 24 bis 48 Stunden. Sie sehen die Analyse zum Objektzugriff auf täglicher Basis, unterteilt nach Objektaltersgruppe in der Amazon S3-Konsole.

Wie viel von meinem Speicher wird selten abgerufen?

Die Amazon S3-Konsole zeigt die Zugriffsmuster, gruppiert nach den vordefinierten Objektaltersgruppen, wie im obigen Beispiel gezeigt.



Der angezeigte Text Frequently accessed (Häufig aufgerufen) oder Infrequently accessed (Selten aufgerufen) ist als visuelle Hilfe bei der Erstellung des Lebenszyklus gedacht.

Wie kann ich die Daten der Speicherklassenanalyse exportieren?

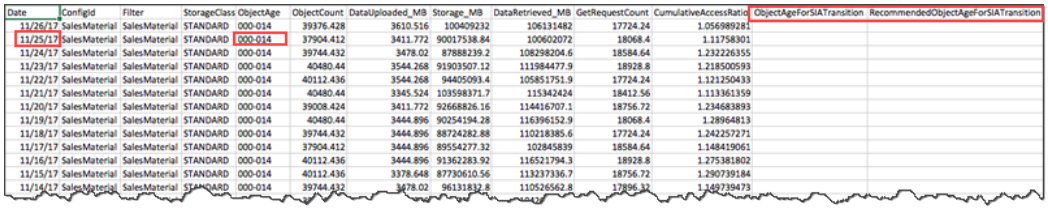
Sie können festlegen, dass die Speicherklassenanalyse Analyseberichte in einer flachen .csv-Datei (durch Kommas getrennte Werte) exportiert. Berichte werden täglich aktualisiert und basieren auf den von Ihnen konfigurierten Objektaltersgruppenfiltern. Bei Verwendung der Amazon S3-Konsole können Sie die Exportberichtsoption wählen, wenn Sie einen Filter erstellen. Wenn Sie Dateien zum Export auswählen, geben Sie einen Ziel-Bucket und ein optionales Ziel-Präfix an, wohin die Datei geschrieben werden soll. Sie können die Daten in einen Ziel-Bucket in einem anderen Konto exportieren. Der Ziel-Bucket muss sich in derselben Region befinden wie der Bucket, dessen Analyse Sie konfiguriert haben.

Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket einrichten, um Amazon S3 die Berechtigung zu erteilen, zu überprüfen, welchem AWS-Konto der Bucket gehört, und Objekte in den Bucket an dem definierten Standort zu schreiben. Eine Beispielrichtlinie finden Sie unter Gewähren von Berechtigungen für Amazon S3-Bestand und Amazon S3-Analysen (p. 453).

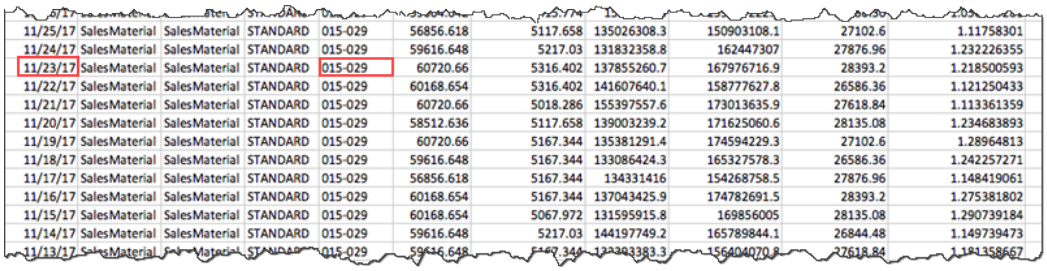
Nachdem Sie Speicherklassenanalyseberichte konfiguriert haben, erhalten Sie den exportierten Bericht nach 24 Stunden täglich. Anschließend setzt Amazon S3 die Überwachung fort und stellt tägliche Berichte bereit.

Sie können die CSV-Datei in einer Tabellenkalkulation öffnen oder die Datei in andere Anwendungen importieren, wie beispielsweise Amazon QuickSight. Weitere Informationen über die Verwendung von Amazon S3-Dateien mit Amazon QuickSight finden Sie unter Erstellen eines Datasets mit Amazon Amazon S3-Dateien im Amazon QuickSight-Benutzerhandbuch.

Daten in den exportierten Dateien werden nach dem Datum innerhalb der Objektaltersgruppe sortiert, wie in den folgenden Beispielen gezeigt. Wenn die Speicherklasse STANDARD ist, enthält die Zeile Daten für die Spalten `ObjectAgeForSIATransition` und `RecommendedObjectAgeForSIATransition`.

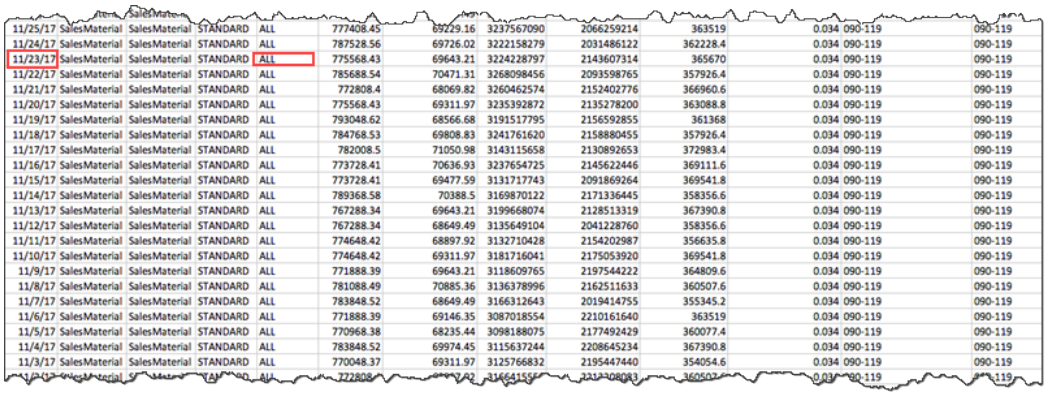


Date	ConfigId	Filter	StorageClass	ObjectAge	ObjectCount	DataUploaded_MB	Storage_MB	DataRetrieved_MB	GetRequestCount	CumulativeAccessRatio	ObjectAgeForSIATransition	RecommendedObjectAgeForSIATransition
11/26/17	SalesMaterial	SalesMaterial	STANDARD	000-014	39376.428	3610.516	100409232	106131482	17724.24	1.056989281		
11/25/17	SalesMaterial	SalesMaterial	STANDARD	000-014	37904.412	3411.772	90017538.84	100602072	18068.4	1.11758301		
11/24/17	SalesMaterial	SalesMaterial	STANDARD	000-014	39744.432	3478.02	87888239.2	108298204.6	18584.64	1.232226355		
11/23/17	SalesMaterial	SalesMaterial	STANDARD	000-014	40480.44	3544.268	91903507.12	111984477.9	18928.8	1.218500593		
11/22/17	SalesMaterial	SalesMaterial	STANDARD	000-014	40112.436	3544.268	94405093.4	105851751.9	17724.24	1.121250433		
11/21/17	SalesMaterial	SalesMaterial	STANDARD	000-014	40480.44	3345.524	103598371.7	115342424	18412.56	1.113361359		
11/20/17	SalesMaterial	SalesMaterial	STANDARD	000-014	39008.424	3411.772	92668826.16	114416707.1	18756.72	1.234683893		
11/19/17	SalesMaterial	SalesMaterial	STANDARD	000-014	40480.44	3444.896	90254194.28	116396152.9	18068.4	1.28964813		
11/18/17	SalesMaterial	SalesMaterial	STANDARD	000-014	39744.432	3444.896	88724282.88	110218385.6	17724.24	1.242257271		
11/17/17	SalesMaterial	SalesMaterial	STANDARD	000-014	37904.412	3444.896	89554277.32	102845839	18584.64	1.148419061		
11/16/17	SalesMaterial	SalesMaterial	STANDARD	000-014	40112.436	3444.896	91362283.92	116521794.3	18928.8	1.275381802		
11/15/17	SalesMaterial	SalesMaterial	STANDARD	000-014	40112.436	3378.648	87730610.56	113237336.7	18756.72	1.290739184		
11/14/17	SalesMaterial	SalesMaterial	STANDARD	000-014	39744.432	3478.02	96131832.8	110526562.8	17896.32	1.149739473		



11/25/17	SalesMaterial	SalesMaterial	STANDARD	015-029	56856.618	5117.658	135026308.3	150903108.1	27102.6	1.11758301
11/24/17	SalesMaterial	SalesMaterial	STANDARD	015-029	59616.648	5217.03	131832358.8	162447307	27876.96	1.232226355
11/23/17	SalesMaterial	SalesMaterial	STANDARD	015-029	60720.66	5316.402	137855260.7	167976716.9	28393.2	1.218500593
11/22/17	SalesMaterial	SalesMaterial	STANDARD	015-029	60168.654	5316.402	141607640.1	158777627.8	26586.36	1.121250433
11/21/17	SalesMaterial	SalesMaterial	STANDARD	015-029	60720.66	5018.286	155397557.6	173013635.9	27618.84	1.113361359
11/20/17	SalesMaterial	SalesMaterial	STANDARD	015-029	58512.636	5117.658	139003239.2	171625060.6	28135.08	1.234683893
11/19/17	SalesMaterial	SalesMaterial	STANDARD	015-029	60720.66	5167.344	135381291.4	174594229.3	27102.6	1.28964813
11/18/17	SalesMaterial	SalesMaterial	STANDARD	015-029	59616.648	5167.344	133086424.3	165327578.3	26586.36	1.242257271
11/17/17	SalesMaterial	SalesMaterial	STANDARD	015-029	56856.618	5167.344	134331416	154268758.5	27876.96	1.148419061
11/16/17	SalesMaterial	SalesMaterial	STANDARD	015-029	60168.654	5167.344	137043425.9	174782691.5	28393.2	1.275381802
11/15/17	SalesMaterial	SalesMaterial	STANDARD	015-029	60168.654	5067.972	131595915.8	169856005	28135.08	1.290739184
11/14/17	SalesMaterial	SalesMaterial	STANDARD	015-029	59616.648	5217.03	144197749.2	165789844.1	26844.48	1.149739473
11/13/17	SalesMaterial	SalesMaterial	STANDARD	015-029	59616.648	5167.344	123303383.3	156404070.8	27618.84	1.191358667

Am Ende des Berichts wird die Objektaltersgruppe als ALL angegeben. Die ALL-Zeilen enthalten kumulative Summen, einschließlich Objekte, die kleiner als 128 KB sind, für alle Altersgruppen für diesen Tag.



11/25/17	SalesMaterial	SalesMaterial	STANDARD	ALL	777408.45	69229.16	3237567090	2066259214	363519	0.034	090-119	090-119
11/24/17	SalesMaterial	SalesMaterial	STANDARD	ALL	787528.56	69726.02	3222158279	2031486122	362228.4	0.034	090-119	090-119
11/23/17	SalesMaterial	SalesMaterial	STANDARD	ALL	775568.43	69643.21	3224228797	2143607314	365670	0.034	090-119	090-119
11/22/17	SalesMaterial	SalesMaterial	STANDARD	ALL	785688.54	70471.31	3268098456	2093598765	357926.4	0.034	090-119	090-119
11/21/17	SalesMaterial	SalesMaterial	STANDARD	ALL	772808.4	68069.82	3260462574	2152402776	366960.6	0.034	090-119	090-119
11/20/17	SalesMaterial	SalesMaterial	STANDARD	ALL	775568.43	69311.97	3235392872	2135278200	363088.8	0.034	090-119	090-119
11/19/17	SalesMaterial	SalesMaterial	STANDARD	ALL	793048.62	68566.68	3191517795	2156592855	361368	0.034	090-119	090-119
11/18/17	SalesMaterial	SalesMaterial	STANDARD	ALL	784768.53	69808.83	3241761620	2158880455	357926.4	0.034	090-119	090-119
11/17/17	SalesMaterial	SalesMaterial	STANDARD	ALL	782008.5	71050.98	3143115658	2130892653	372983.4	0.034	090-119	090-119
11/16/17	SalesMaterial	SalesMaterial	STANDARD	ALL	773728.41	70636.93	3237654725	2145622446	369111.6	0.034	090-119	090-119
11/15/17	SalesMaterial	SalesMaterial	STANDARD	ALL	773728.41	69477.59	3131717743	2091869264	369541.8	0.034	090-119	090-119
11/14/17	SalesMaterial	SalesMaterial	STANDARD	ALL	789368.58	70388.5	3169870122	2171336445	358356.6	0.034	090-119	090-119
11/13/17	SalesMaterial	SalesMaterial	STANDARD	ALL	767288.34	69643.21	3199668074	2128513319	367390.8	0.034	090-119	090-119
11/12/17	SalesMaterial	SalesMaterial	STANDARD	ALL	767288.34	68649.49	3135649104	2041228760	358356.6	0.034	090-119	090-119
11/11/17	SalesMaterial	SalesMaterial	STANDARD	ALL	774648.42	68897.92	3132710428	2154202987	356635.8	0.034	090-119	090-119
11/10/17	SalesMaterial	SalesMaterial	STANDARD	ALL	774648.42	69311.97	3181716041	2175053920	369541.8	0.034	090-119	090-119
11/9/17	SalesMaterial	SalesMaterial	STANDARD	ALL	771888.39	69643.21	3118609765	2197544222	364809.6	0.034	090-119	090-119
11/8/17	SalesMaterial	SalesMaterial	STANDARD	ALL	781088.49	70885.36	3136378996	2162511633	360507.6	0.034	090-119	090-119
11/7/17	SalesMaterial	SalesMaterial	STANDARD	ALL	783848.52	68649.49	3166312643	2019414755	355345.2	0.034	090-119	090-119
11/6/17	SalesMaterial	SalesMaterial	STANDARD	ALL	771888.39	69146.35	3087018554	2210161640	363519	0.034	090-119	090-119
11/5/17	SalesMaterial	SalesMaterial	STANDARD	ALL	770968.38	68235.44	3098188075	2177492429	360077.4	0.034	090-119	090-119
11/4/17	SalesMaterial	SalesMaterial	STANDARD	ALL	783848.52	69974.45	3115637244	2208645234	367390.8	0.034	090-119	090-119
11/3/17	SalesMaterial	SalesMaterial	STANDARD	ALL	770048.37	69311.97	3125766832	2195447440	354054.6	0.034	090-119	090-119
[illegible]												

Der nächste Abschnitt beschreibt die im Bericht verwendeten Spalten.

Layout der exportierten Datei

In der folgenden Tabelle wird das Layout der exportierten beschrieben.

REST APIs für Amazon S3-Analyse

Die folgenden REST-Operationen werden für Lagerbestandslisten verwendet.

- DELETE Bucket analytics configuration
- GET Bucket analytics configuration
- List Bucket Analytics Configuration
- PUT Bucket analytics configuration

Amazon S3-Sicherheit

Cloud-Sicherheit hat bei AWS höchste Priorität. Als AWS-Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die eingerichtet wurde, um die Anforderungen der anspruchsvollsten Organisationen in puncto Sicherheit zu erfüllen.

Sicherheit ist eine übergreifende Verantwortlichkeit zwischen AWS und Ihnen. Im Modell der übergreifenden Verantwortlichkeit wird Folgendes mit „Sicherheit der Cloud“ bzw. „Sicherheit in der Cloud“ umschrieben:

- Sicherheit der Cloud – AWS ist für den Schutz der Infrastruktur verantwortlich, in der AWS-Services in der AWS Cloud ausgeführt werden. AWS bietet Ihnen außerdem Services, die Sie auf sichere Weise verwenden können. Die Wirksamkeit unserer Sicherheitsfunktionen wird regelmäßig von externen Prüfern im Rahmen des AWS-Compliance-Programms getestet und überprüft. Weitere Informationen zu den Compliance-Programmen für Amazon S3 finden Sie unter Im Rahmen des Compliance-Programms zugelassene AWS-Services.
- Sicherheit in der Cloud in – Ihr Verantwortungsumfang wird durch den AWS-Service bestimmt, den Sie verwenden. In Ihre Verantwortung fallen außerdem weitere Faktoren, wie z. B. die Vertraulichkeit der Daten, die Anforderungen Ihrer Organisation sowie geltende Gesetze und Vorschriften.

Diese Dokumentation beschreibt, wie Sie das Modell der übergreifenden Verantwortlichkeit bei der Verwendung von Amazon S3 anwenden können. Die folgenden Themen veranschaulichen, wie Sie Amazon S3 zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfigurieren können. Sie erfahren außerdem, wie Sie andere AWS-Services verwenden können, die Ihnen beim Überwachen und Sichern Ihrer Amazon S3-Ressourcen helfen.

Themen

Datenschutz in Amazon S3

Amazon S3 bietet eine hoch robuste Dateispeicherinfrastruktur für aufgabenkritische und primäre Datenspeicherungen. Objekte werden redundant auf mehreren Geräten in verschiedenen Anlagen einer Amazon S3-Region gespeichert. Zur Unterstützung der Datenhaltbarkeit werden Ihre Daten mit den Amazon S3 `PUT`- und `PUT Object copy`-Operationen synchron über mehrere Anlagen gespeichert. Nach der Speicherung wird die Haltbarkeit von Amazon S3 durch schnelle Erkennung und Behebung etwaiger Redundanzverluste bewahrt.

Der Amazon S3-Standardspeicher bietet folgende Funktionen:

- Über das Amazon S3 Service Level Agreement abgesichert
- Auf 99,999999999 %-ige Zuverlässigkeit und 99,99 ige Verfügbarkeit von Objekten über einen Zeitraum von einem Jahr ausgelegt.
- Darauf ausgelegt, dass auch beim gleichzeitigen Ausfall von zwei getrennten Speichereinrichtungen kein Datenverlust auftritt

Darüber hinaus schützt Amazon S3 Ihre Daten durch Versioning. Sie können Versioning verwenden, um sämtliche Versionen aller Objekte in Ihrem Amazon S3-Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen. Standardmäßig wird durch Anforderungen die zuletzt geschriebene Version abgerufen. Ältere Versionen eines Objekts können abgerufen werden, indem die entsprechende Version des Objekts in der Anforderung angegeben wird.

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls den Adressdatenschutz in Amazon S3:

- Serverseitige Implementierung
- Erzwingen der Verschlüsselung von Daten bei der Übertragung
- Erwägen Sie die Verwendung von Amazon Macie mit Amazon S3
- Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets
- Überwachen von AWS-Sicherheitshinweisen

Richtlinie für den Datenverkehr zwischen Netzwerken

In diesem Thema wird beschrieben, wie Amazon S3 Verbindungen vom Service zu anderen Speicherorten sichert.

Datenverkehr zwischen Service und lokalen Clients und Anwendungen

Sie haben zwei Verbindungsoptionen zwischen Ihrem privaten Netzwerk und AWS.

- AWS Site-to-Site VPN-Verbindung Weitere Informationen finden Sie unter Was istAWS Site-to-Site-VPN?
- AWS Direct Connect-Verbindung Weitere Informationen finden Sie unter Was istAWS Direct Connect?

Der Zugriff auf Amazon S3 über das Netzwerk erfolgt durch veröffentlichte AWS-APIS. Clients müssen Transport Layer Security (TLS) 1.0 unterstützen. Wir empfehlen TLS 1.2 oder höher. Clients müssen außerdem Cipher Suites mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi. Außerdem müssen Sie die Anfragen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signieren, die einem IAM-Prinzipal zugeordnet sind. Sie können auch den AWS Security Token Service (STS) verwenden, um temporäre Sicherheitsanmeldeinformationen zu generieren.

Datenverkehr zwischen AWS-Ressourcen in derselben Region

Ein Amazon Virtual Private Cloud (Amazon VPC)-Endpunkt für Amazon S3 ist eine logische Entity innerhalb einer VPC, die ausschließlich Verbindungen mit Amazon S3 zulässt. Der Amazon VPC leitet Anfragen an Amazon S3 weiter und die Antworten zurück an die VPC. Weitere Informationen finden Sie unter VPC-Endpunkte im Amazon VPC Benutzerhandbuch. Dieser Abschnitt enthält Beispiele für Bucket-Richtlinien, die für die Steuerung des Zugriffs auf S3-Buckets von VPC-Endpunkten aus verwendet werden können. Siehe Beispiel für Bucket-Richtlinien für VPC-Endpunkte für Amazon S3 (p. 454).

Datenschutz durch Verschlüsselung

Dieser Datenschutz bezieht sich auf Daten bei der Übertragung (wenn sie zu Amazon S3 oder von diesem geschickt werden) sowie auf ruhende Daten (die in Amazon S3-Rechenzentren auf Datenträgern gespeichert sind). Sie können Daten mithilfe von Secure Sockets Layer (SSL) oder einer clientseitigen Verschlüsselung während der Übertragung schützen. Es gibt die folgenden Optionen, Daten im Ruhezustand in Amazon S3 zu schützen:

- Verwendung der serverseitigen Verschlüsselung – Sie fordern Amazon S3 auf, Ihr Objekt zu verschlüsseln, bevor es dieses Objekt auf Datenträger in seinen Rechenzentren schreibt, und es wieder zu entschlüsseln, wenn Sie die Objekte herunterladen.
- Verwendung der clientseitigen Verschlüsselung – Sie können Daten clientseitig verschlüsseln und die verschlüsselten Daten auf Amazon S3 hochladen. In diesem Fall verwalten Sie den Verschlüsselungsprozess, die Verschlüsselungsschlüssel und die zugehörigen Tools.

Weitere Informationen zur serverseitigen Verschlüsselung und zur clientseitigen Verschlüsselung finden Sie in den unten aufgeführten Themen.

Themen

Schützen von Daten mithilfe serverseitiger Verschlüsselung

Die serverseitige Verschlüsselung ist die Verschlüsselung von Daten an seinem Ziel durch die Anwendung oder den Service, der sie empfängt. Amazon S3 verschlüsselt Ihre Daten auf Objektebene, indem es sie auf Festplatten in seinen Rechenzentren schreibt und für Sie entschlüsselt, wenn Sie darauf zugreifen. Wenn Sie Ihre Anforderung authentifizieren und Zugriffsberechtigungen besitzen, gibt es in Bezug auf die Art und Weise, wie Sie auf verschlüsselte oder nicht verschlüsselte Objekte zugreifen, keinen Unterschied. Wenn Sie beispielsweise Ihre Objekte unter Verwendung einer vorsignierten URL teilen, verhält sich die URL für verschlüsselte und unverschlüsselte Objekte gleich. Wenn Sie die Objekte in Ihrem Bucket auflisten, gibt die Listen-API außerdem eine Liste aller Objekte zurück, unabhängig davon, ob sie verschlüsselt sind.

> Note
>
> Sie können nicht gleichzeitig unterschiedliche Arten serverseitiger Verschlüsselung auf dasselbe Objekt anwenden.

Es bieten sich Ihnen drei sich gegenseitig ausschließende Optionen, abhängig davon, wie Sie die Zugriffsschlüssel verwalten möchten.

Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)

Wenn Sie eine serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) verwenden, wird jedes Objekt wird mit einem eindeutigen Schlüssel verschlüsselt. Als zusätzlicher Schutz wird der eigentliche Schlüssel mit einem Master-Schlüssel verschlüsselt, der regelmäßig rotiert wird. Die serverseitige Amazon S3-Verschlüsselung verwendet eine der stärksten Blockverschlüsselungen, die verfügbar sind, 256-bit Advanced Encryption Standard (AES-256), um Ihre Daten zu verschlüsseln. Weitere Informationen finden Sie unter Daten schützen durch serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3) (p. 298).

Serverseitige Verschlüsselung mit Kundenmasterschlüsseln (CMKs), die in AWS Key Management Service (SSE-KMS) gespeichert sind

Serverseitige Verschlüsselung mit Kundenmasterschlüsseln (CMKs), die in AWS Key Management Service (SSE-KMS) gespeichert sind ist mit SSE-S3 vergleichbar, jedoch mit einigen zusätzlichen Vorteilen und Kosten für die Nutzung dieses Services. Es gibt separate Berechtigungen für die Verwendung eines CMK, das einen zusätzlichen Schutz vor unbefugtem Zugriff auf Ihre Objekte in Amazon S3 bietet. SSE-KMS bietet Ihnen außerdem ein Prüfprotokoll, das anzeigt, wann und von wem Ihr CMK verwendet wurde. Darüber hinaus können Sie kundenseitig verwaltete CMKs erstellen und verwalten oder von AWS verwaltete CMKs verwenden, die für Sie, Ihren Service und Ihre Region einzigartig sind. Weitere Informationen finden Sie unter Schutz von Daten durch serverseitige Verschlüsselung mit CMKs, die in AWS Key Management Service (SSE-KMS) gespeichert sind (p. 291).

Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

Bei serverseitiger Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) verwalten Sie die Verschlüsselungsschlüssel, und Amazon S3 verwaltet die Verschlüsselung, wenn es auf Datenträger schreibt, und die Entschlüsselung, wenn Sie auf Ihre Objekte zugreifen. Weitere Informationen finden Sie unter Schützen von Daten durch eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) (p. 307).

Schutz von Daten durch serverseitige Verschlüsselung mit CMKs, die in AWS Key Management Service (SSE-KMS) gespeichert sind

Die serverseitige Verschlüsselung ist die Verschlüsselung von Daten am Ziel durch die Anwendung oder den Service, die oder der sie empfängt. AWS Key Management Service (AWS KMS) ist ein Service, der sichere, hoch verfügbare Hard- und Software kombiniert, um ein für die Cloud skaliertes Schlüsselmanagementsystem bereitzustellen. Amazon S3 verwendet AWS KMS-.Kundenhauptschlüssel (Customer Master Keys, CMKs), um Ihre Amazon S3-Objekte zu verschlüsseln. AWS KMS verschlüsselt ausschließlich die Objektdaten. Objektmetadaten werden grundsätzlich nicht verschlüsselt.

Bei der Verwendung von CMKs können Sie mit AWS KMS über die AWS Management Console oder die AWS KMS APIs zentral CMKs erstellen, Richtlinien zur Kontrolle der Verwendungsmöglichkeiten von CMKs definieren und die Verwendung von CMKs prüfen, um nachzuweisen, dass sie korrekt verwendet werden. Sie können mit diesen CMKs Ihre Daten in Amazon S3-Buckets schützen. Wenn Sie die SSE-KMS-Verschlüsselung mit einem S3-Bucket verwenden, muss sich der AWS KMS-CMK in derselben Region wie der Bucket befinden.

Für die Verwendung von AWS KMS-CMKs fallen zusätzliche Kosten an. Weitere Informationen finden Sie unter AWS Key Management Service-Konzepte – Kundenhauptschlüssel (CMKs) und Preise für den AWS Key Management Service im AWS Key Management Service Developer Guide.

> **Note**
>
> Sie benötigen die Berechtigung `kms:Decrypt`, wenn Sie ein mit einem AWS KMS-CMK verschlüsseltes Amazon S3-Objekt hoch- oder herunterladen. Sie benötigen diese Berechtigung zusätzlich zu den Berechtigungen `kms:ReEncrypt`, `kms:GenerateDataKey` und `kms:DescribeKey`. Weitere Informationen finden Sie unter Fehler beim Hochladen einer großen Datei zu Amazon S3 mit Verschlüsselung über einen AWS KMS-CMK.

AWS-verwaltete CMKs und kundenverwaltete CMKs

Wenn Sie serverseitige Verschlüsselung mit AWS KMS (SSE-KMS) verwenden, können Sie den standardmäßigen AWS-verwalteten CMK verwenden oder einen kundenverwalteten CMK angeben, den Sie bereits erstellt haben.

Wenn Sie keinen kundenverwalteten CMK angeben, erstellt Amazon S3 automatisch einen AWS-verwalteten CMK in Ihrem AWS-Konto, wenn Sie ein mit SSE-KMS verschlüsseltes Objekt zum ersten Mal zu einem Bucket hinzufügen. Standardmäßig verwendet Amazon S3 dieses CMK für SSE-KMS.

Wenn Sie ein vom Kunden verwaltetes CMK für SSE-KMS verwenden möchten, können Sie das CMK erstellen, bevor Sie SSE-KMS konfigurieren. Wenn Sie dann SSE-KMS für Ihren Bucket konfigurieren, können Sie den vorhandenen vom Kunden verwalteten CMK angeben.

Durch die Erstellung eines eigenen kundenverwalteten CMK erhalten Sie mehr Flexibilität und Kontrolle über das CMK. Beispielsweise können Sie kundenverwaltete CMKs erstellen, drehen und deaktivieren. Sie können auch Zugriffskontrollen definieren und die vom Kunden verwalteten CMKs prüfen, mit denen Sie Ihre Daten schützen. Weitere Informationen über vom Kunden verwaltete und von AWS verwaltete CMKs finden Sie unter AWS KMS-Konzepte im AWS Key Management Service Developer Guide.

> **Important**
>
> Wenn Sie in Amazon S3 einen AWS KMS-CMK für Server-seitige Verschlüsselung verwenden, müssen Sie einen symmetrischen CMK auswählen. Amazon S3 unterstützt ausschließlich symmetrische CMKs, asymmetrische CMKs dagegen nicht. Weitere Informationen finden Sie unter Verwenden von symmetrischen und asymmetrischen Schlüsseln im AWS Key Management Service-Entwicklerhandbuch.

AWS-Signaturversion 4

Wenn Sie Objekte hochladen oder auf Objekte zugreifen, die mit SSE-KMS verschlüsselt sind, müssen Sie AWS-Signaturversion 4 für zusätzliche Sicherheit verwenden. Weitere Informationen darüber, wie Sie dies mit einem AWS-SDK tun können, finden Sie unter Signaturversion in der Anforderungsauthentifizierung festlegen.

> **Important**
>
> Alle GET- und PUT-Anforderungen für ein durch AWS KMS geschütztes Objekt schlagen fehl, wenn Sie nicht über SSL oder unter Verwendung von SigV4 erfolgen.

SSE-KMS-Highlights

Die Highlights von SSE-KMS sind wie folgt:

- Sie können einen vom Kunden verwalteten CMK wählen, den Sie erstellen und verwalten, oder Sie können einen von AWS verwalteten CMK auswählen, den Amazon S3 in Ihrem AWS-Konto erstellt und für Sie verwaltet. Wie ein vom Kunden verwalteter CMK ist Ihr AWS-verwalteter CMK einzigartig für Ihr AWS-Konto und Ihre Region. Nur Amazon S3 hat die Berechtigung, diesen CMK in Ihrem Namen zu verwenden. Amazon S3 unterstützt nur symmetrische CMKs.
- Über die AWS KMS-Konsole können Sie prüffähige vom Kunden verwaltete CMKs erstellen, rotieren und deaktivieren.
- Das ETag in der Antwort ist nicht das MD5 der Objektdaten.
- Die für die Verschlüsselung Ihrer Daten verwendeten Datenschlüssel sind ebenfalls verschlüsselt und werden zusammen mit den Daten gespeichert, die sie schützen.
- Die Sicherheitskontrollen in AWS KMS tragen dazu bei, dass Sie Ihre Compliance-Anforderungen bezüglich der Verschlüsselung erfüllen.

Erzwingen serverseitiger Verschlüsselung

Um die serverseitige Verschlüsselung aller Objekte in einem bestimmten Amazon S3-Bucket anzufordern, können Sie eine Bucket-Richtlinie verwenden. Beispielsweise verweigert die folgende Bucket-Richtlinie jedem die `s3:PutObject`-Berechtigung zum Hochladen von Objekten, wenn die Anforderung nicht den `x-amz-server-side-encryption`-Header enthält, der eine serverseitige Verschlüsselung mit SSE-KMS anfordert.

```
{
   "Version":"2012-10-17",
   "Id":"PutObjPolicy",
   "Statement":[{
         "Sid":"DenyUnEncryptedObjectUploads",
         "Effect":"Deny",
         "Principal":"*",
```

```
            "Action":"s3:PutObject",
            "Resource":"arn:aws:s3:::YourBucket/*",
            "Condition":{
               "StringNotEquals":{
                  "s3:x-amz-server-side-encryption":"aws:kms"
               }
            }
         }
      ]
}
```

Um festzulegen, dass ein bestimmter AWS KMS-CMK verwendet werden muss, um die Objekte in einem Bucket zu verschlüsseln, können Sie den `s3:x-amz-server-side-encryption-aws-kms-key-id`-Bedingungsschlüssel verwenden. Um den AWS KMS-CMK festzulegen, müssen Sie einen Amazon-Ressourcennamen (ARN) verwenden, der im Format `arn:aws:kms:`*`region`*`:`*`acct-id`*`:key/`*`key-id`*"
vorliegt.

> **Note**
>
> Wenn Sie ein Objekt hochladen, können Sie das AWS KMS-CMK über den `x-amz-server-side-encryption-aws-kms-key-id`-Header festlegen. Wenn der Header nicht in der Anforderung vorhanden ist, geht Amazon S3 von einem von AWS verwalteten CMK aus. Unabhängig davon muss die AWS KMS-Schlüssel-ID, die Amazon S3 für die Objektverschlüsselung verwendet, mit der AWS KMS-Schlüssel-ID in der Richtlinie übereinstimmen. Andernfalls verweigert Amazon S3 die Anforderung.

Die vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel und weitere Informationen zur Angabe von Bedingungsschlüsseln finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382).

Verwenden von AWS Key Management Service in der Amazon S3-Konsole

Weitere Informationen zur Verwendung der Amazon S3-Konsole mit in AWS KMS gespeicherten CMKs finden Sie unter Wie füge ich eine Verschlüsselung zu einem S3-Objekt hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

API-Unterstützung für AWS Key Management Service in Amazon S3

Um SSE-KMS in den REST-APIs der Objekterstellung anzufordern, verwenden Sie den `x-amz-server-side-encryption`-Header. Um die ID des AWS KMS-CMKs anzugeben, der für das Objekt verwendet wurde, verwenden Sie `x-amz-server-side-encryption-aws-kms-key-id`. Die Amazon S3-API unterstützt außerdem Verschlüsselungskontext, mit dem Header `x-amz-server-side-encryption-context`. Weitere Informationen finden Sie unter Angeben von AWS Key Management Service in Amazon S3 unter Verwendung der REST-API (p. 296).

Die AWS-SDKs stellen außerdem Wrapper-APIs bereit, mit denen Sie SSE-KMS mit Amazon S3 anfordern können. Weitere Informationen finden Sie unter Festlegen des AWS Key Management Service in Amazon S3 mit den AWS-SDKs (p. 293).

Festlegen des AWS Key Management Service in Amazon S3 mit den AWS-SDKs

Wenn Sie AWS-SDKs verwenden, können Sie Amazon S3 auffordern, AWS Key Management Service (AWS KMS) Kundenstammschlüssel (CMKs) zu verwenden. Dieser Abschnitt enthält Beispiele für die Verwendung des AWS-SDKs für Java und .NET. Informationen zu anderen SDKs finden Sie unter Beispiel-Code und Bibliotheken.

> **Important**
>
> Wenn Sie in Amazon S3 einen AWS KMS-CMK für Server-seitige Verschlüsselung verwenden, müssen Sie einen symmetrischen CMK auswählen. Amazon S3 unterstützt ausschließlich symmetrische CMKs, asymmetrische CMKs dagegen nicht. Weitere Informationen finden Sie

unter Verwenden von symmetrischen und asymmetrischen Schlüsseln im AWS Key Management Service-Entwicklerhandbuch.

AWS-SDK für Java

Dieser Abschnitt erklärt verschiedene Amazon S3-Operationen mit dem AWS SDK for Java und wie Sie die AWS KMS-CMKs verwenden.

Put-Operation

Wenn Sie ein Objekt mit dem AWS-SDK für Java hochladen, können Sie Amazon S3 auffordern, einen AWS KMS-CMK zu verwenden, indem Sie die Eigenschaft `SSEAwsKeyManagementParams` hinzufügen, wie in der folgenden Anforderung gezeigt.

```
PutObjectRequest putRequest = new PutObjectRequest(bucketName,
   keyName, file).withSSEAwsKeyManagementParams(new SSEAwsKeyManagementParams());
```

In diesem Fall verwendet Amazon S3 das den von AWS verwalteten CMK (siehe Serverseitige Verschlüsselung mit in AWS KMS gespeicherten CMKs verwenden (p. 291)). Sie können optional einen symmetrischen kundenverwalteten CMK erstellen und diesen in der Anfrage angeben.

```
PutObjectRequest putRequest = new PutObjectRequest(bucketName,
   keyName, file).withSSEAwsKeyManagementParams(new SSEAwsKeyManagementParams(keyID));
```

Weitere Informationen zum Erstellen von kundenverwalteten CMKs finden Sie unter Programmieren der AWS KMS-API in der AWS Key Management Service Developer Guide.

Funktionierende Codebeispiele zum Hochladen eines Objekts finden Sie unter den folgenden Themen. Sie müssen diese Codebeispiele aktualisieren und Verschlüsselungsinformationen bereitstellen, wie im vorigen Codefragment gezeigt.

- Weitere Informationen zum Hochladen eines Objekts in einer einzigen Operation finden Sie unter Hochladen eines Objekts mit AWS SDK for Java (p. 194).
- Weitere Informationen zu mehrteiligen Uploads finden Sie unter den folgenden Themen:
  - Weitere Informationen über die Verwendung der High-Level-API für mehrteilige Uploads finden Sie unter Hochladen einer Datei (p. 206).
  - Weitere Informationen über die Verwendung der Low-Level-API für mehrteilige Uploads finden Sie unter Hochladen einer Datei (p. 209).

Kopier-Operation

Beim Kopieren von Objekten fügen Sie die gleichen Anfrageeigenschaften (`ServerSideEncryptionMethod` und `ServerSideEncryptionKeyManagementServiceKeyId`) hinzu, um Amazon S3 anzuweisen, einen AWS KMS-CMK zu verwenden. Weitere Informationen über das Kopieren von Objekten finden Sie unter Objekte kopieren (p. 233).

Vorsignierte URLs

Beim Erstellen einer vorsignierten URL für ein mit einer AWS KMS-CMK verschlüsseltes Objekt müssen Sie die Signaturversion 4 explizit angeben.

```
ClientConfiguration clientConfiguration = new ClientConfiguration();
clientConfiguration.setSignerOverride("AWSS3V4SignerType");
AmazonS3Client s3client = new AmazonS3Client(
        new ProfileCredentialsProvider(), clientConfiguration);
...
```

Ein Codebeispiel finden Sie unter Generieren einer vorsignierten Objekt-URL mit AWS SDK for Java (p. 191).

AWS-SDK für .NET

In diesem Abschnitt werden verschiedene Amazon S3-Operationen unter Verwendung des AWS-SDK für .NET und die Verwendung der AWS KMS-CMKs erläutert.

Put-Operation

Wenn Sie ein Objekt mit dem AWS-SDK für .NET hochladen, können Sie Amazon S3 anweisen, einen AWS KMS-CMK zu verwenden, indem Sie die `ServerSideEncryptionMethod`-Eigenschaft hinzufügen, wie in der folgenden Anforderung gezeigt.

```
PutObjectRequest putRequest = new PutObjectRequest
   {
       BucketName = bucketName,
       Key = keyName,
       // other properties.
       ServerSideEncryptionMethod = ServerSideEncryptionMethod.AWSKMS
   };
```

In diesem Fall verwendet Amazon S3 den von AWS verwalteten CMK. Weitere Informationen finden Sie unter Schutz von Daten durch serverseitige Verschlüsselung mit CMKs, die in AWS Key Management Service (SSE-KMS) gespeichert sind (p. 291). Sie können optional Ihren eigenen kundenverwalteten CMK erstellen und diesen in der Anfrage angeben.

```
PutObjectRequest putRequest1 = new PutObjectRequest
{
    BucketName = bucketName,
    Key = keyName,
    // other properties.
    ServerSideEncryptionMethod = ServerSideEncryptionMethod.AWSKMS,
    ServerSideEncryptionKeyManagementServiceKeyId = keyId
};
```

Weitere Informationen zum Erstellen von kundenverwalteten CMKs finden Sie unter Programmieren der AWS KMS-API in der AWS Key Management Service Developer Guide.

Funktionierende Codebeispiele zum Hochladen eines Objekts finden Sie unter den folgenden Themen. Sie müssen diese Codebeispiele aktualisieren und Verschlüsselungsinformationen bereitstellen, wie im vorigen Codefragment gezeigt.

- Weitere Informationen zum Hochladen eines Objekts in einer einzigen Operation finden Sie unter Hochladen eines Objekts mit AWS SDK für .NET (p. 195).
- Weitere Informationen zu mehrteiligen Uploads finden Sie unter den folgenden Themen:
  - Weitere Informationen über die Verwendung der High-Level-API für mehrteilige Uploads finden Sie unter Hochladen einer Datei in einen S3-Bucket unter Verwendung des AWS SDK für .NET (High-Level-API) (p. 213)
  - Weitere Informationen über die Verwendung der Low-Level-API für mehrteilige Uploads finden Sie unter Hochladen einer Datei in einen S3-Bucket unter Verwendung von AWS SDK für .NET (Low-Level-API) (p. 220)

Kopier-Operation

Beim Kopieren von Objekten fügen Sie die gleichen Anfrageeigenschaften (`ServerSideEncryptionMethod` und `ServerSideEncryptionKeyManagementServiceKeyId`)

hinzu, um Amazon S3 anzuweisen, einen AWS KMS-CMK zu verwenden. Weitere Informationen über das Kopieren von Objekten finden Sie unter Objekte kopieren (p. 233).

Vorsignierte URLs

Beim Erstellen einer vorsignierten URL für ein mit einer AWS KMS-CMK verschlüsseltes Objekt müssen Sie die Signaturversion 4 explizit angeben.

```
AWSConfigs.S3Config.UseSignatureVersion4 = true;
```

Ein Codebeispiel finden Sie unter Generieren einer vorsignierten Objekt-URL mit AWS SDK für .NET (p. 192).

Angeben von AWS Key Management Service in Amazon S3 unter Verwendung der REST-API

Wenn Sie ein Objekt erstellen, d. h., wenn Sie ein neues Objekt hochladen oder ein vorhandenes Objekt kopieren, können Sie für die Verschlüsselung Ihrer Daten die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS)-Kundenmasterschlüsseln (CMKs) angeben. Fügen Sie hierzu der Anforderung den Header `x-amz-server-side-encryption` hinzu. Legen Sie den Wert des Headers auf den Verschlüsselungsalgorithmus `aws:kms` fest. Amazon S3 bestätigt durch Rückgabe des Antwort-Headers `x-amz-server-side-encryption`, dass Ihr Objekt unter Verwendung von SSE-KMS gespeichert wurde.

Wenn Sie den Header `x-amz-server-side-encryption` mit dem Wert `aws:kms` angeben, können Sie auch die folgenden Anforderungs-Header verwenden:

- `x-amz-server-side-encryption-aws-kms-key-id`
- `x-amz-server-side-encryption-context`

Amazon S3-REST-APIs, die SSE-KMS unterstützen

Die folgenden REST-APIs akzeptieren die Anforderungs-Header `x-amz-server-side-encryption`, `x-amz-server-side-encryption-aws-kms-key-id` und `x-amz-server-side-encryption-context`.

- PUT Object: Wenn Sie Daten über die PUT API hochladen, können Sie diese Anforderungs-Header angeben.
- PUT Object – Copy: Wenn Sie ein Objekt kopieren, erhalten Sie ein Quell- und ein Zielobjekt. Wenn Sie SSE-KMS-Header mit der Operation COPY übergeben, werden sie nur auf das Zielobjekt angewendet. Beim Kopieren eines vorhandenen Objekts wird das Zielobjekt unabhängig davon, ob das Quellobjekt verschlüsselt ist, nicht verschlüsselt, es sei denn, Sie fordern die serverseitige Verschlüsselung explizit an.
- Operation POST: Wenn Sie eine POST-Operation für das Hochladen eines Objekts verwenden, geben Sie die Informationen in die Formularfelder und nicht in die Anforderungs-Header ein.
- Initiieren eines mehrteiligen Uploads: Wenn Sie große Objekte über die API für mehrteilige Uploads hochladen, können Sie diese Header angeben. Sie geben diese Header in der Initiierungsanforderung an.

Die Antwort-Header der folgenden REST-APIs geben den Header `x-amz-server-side-encryption` zurück, wenn ein Objekt unter Verwendung der serverseitigen Verschlüsselung gespeichert wird.

- PUT Object
- PUT Object – Kopieren
- POST Object

- Initiieren eines mehrteiligen Uploads
- Upload Part
- Hochladen eines Teiluploads – Kopieren
- Abschließen eines mehrteiligen Uploads
- Get Object
- Head Object

Important

- Alle GET- und PUT-Anforderungen für ein Objekt, das durch AWS KMS geschützt wird, schlagen fehl, wenn Sie diese nicht mit Secure Sockets Language (SSL) oder Signature Version 4 erstellen.
- Wenn Ihr Objekt SSE-KMS verwendet, dürfen Sie keine Verschlüsselungsanforderungs-Header für `GET`- und `HEAD`-Anforderungen senden. Andernfalls erhalten Sie den Fehler HTTP 400 BadRequest.

Verschlüsselungskontext (`x-amz-server-side-encryption-context`)

Wenn Sie `x-amz-server-side-encryption:aws:kms` angeben, unterstützt die Amazon S3-API einen Verschlüsselungskontext mit dem Header `x-amz-server-side-encryption-context`. Ein Verschlüsselungskontext ist ein optionaler Satz von Schlüssel-Wert-Paaren, die zusätzliche kontextbezogene Informationen zu den Daten enthalten können. Weitere Informationen zum Verschlüsselungskontext finden Sie unter AWS Key Management Service-Konzepte – Verschlüsselungskontext im AWS Key Management Service Developer Guide.

Der Verschlüsselungskontext kann ein beliebiger Wert sein, vorausgesetzt, der Header entspricht dem mit Base64-codierten JSON-Format. Da der Verschlüsselungskontext jedoch nicht verschlüsselt ist und protokolliert wird, wenn die AWS CloudTrail-Protokollierung aktiviert ist, sollte Ihr Verschlüsselungskontext keine sensiblen Informationen enthalten. Ihr Kontext sollte darüber hinaus die Daten beschreiben, die verschlüsselt oder entschlüsselt werden, damit Sie die von AWS KMS generierten CloudTrail-Ereignisse besser verstehen.

Daher fügt Amazon S3 möglicherweise einen vordefinierten Schlüssel `aws:s3:arn` mit einem Wert an, der gleich dem Amazon-Ressourcennamen (ARN) des Objekts für den von Ihnen bereitgestellten Verschlüsselungskontext ist. Dies geschieht nur, wenn sich der Schlüssel `aws:s3:arn` noch nicht im von Ihnen bereitgestellten Verschlüsselungskontext befindet. In diesem Fall wird dieser vordefinierte Schlüssel angefügt, wenn Amazon S3 Ihre Put-Anforderungen verarbeitet. Wenn sich dieser `aws:s3:arn`-Schlüssel bereits in Ihrem Verschlüsselungskontext befindet, wird der Schlüssel Ihrem Verschlüsselungskontext nicht noch einmal hinzugefügt.

Wenn Sie diesen vordefinierten Schlüssel als Teil Ihres Verschlüsselungskontexts verwenden, können Sie relevante Anforderungen in CloudTrail verfolgen. So können Sie stets sehen, welcher ARN des Amazon S3-Objekts mit welchem Verschlüsselungsschlüssel verwendet wurde. Wenn dieser vordefinierte Schlüssel Teil Ihres Verschlüsselungskontexts ist, wird darüber hinaus sichergestellt, dass der Verschlüsselungskontext zwischen verschiedenen Amazon S3-Objekten nicht identisch ist. Dies bedeutet eine zusätzliche Sicherheit für Ihre Objekte. Ihr vollständiger Verschlüsselungskontext wird überprüft, ob der Wert gleich dem Objekt-ARN ist.

AWS KMS-Schlüssel-ID (`x-amz-server-side-encryption-aws-kms-key-id`)

Sie können den Header `x-amz-server-side-encryption-aws-kms-key-id` verwenden, um die ID des vom Kunden verwalteten CMK anzugeben, der zum Schutz der Daten verwendet wird. Wenn Sie `x-amz-server-side-encryption:aws:kms`, jedoch nicht `x-amz-server-side-encryption-aws-kms-key-id` angeben, verwendet Amazon S3 den von AWS verwalteten CMK in AWS KMS, um die Daten

zu schützen. Wenn Sie einen vom Kunden verwaltetes AWS KMS-CMK verwenden möchten, müssen Sie die `x-amz-server-side-encryption-aws-kms-key-id` des vom Kunden verwalteten CMK angeben.

Important

Wenn Sie in Amazon S3 einen AWS KMS-CMK für Server-seitige Verschlüsselung verwenden, müssen Sie einen symmetrischen CMK auswählen. Amazon S3 unterstützt ausschließlich symmetrische CMKs, asymmetrische CMKs dagegen nicht. Weitere Informationen finden Sie unter Verwenden von symmetrischen und asymmetrischen Schlüsseln im AWS Key Management Service-Entwicklerhandbuch.

Daten schützen durch serverseitige Verschlüsselung mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3)

Die serverseitige Verschlüsselung schützt Ruhedaten. Amazon S3 verschlüsselt jedes Objekt mit einem eindeutigen Schlüssel. Als zusätzlicher Schutz wird der Schlüssel selbst mit einem Masterschlüssel verschlüsselt, der regelmäßig rotiert wird. Die serverseitige Amazon S3-Verschlüsselung verwendet eine der stärksten Blockverschlüsselungen, die zur Verschlüsselung Ihrer Daten verfügbar sind, nämlich 256-bit Advanced Encryption Standard (AES-256).

Wenn Sie eine serverseitige Verschlüsselung für alle in einem Bucket gespeicherten Objekte benötigen, verwenden Sie eine Bucket-Richtlinie. Beispielsweise verweigert die folgende Bucket-Richtlinie Berechtigungen zum Hochladen von Objekten, wenn die Anforderung nicht den `x-amz-server-side-encryption`-Header enthält, der eine serverseitige Verschlüsselung anfordert:

```
{
  "Version": "2012-10-17",
  "Id": "PutObjPolicy",
  "Statement": [
    {
      "Sid": "DenyIncorrectEncryptionHeader",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::YourBucket/*",
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption": "AES256"
        }
      }
    },
    {
      "Sid": "DenyUnEncryptedObjectUploads",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::YourBucket/*",
      "Condition": {
        "Null": {
          "s3:x-amz-server-side-encryption": "true"
        }
      }
    }
  ]
}
```

Note

- Die serverseitige Verschlüsselung verschlüsselt nur die Objektdaten, nicht die Metadaten des Objekts.

API Support für die serverseitige Verschlüsselung

Um eine serverseitige Verschlüsselung mit den REST-APIs für die Objekterstellung anzufordern, stellen Sie den Anforderungs-Header `x-amz-server-side-encryption` bereit. Weitere Information zu den REST-APIs finden Sie unter Festlegen einer serverseitigen Verschlüsselung mit der REST API (p. 306).

Die folgenden Amazon S3-APIs unterstützen diesen Header:

- PUT-Operationen – Geben Sie den Anforderungs-Header an, wenn Sie Daten mithilfe der PUT-API hochladen. Weitere Informationen finden Sie unter PUT Object.
- Initiate Multipart Upload (Mehrteiligen Upload initiieren) – Geben Sie den Header in der Initiierungsanforderung an, wenn Sie große Objekte mit der API für mehrteilige Uploads hochladen. Weitere Informationen finden Sie unter Mehrteiligen Upload initiieren.
- COPY-Operationen – Beim Kopieren eines Objekts haben Sie ein Quell- und ein Zielobjekt. Weitere Informationen finden Sie unter PUT Object – Copy.

> **Note**
>
> Wenn Sie eine POST-Operation für das Hochladen eines Objekts verwenden, anstatt den Anforderungs-Header anzugeben, geben Sie dieselben Informationen in die Formularfelder ein. Weitere Informationen finden Sie unter POST Object.

Die AWS-SDKs bieten außerdem Wrapper-APIs, mit denen Sie eine serverseitige Verschlüsselung anfordern können. Sie können auch die AWS Management Console verwenden, um Objekte hochzuladen und eine serverseitige Verschlüsselung anzufordern.

Festlegen einer serverseitigen Verschlüsselung mit der AWS SDK for Java

Wenn Sie mit AWS SDK for Java ein Objekt hochladen, können Sie es mit serverseitiger Verschlüsselung verschlüsseln. Um eine serverseitige Verschlüsselung anzufordern, legen Sie mit der `ObjectMetadata`-Eigenschaft der `PutObjectRequest` den Anforderungs-Header `x-amz-server-side-encryption` fest. Wenn Sie die Methode `putObject()` des `AmazonS3Client` aufrufen, verschlüsselt und speichert Amazon S3 die Daten.

Sie können auch eine serverseitige Verschlüsselung anfordern, wenn Sie Objekte mit der API für mehrteilige Uploads hochladen:

- Wenn Sie die High-Level-API für mehrteilige Uploads verwenden, wenden Sie mit der Methode `TransferManager` serverseitige Verschlüsselung auf Objekte an, während Sie sie hochladen. Sie können eine beliebige der Upload-Methoden nutzen, die `ObjectMetadata` als Parameter entgegennehmen. Weitere Informationen finden Sie unter Verwendung des AWS Java SDK für mehrteilige Uploads (High-Level-API) (p. 205).
- Wenn Sie die Low-Level-API für mehrteilige Uploads verwenden, legen Sie eine serverseitige Verschlüsselung beim Initiieren des mehrteiligen Uploads fest. Sie fügen die Eigenschaft `ObjectMetadata` beim Aufruf der Methode `InitiateMultipartUploadRequest.setObjectMetadata()` hinzu. Weitere Informationen finden Sie unter Hochladen einer Datei (p. 209).

Sie können den Verschlüsselungsstatus eines Objekts (Verschlüsseln eines unverschlüsselten Objekts oder Entschlüsseln eines verschlüsselten Objekts) nicht direkt ändern. Um den Verschlüsselungsstatus eines Objekts zu ändern, erstellen Sie eine Kopie des Objekts, geben den gewünschten Verschlüsselungsstatus der Kopie an und löschen dann das ursprüngliche Objekt. Amazon S3 verschlüsselt das kopierte Objekt nur, wenn Sie die serverseitige Verschlüsselung ausdrücklich anfordern. Um die Verschlüsselung des kopierten Objekts über die Java-API anzufordern, geben Sie unter Verwendung der `ObjectMetadata`-Eigenschaft eine serverseitige Verschlüsselung in der `CopyObjectRequest` an.

Example Beispiel

Das folgende Beispiel veranschaulicht, wie Sie eine serverseitige Verschlüsselung unter Verwendung des AWS SDK for Java festlegen. Es veranschaulicht, wie Sie die folgenden Aufgaben ausführen:

- Hochladen eines neuen Objekts unter Verwendung einer serverseitigen Verschlüsselung
- Ändern des Verschlüsselungsstatus eines Objekts (in diesem Beispiel Verschlüsseln eines zuvor unverschlüsselten Objekts) durch Anfertigen einer Kopie des Objekts
- Überprüfen des Verschlüsselungsstatus des Objekts

Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Festlegen einer serverseitigen Verschlüsselung mit der REST API (p. 306). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.internal.SSEResultBase;
import com.amazonaws.services.s3.model.*;

import java.io.ByteArrayInputStream;

public class SpecifyServerSideEncryption {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyNameToEncrypt = "*** Key name for an object to upload and encrypt ***";
        String keyNameToCopyAndEncrypt = "*** Key name for an unencrypted object to be
 encrypted by copying ***";
        String copiedObjectKeyName = "*** Key name for the encrypted copy of the
 unencrypted object ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Upload an object and encrypt it with SSE.
            uploadObjectWithSSEEncryption(s3Client, bucketName, keyNameToEncrypt);

            // Upload a new unencrypted object, then change its encryption state
            // to encrypted by making a copy.
            changeSSEEncryptionStatusByCopying(s3Client,
                    bucketName,
                    keyNameToCopyAndEncrypt,
                    copiedObjectKeyName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
   private static void uploadObjectWithSSEEncryption(AmazonS3 s3Client, String bucketName,
String keyName) {
       String objectContent = "Test object encrypted with SSE";
       byte[] objectBytes = objectContent.getBytes();

       // Specify server-side encryption.
       ObjectMetadata objectMetadata = new ObjectMetadata();
       objectMetadata.setContentLength(objectBytes.length);
       objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
       PutObjectRequest putRequest = new PutObjectRequest(bucketName,
               keyName,
               new ByteArrayInputStream(objectBytes),
               objectMetadata);

       // Upload the object and check its encryption status.
       PutObjectResult putResult = s3Client.putObject(putRequest);
       System.out.println("Object \"" + keyName + "\" uploaded with SSE.");
       printEncryptionStatus(putResult);
   }

   private static void changeSSEEncryptionStatusByCopying(AmazonS3 s3Client,
                                                          String bucketName,
                                                          String sourceKey,
                                                          String destKey) {
       // Upload a new, unencrypted object.
       PutObjectResult putResult = s3Client.putObject(bucketName, sourceKey, "Object
example to encrypt by copying");
       System.out.println("Unencrypted object \"" + sourceKey + "\" uploaded.");
       printEncryptionStatus(putResult);

       // Make a copy of the object and use server-side encryption when storing the copy.
       CopyObjectRequest request = new CopyObjectRequest(bucketName,
               sourceKey,
               bucketName,
               destKey);
       ObjectMetadata objectMetadata = new ObjectMetadata();
       objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
       request.setNewObjectMetadata(objectMetadata);

       // Perform the copy operation and display the copy's encryption status.
       CopyObjectResult response = s3Client.copyObject(request);
       System.out.println("Object \"" + destKey + "\" uploaded with SSE.");
       printEncryptionStatus(response);

       // Delete the original, unencrypted object, leaving only the encrypted copy in
Amazon S3.
       s3Client.deleteObject(bucketName, sourceKey);
       System.out.println("Unencrypted object \"" + sourceKey + "\" deleted.");
   }

   private static void printEncryptionStatus(SSEResultBase response) {
       String encryptionStatus = response.getSSEAlgorithm();
       if (encryptionStatus == null) {
           encryptionStatus = "Not encrypted with SSE";
       }
       System.out.println("Object encryption status is: " + encryptionStatus);
   }
}
```

Angeben einer serverseitigen Verschlüsselung mit AWS SDK für .NET

Wenn Sie ein Objekt hochladen, können Sie Amazon S3 dazu anweisen, es zu verschlüsseln. Um den Verschlüsselungsstatus eines vorhandenen Objekts zu ändern, erstellen Sie eine Kopie des Objekts und

löschen dann das Quellobjekt. Beachten Sie, dass die Kopieroperation das Ziel nur verschlüsselt, wenn Sie auf dem Zielobjekt ausdrücklich eine serverseitige Verschlüsselung anfordern. Um eine serverseitige Verschlüsselung in der `CopyObjectRequest` festzulegen, fügen Sie Folgendes hinzu:

```
ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256
```

Ein funktionierendes Beispiel, das zeigt, wie ein Objekt kopiert wird, finden Sie unter Kopieren eines Amazon S3-Objekts in einer einzigen Operation unter Verwendung von AWS SDK für .NET (p. 235).

Das folgende Beispiel lädt ein Objekt hoch. In der Anforderung weist das Beispiel Amazon S3 dazu an, das Objekt zu verschlüsseln. Das Beispiel ruft dann Objekt-Metadaten ab und überprüft die verwendete Verschlüsselungsmethode. Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SpecifyServerSideEncryptionTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for object created ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            WritingAnObjectAsync().Wait();
        }

        static async Task WritingAnObjectAsync()
        {
            try
            {
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    ContentBody = "sample text",
                    ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256
                };

                var putResponse = await client.PutObjectAsync(putRequest);

                // Determine the encryption state of an object.
                GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };
                GetObjectMetadataResponse response = await
 client.GetObjectMetadataAsync(metadataRequest);
                ServerSideEncryptionMethod objectEncryption =
 response.ServerSideEncryptionMethod;

                Console.WriteLine("Encryption method used: {0}",
 objectEncryption.ToString());
```

```
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing an
 object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Festlegen einer serverseitigen Verschlüsselung mit dem AWS SDK für PHP

In diesem Thema wird die Verwendung von Klassen aus Version 3 des AWS SDK für PHP gezeigt, um Objekten, die Sie zu Amazon Simple Storage Service (Amazon S3) hochladen, eine serverseitige Verschlüsselung hinzuzufügen. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist.

Um ein Objekt auf Amazon S3 hochzuladen, verwenden Sie die Methode Aws\S3\S3Client::putObject(). Um Ihrer Upload-Anfrage den Anfrage-Header `x-amz-server-side-encryption` hinzuzufügen, geben Sie den Parameter `ServerSideEncryption` mit dem Wert `AES256` an, wie im folgenden Codebeispiel veranschaulicht. Weitere Informationen zur serverseitigen Verschlüsselungsanforderungen finden Sie unter Festlegen einer serverseitigen Verschlüsselung mit der REST API (p. 306).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

// $filepath should be an absolute path to a file on disk.
$filepath = '*** Your File Path ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Upload a file with server-side encryption.
$result = $s3->putObject([
    'Bucket'               => $bucket,
    'Key'                  => $keyname,
    'SourceFile'           => $filepath,
    'ServerSideEncryption' => 'AES256',
]);
```

Amazon S3 gibt als Antwort den Header `x-amz-server-side-encryption` mit dem Wert zurück, den der Verschlüsselungsalgorithmus für die Verschlüsselung der Daten Ihres Objekts verwendet hat.

Wenn Sie große Objekte mithilfe der API für mehrteilige Uploads hochladen, können Sie für die hochzuladenden Objekte folgendermaßen eine serverseitige Verschlüsselung angeben:

- Wenn Sie die Low-Level-API für mehrteilige Uploads verwenden, geben Sie die serverseitige Verschlüsselung beim Aufruf der Methode Aws\S3\S3Client::createMultipartUpload() an. Um Ihrer Upload-Anfrage den Anfrage-Header `x-amz-server-side-encryption` hinzuzufügen, geben Sie

für den Parameter `array` den `ServerSideEncryption`-Schlüssel mit dem Wert `AES256` an. Weitere Informationen zur Low-Level-API für mehrteilige Uploads finden Sie unter Verwendung des AWS PHP SDK für mehrteilige Uploads (Low-Level-API) (p. 224).

- Wenn Sie die High-Level-API für mehrteilige Uploads verwenden, geben Sie die serverseitige Verschlüsselung mittels des Parameters `ServerSideEncryption` der Methode CreateMultipartUploadan. Ein Beispiel für die Verwendung der Methode `setOption()` mit der High-Level-API für mehrteilige Uploads finden Sie unter Verwenden des AWS PHP SDK für mehrteilige Uploads (p. 223).

Bestimmung des verwendeten Verschlüsselungsalgorithmus

Um den Verschlüsselungsstatus eines vorhandenen Objekts zu bestimmen, rufen Sie die Objektmetadaten mit der Methode Aws\S3\S3Client::headObject() ab, wie im folgenden PHP-Codebeispiel veranschaulicht.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Check which server-side encryption algorithm is used.
$result = $s3->headObject([
    'Bucket' => $bucket,
    'Key'    => $keyname,
]);
echo $result['ServerSideEncryption'];
```

Ändern der serverseitigen Verschlüsselung eines vorhandenen Objekts (Kopieroperation)

Um den Verschlüsselungsstatus eines vorhandenen Objekts zu ändern, erstellen Sie mittels der Methode Aws\S3\S3Client::copyObject() eine Kopie des Objekts und löschen dann das Quellobjekt. Beachten Sie, dass `copyObject()` das Ziel standardmäßig nicht verschlüsselt, es sei denn, Sie fordern explizit eine serverseitige Verschlüsselung des Zielobjekts an, indem Sie den Parameter `ServerSideEncryption` mit dem Wert `AES256` angeben. Im folgenden PHP-Codebeispiel wird eine Kopie eines Objekts erstellt und dem kopierten Objekt eine serverseitige Verschlüsselung hinzugefügt.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$sourceBucket = '*** Your Source Bucket Name ***';
$sourceKeyname = '*** Your Source Object Key ***';

$targetBucket = '*** Your Target Bucket Name ***';
$targetKeyname = '*** Your Target Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Copy an object and add server-side encryption.
$s3->copyObject([
    'Bucket'               => $targetBucket,
```

```
    'Key'                  => $targetKeyname,
    'CopySource'           => "{$sourceBucket}/{$sourceKeyname}",
    'ServerSideEncryption' => 'AES256',
]);
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Festlegen einer serverseitigen Verschlüsselung mit der AWS SDK für Ruby

Beim Hochladen eines Objekts mit AWS SDK für Ruby können Sie festlegen, dass das Objekt im Ruhezustand mit serverseitigen Verschlüsselung (SSE) verschlüsselt gespeichert werden soll. Wenn Sie das Objekt zurücklesen, wird es automatisch entschlüsselt.

Das folgende Beispiel mit AWS SDK für Ruby Version 3 zeigt, wie Sie das Verschlüsseln einer zu Amazon S3 hochgeladenen Datei im Ruhezustand angeben.

```
# The following example demonstrates how to specify that a file uploaded to Amazon S3 be
 encrypted at rest.
require 'aws-sdk-s3'

regionName = 'us-west-2'
bucketName = 'my-bucket'
key = 'key'
filePath = 'local/path/to/file'
encryptionType = 'AES256'

s3 = Aws::S3::Resource.new(region:regionName)
obj = s3.bucket(bucketName).object(key)
obj.upload_file(filePath, :server_side_encryption => encryptionType)
```

Ein Beispiel, das zeigt, wie ein Objekt ohne SSE hochgeladen wird, finden Sie unter Hochladen eines Objekts mit AWS SDK für Ruby (p. 197).

Bestimmung des verwendeten Verschlüsselungsalgorithmus

Das folgende Beispiel zeigt, wie Sie den Verschlüsselungsstatus eines vorhandenen Objekts bestimmen.

```
# Determine server-side encryption of an object.
require 'aws-sdk-s3'

regionName = 'us-west-2'
bucketName='bucket-name'
key = 'key'

s3 = Aws::S3::Resource.new(region:regionName)
enc = s3.bucket(bucketName).object(key).server_side_encryption
enc_state = (enc != nil) ? enc : "not set"
puts "Encryption state is #{enc_state}."
```

Wenn die serverseitige Verschlüsselung für das in Amazon S3 gespeicherte Objekt nicht verwendet wird, gibt die Methode weiterhin null zurück.

Ändern der serverseitigen Verschlüsselung eines vorhandenen Objekts (Kopieroperation)

Um den Verschlüsselungsstatus eines vorhandenen Objekts zu ändern, erstellen Sie eine Kopie des Objekts und löschen dann das Quellobjekt. Beachten Sie, dass die Methoden zum Kopieren

das Ziel standardmäßig nicht verschlüsseln, es sei denn, Sie fordern explizit eine serverseitige Verschlüsselung an. Sie können die Verschlüsselung des Zielobjekts anfordern, indem Sie den `server_side_encryption`-Wert im Hash-Argument der Option angeben, wie im folgenden Ruby-Codebeispiel gezeigt. Das Code-Beispiel zeigt, wie ein Objekt kopiert und die Kopie verschlüsselt werden.

```
require 'aws-sdk-s3'

regionName = 'us-west-2'
encryptionType = 'AES256'

s3 = Aws::S3::Resource.new(region:regionName)
bucket1 = s3.bucket('source-bucket-name')
bucket2 = s3.bucket('target-bucket-name')
obj1 = bucket1.object('Bucket1Key')
obj2 = bucket2.object('Bucket2Key')

obj1.copy_to(obj2, :server_side_encryption => encryptionType)
```

Ein funktionierendes Beispiel, das zeigt, wie ein Objekt ohne Verschlüsselung kopiert wird, finden Sie unter Kopieren eines Objekts mit AWS SDK für Ruby (p. 238).

Festlegen einer serverseitigen Verschlüsselung mit der REST API

Wenn das Objekt erstellt wird, d. h. wenn Sie ein neues Objekt hochladen oder eine Kopie eines vorhandenen Objekts erstellen, können Sie angeben, ob Amazon S3 Ihre Daten verschlüsseln soll, indem Sie der Anforderung den Header `x-amz-server-side-encryption` hinzufügen. Legen Sie den Wert des Headers auf den Verschlüsselungsalgorithmus `AES256` fest, den Amazon S3 unterstützt. Amazon S3 bestätigt, dass Ihr Objekt unter Verwendung der serverseitigen Verschlüsselung gespeichert ist, indem der Antwort-Header `x-amz-server-side-encryption` zurückgegeben wird.

Die folgenden REST Upload-APIs akzeptieren den Anforderungs-Header `x-amz-server-side-encryption`.

- PUT Object
- PUT Object – Kopieren
- POST Object
- Initiieren eines mehrteiligen Uploads

Wenn Sie große Objekte mit der API für mehrteiligen Upload hochladen, können Sie die serverseitige Verschlüsslung vorgeben, indem Sie der Anforderung zum Initiieren eines mehrteiligen Uploads den `x-amz-server-side-encryption`-Header hinzufügen. Beim Kopieren eines vorhandenen Objekts wird das Zielobjekt unabhängig davon, ob das Quellobjekt verschlüsselt ist, nicht verschlüsselt, es sei denn, Sie fordern die serverseitige Verschlüsselung explizit an.

Die Antwort-Header der folgenden REST-APIs geben den Header `x-amz-server-side-encryption` zurück, wenn ein Objekt unter Verwendung der serverseitigen Verschlüsselung gespeichert wird.

- PUT Object
- PUT Object – Kopieren
- POST Object
- Initiieren eines mehrteiligen Uploads
- Upload Part
- Hochladen eines Teiluploads – Kopieren
- Abschließen eines mehrteiligen Uploads
- Get Object

- Head Object

> **Note**
>
> Die Anforderungs-Header für die Verschlüsselung sollten nicht für `GET`-Anforderungen und `HEAD`-Anforderungen gesendet werden, wenn Ihr Objekt SSE-S3 verwendet oder ein HTTP 400 BadRequest-Fehler zurückgemeldet wurde.

Festlegen einer serverseitigen Verschlüsselung mit der AWS Management Console

Beim Hochladen eines Objekts unter Verwendung der AWS Management Console können Sie eine serverseitige Verschlüsselung festlegen. Ein Beispiel für das Hochladen von Objekten finden Sie unter Hochladen von S3-Objekten.

Wenn Sie ein Objekt unter Verwendung der AWS Management Console kopieren, kopiert die Konsole das Objekt unverändert. Das bedeutet, dass das Zielobjekt verschlüsselt wird, wenn die Kopierquelle verschlüsselt ist.Die Konsole ermöglicht Ihnen auch, einem Objekt Verschlüsselung hinzuzufügen. Weitere Informationen finden Sie unter Wie füge ich einem S3-Objekt Verschlüsselung hinzu?.

Weitere Infos

- Amazon S3-Standardverschlüsselung für S3-Buckets (p. 72)

Schützen von Daten durch eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C)

Die serverseitige Verschlüsselung dient zum Schutz ruhender Daten. Die serverseitige Verschlüsselung verschlüsselt nur die Objektdaten, nicht die Metadaten des Objekts. Die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) gestattet Ihnen, eigene Verschlüsselungsschlüssel einzurichten. Mit dem Verschlüsselungsschlüssel, den Sie als Teil Ihrer Anforderung bereitstellen, verwaltet Amazon S3 die Verschlüsselung, wenn es auf Datenträger schreibt, und die Entschlüsselung, wenn Sie auf Ihre Objekte zugreifen. Sie müssen also für die Datenverschlüsselung und -entschlüsselung keinen Code mehr verwalten. Sie müssen nur noch die von Ihnen bereitgestellten Verschlüsselungsschlüssel verwalten.

Wenn Sie ein Objekt hochladen, verwendet Amazon S3 den von Ihnen bereitgestellten Verschlüsselungsschlüssel, um eine AES-256-Verschlüsselung auf Ihre Daten anzuwenden, und entfernt den Verschlüsselungsschlüssel aus dem Speicher. Wenn Sie ein Objekt abrufen, müssen Sie denselben Verschlüsselungsschlüssel als Teil Ihrer Anforderung verwenden. Amazon S3 überprüft zunächst, ob der von Ihnen bereitgestellte Verschlüsselungsschlüssel übereinstimmt, und entschlüsselt vor der Rückgabe der Objektdaten das Objekt.

> **Important**
>
> Amazon S3 speichert den von Ihnen bereitgestellten Verschlüsselungsschlüssel nicht. Stattdessen wird ein zufällig mit einem Salt versehenen HMAC-Wert des Verschlüsselungsschlüssels gespeichert, um zukünftige Anfragen zu überprüfen. Der mit einem Salt versehene HMAC-Wert kann nicht verwendet werden, um den Wert des Verschlüsselungsschlüssels abzuleiten oder den Inhalt des verschlüsselten Objekts zu entschlüsseln. Das bedeutet, wenn Sie den Verschlüsselungsschlüssel verlieren, verlieren Sie das Objekt.

Übersicht über SSE-C

Dieser Abschnitt enthält eine Übersicht über SSE-C:

- Sie müssen HTTPS verwenden.

Important

Amazon S3 weist Anfragen über HTTP zurück, wenn SSE-C verwendet wird. Aus Sicherheitsgründen sollten Sie jeden Schlüssel, den Sie versehentlich mittels HTTP senden, als nicht vertrauenswürdig betrachten. Sie sollten den Schlüssel verwerfen und wie erforderlich rotieren.

- Das ETag in der Antwort ist nicht das MD5 der Objektdaten.
- Sie verwalten eine Zuweisung, welcher Verschlüsselungsschlüssel für die Verschlüsselung welches Objekts verwendet wurde. Amazon S3 speichert Verschlüsselungsschlüssel nicht. Sie sind dafür verantwortlich, zu verwalten, welchen Verschlüsselungsschlüssel Sie für welches Objekt angegeben haben.
  - Wenn Ihr Bucket versionsfähig ist, kann jede Objektversion, die Sie mit dieser Funktion hochladen, ihren eigenen Verschlüsselungsschlüssel haben. Sie sind dafür verantwortlich, zu verwalten, welcher Verschlüsselungsschlüssel für welche Objektversion verwendet wurde.
  - Sie verwalten die Verschlüsselungsschlüssel auf der Clientseite, deshalb verwalten Sie auch alle zusätzlichen Sicherungsmechanismen auf der Clientseite, wie beispielsweise die Schlüsselrotation.

Warning

Wenn Sie den Verschlüsselungsschlüssel verlieren, schlagen alle GET-Anfragen für ein Objekt ohne seinen Verschlüsselungsschlüssel fehl, und Sie verlieren das Objekt.

Themen

Weitere Informationen finden Sie unter den folgenden Themen:

- Festlegen einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung des AWS SDK for Java (p. 308)
- Festlegen einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung des AWS SDK für .NET (p. 313)
- Angabe einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung der REST API (p. 320)

Festlegen einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung des AWS SDK for Java

Das folgende Beispiel zeigt, wie die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) für Objekte angefordert wird. Das Beispiel führt die folgenden Operationen aus. Jede Operation zeigt, wie SSE-C-bezogene Header in der Anfrage angegeben werden:

- Put object (Objekt hochladen) – Lädt ein Objekt hoch und fordert die serverseitige Verschlüsselung mit einem vom Kunden bereitgestellten Verschlüsselungsschlüssel an.
- Get object (Objekt abrufen) – Lädt das im vorherigen Schritt hochgeladene Objekt herunter. Sie stellen in der Anfrage dieselben Verschlüsselungsinformationen bereit, die Sie beim Hochladen des Objekts bereitgestellt haben. Amazon S3 benötigt diese Informationen, um das Objekt zu entschlüsseln und an Sie zurückzugeben.
- Get object metadata (Objektmetadaten abrufen) – Ruft die Metadaten des Objekts ab. Sie stellen dieselben Verschlüsselungsinformationen bereit, die beim Erstellen des Objekts verwendet wurden.
- Copy object (Objekt kopieren) – Erstellt eine Kopie des zuvor hochgeladenen Objekts. Da das Quellobjekt mit SSE-C gespeichert wurde, müssen Sie seine Verschlüsselungsinformationen in Ihrer Kopieranfrage bereitstellen. Standardmäßig verschlüsselt Amazon S3 die Kopie des Objekts nur, wenn Sie dies ausdrücklich anfordern. Dieses Beispiel weist Amazon S3 dazu an, eine verschlüsselte Kopie des Objekts unter Verwendung eines neuen SSE-C-Schlüssels zu speichern.

Note

Dieses Beispiel zeigt, wie ein Objekt in einer einzigen Operation hochgeladen wird. Wenn Sie die API für mehrteilige Uploads verwenden, um große Objekte hochzuladen, geben Sie Verschlüsslungsinformationen genauso wie in diesem Beispiel veranschaulicht ein. Beispiele für mehrteilige Uploads mit AWS SDK for Java finden Sie unter Verwendung des AWS Java SDK für mehrteilige Uploads (High-Level-API) (p. 205) und Verwendung des AWS Java SDK für mehrteilige Uploads (Low-Level-API) (p. 209).

Um die erforderlichen Verschlüsselungsinformationen hinzuzufügen, schließen Sie in Ihre Anfrage einen `SSECustomerKey` ein. Weitere Informationen über die `SSECustomerKey`-Klasse finden Sie unter Angabe einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung der REST API (p. 320).

Weitere Informationen über SSE-C finden Sie unter Schützen von Daten durch eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) (p. 307). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import javax.crypto.KeyGenerator;
import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStreamReader;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;

public class ServerSideEncryptionUsingClientSideEncryptionKey {
    private static SSECustomerKey SSE_KEY;
    private static AmazonS3 S3_CLIENT;
    private static KeyGenerator KEY_GENERATOR;

    public static void main(String[] args) throws IOException, NoSuchAlgorithmException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String uploadFileName = "*** File path ***";
        String targetKeyName = "*** Target key name ***";

        // Create an encryption key.
        KEY_GENERATOR = KeyGenerator.getInstance("AES");
        KEY_GENERATOR.init(256, new SecureRandom());
        SSE_KEY = new SSECustomerKey(KEY_GENERATOR.generateKey());

        try {
            S3_CLIENT = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Upload an object.
```

```
            uploadObject(bucketName, keyName, new File(uploadFileName));

            // Download the object.
            downloadObject(bucketName, keyName);

            // Verify that the object is properly encrypted by attempting to retrieve it
            // using the encryption key.
            retrieveObjectMetadata(bucketName, keyName);

            // Copy the object into a new object that also uses SSE-C.
            copyObject(bucketName, keyName, targetKeyName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void uploadObject(String bucketName, String keyName, File file) {
        PutObjectRequest putRequest = new PutObjectRequest(bucketName, keyName,
file).withSSECustomerKey(SSE_KEY);
        S3_CLIENT.putObject(putRequest);
        System.out.println("Object uploaded");
    }

    private static void downloadObject(String bucketName, String keyName) throws
IOException {
        GetObjectRequest getObjectRequest = new GetObjectRequest(bucketName,
keyName).withSSECustomerKey(SSE_KEY);
        S3Object object = S3_CLIENT.getObject(getObjectRequest);

        System.out.println("Object content: ");
        displayTextInputStream(object.getObjectContent());
    }

    private static void retrieveObjectMetadata(String bucketName, String keyName) {
        GetObjectMetadataRequest getMetadataRequest = new
GetObjectMetadataRequest(bucketName, keyName)
                .withSSECustomerKey(SSE_KEY);
        ObjectMetadata objectMetadata = S3_CLIENT.getObjectMetadata(getMetadataRequest);
        System.out.println("Metadata retrieved. Object size: " +
objectMetadata.getContentLength());
    }

    private static void copyObject(String bucketName, String keyName, String targetKeyName)
            throws NoSuchAlgorithmException {
        // Create a new encryption key for target so that the target is saved using SSE-C.
        SSECustomerKey newSSEKey = new SSECustomerKey(KEY_GENERATOR.generateKey());

        CopyObjectRequest copyRequest = new CopyObjectRequest(bucketName, keyName,
bucketName, targetKeyName)
                .withSourceSSECustomerKey(SSE_KEY)
                .withDestinationSSECustomerKey(newSSEKey);

        S3_CLIENT.copyObject(copyRequest);
        System.out.println("Object copied");
    }

    private static void displayTextInputStream(S3ObjectInputStream input) throws
IOException {
        // Read one line at a time from the input stream and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
```

```
        String line;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

Weitere Amazon S3-Operationen mit SSE-C unter Verwendung des AWS SDK for Java

Das Beispiel im obigen Abschnitt zeigt, wie Sie eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) in den PUT-, GET-, Head- und Copy-Operationen anfordern. In diesem Abschnitt werden andere APIs beschrieben, die SSE-C unterstützen.

Um große Objekte hochzuladen, können Sie die API für mehrteilige Uploads verwenden (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)). Sie können entweder High-Level- und Low-Level-APIs zum Hochladen großer Objekte verwenden. Diese APIs unterstützen verschlüsselungsbezogene Header in der Anforderung.

- Wenn Sie die High-Level-`TransferManager`-API verwenden, geben Sie die verschlüsselungsspezifischen Header in der `PutObjectRequest` an (siehe Verwendung des AWS Java SDK für mehrteilige Uploads (High-Level-API) (p. 205)).
- Beim Verwenden der Low-Level-API geben Sie verschlüsselungsbezogene Informationen in der `InitiateMultipartUploadRequest` gefolgt von identischen Verschlüsselungsinformationen in jeder `UploadPartRequest` an. In Ihrer `CompleteMultipartUploadRequest` müssen Sie keine verschlüsselungsspezifischen Header angeben. Beispiele finden Sie unter Verwendung des AWS Java SDK für mehrteilige Uploads (Low-Level-API) (p. 209).

Der folgende Beispiel verwendet den `TransferManager`, um Objekte zu erstellen, und zeigt, wie Sie SSE-C-bezogene Informationen bereitstellen. Das Beispiel erledigt Folgendes:

- Erstellt ein Objekts mit der Methode `TransferManager.upload()`. In der `PutObjectRequest`-Instance stellen Sie Verschlüsselungsschlüsselinformationen bereit. Amazon S3 verschlüsselt das Objekt mit dem vom Kunden bereitgestellten Verschlüsselungsschlüssel.
- Erstellt eine Kopie des Objekts durch Aufrufen der `TransferManager.copy()`-Methode. Das Beispiel weist Amazon S3 an, die Objektkopie unter Verwendung eines neuen `SSECustomerKey` zu verschlüsseln. Da das Quellobjekt mittels SSE-C verschlüsselt ist, stellt `CopyObjectRequest` auch den Verschlüsselungsschlüssel des Quellobjekts bereit. Auf diese Weise kann Amazon S3 das Objekt vor dem Kopieren entschlüsseln.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;
import com.amazonaws.services.s3.model.PutObjectRequest;
import com.amazonaws.services.s3.model.SSECustomerKey;
import com.amazonaws.services.s3.transfer.Copy;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;
```

```
import javax.crypto.KeyGenerator;
import java.io.File;
import java.security.SecureRandom;

public class ServerSideEncryptionCopyObjectUsingHLwithSSEC {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String fileToUpload = "*** File path ***";
        String keyName = "*** New object key name ***";
        String targetKeyName = "*** Key name for object copy ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // Create an object from a file.
            PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName, keyName,
 new File(fileToUpload));

            // Create an encryption key.
            KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
            keyGenerator.init(256, new SecureRandom());
            SSECustomerKey sseCustomerEncryptionKey = new
 SSECustomerKey(keyGenerator.generateKey());

            // Upload the object. TransferManager uploads asynchronously, so this call
 returns immediately.
            putObjectRequest.setSSECustomerKey(sseCustomerEncryptionKey);
            Upload upload = tm.upload(putObjectRequest);

            // Optionally, wait for the upload to finish before continuing.
            upload.waitForCompletion();
            System.out.println("Object created.");

            // Copy the object and store the copy using SSE-C with a new key.
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(bucketName,
 keyName, bucketName, targetKeyName);
            SSECustomerKey sseTargetObjectEncryptionKey = new
 SSECustomerKey(keyGenerator.generateKey());
            copyObjectRequest.setSourceSSECustomerKey(sseCustomerEncryptionKey);
            copyObjectRequest.setDestinationSSECustomerKey(sseTargetObjectEncryptionKey);

            // Copy the object. TransferManager copies asynchronously, so this call returns
 immediately.
            Copy copy = tm.copy(copyObjectRequest);

            // Optionally, wait for the upload to finish before continuing.
            copy.waitForCompletion();
            System.out.println("Copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
}
```

Festlegen einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung des AWS SDK für .NET

Das folgende C#-Codebeispiel zeigt, wie die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) funktioniert. Das Beispiel führt die folgenden Operationen aus. Jede Operation zeigt, wie SSE-C-bezogene Header in der Anfrage angegeben werden.

- Put object (Objekt hochladen) – Lädt ein Objekt hoch und fordert die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln an.
- Get object (Objekt abrufen) – Lädt das im vorherigen Schritt hochgeladene Objekt herunter. Das Beispiel stellt dieselben Verschlüsselungsinformationen bereit, die beim Hochladen des Objekts bereitgestellt wurden. Amazon S3 benötigt diese Informationen, um das Objekt zu entschlüsseln und an Sie zurückzugeben.
- Get object metadata (Objektmetadaten abrufen) – Stellt dieselben Verschlüsselungsinformationen, die beim Erstellen des Objekts verwendet wurden, zum Abruf der Metadaten des Objekts bereit.
- Copy object (Objekt kopieren) – Erstellt eine Kopie des hochgeladenen Objekts. Da das Quellobjekt mit SSE-C gespeichert wurde, muss die Kopieranforderung Verschlüsselungsinformationen bereitstellen. Standardmäßig wird die Kopie eines Objekts nicht von Amazon S3 verschlüsselt. Der Code weist Amazon S3 durch Angabe verschlüsselungsbezogener Informationen dazu an, das kopierte Objekt mit SSE-C zu verschlüsseln. Außerdem speichert er das Ziel.

> **Note**
>
> Beispiele für das Hochladen großer Objekte mithilfe der API für mehrteilige Uploads finden Sie unter Verwendung des AWS SDK für .NET für mehrteilige Uploads (High-Level-API) (p. 213) und Verwendung des AWS SDK für .NET für mehrteilige Uploads (Low-Level-API) (p. 219).

Weitere Informationen über SSE-C finden Sie unter Schützen von Daten durch eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) (p. 307). Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SSEClientEncryptionKeyObjectOperationsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for new object created ***";
        private const string copyTargetKeyName = "*** key name for object copy ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
```

```
            client = new AmazonS3Client(bucketRegion);
            ObjectOpsUsingClientEncryptionKeyAsync().Wait();
        }
        private static async Task ObjectOpsUsingClientEncryptionKeyAsync()
        {
            try
            {
                // Create an encryption key.
                Aes aesEncryption = Aes.Create();
                aesEncryption.KeySize = 256;
                aesEncryption.GenerateKey();
                string base64Key = Convert.ToBase64String(aesEncryption.Key);

                // 1. Upload the object.
                PutObjectRequest putObjectRequest = await UploadObjectAsync(base64Key);
                // 2. Download the object and verify that its contents matches what you
uploaded.
                await DownloadObjectAsync(base64Key, putObjectRequest);
                // 3. Get object metadata and verify that the object uses AES-256
encryption.
                await GetObjectMetadataAsync(base64Key);
                // 4. Copy both the source and target objects using server-side encryption
with
                //    a customer-provided encryption key.
                await CopyObjectAsync(aesEncryption, base64Key);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing an
object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        private static async Task<PutObjectRequest> UploadObjectAsync(string base64Key)
        {
            PutObjectRequest putObjectRequest = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                ContentBody = "sample text",
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };
            PutObjectResponse putObjectResponse = await
client.PutObjectAsync(putObjectRequest);
            return putObjectRequest;
        }
        private static async Task DownloadObjectAsync(string base64Key, PutObjectRequest
putObjectRequest)
        {
            GetObjectRequest getObjectRequest = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                // Provide encryption information for the object stored in Amazon S3.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };
```

```
            using (GetObjectResponse getResponse = await
client.GetObjectAsync(getObjectRequest))
            using (StreamReader reader = new StreamReader(getResponse.ResponseStream))
            {
                string content = reader.ReadToEnd();
                if (String.Compare(putObjectRequest.ContentBody, content) == 0)
                    Console.WriteLine("Object content is same as we uploaded");
                else
                    Console.WriteLine("Error...Object content is not same.");

                if (getResponse.ServerSideEncryptionCustomerMethod ==
ServerSideEncryptionCustomerMethod.AES256)
                    Console.WriteLine("Object encryption method is AES256, same as we
set");
                else
                    Console.WriteLine("Error...Object encryption method is not the same as
AES256 we set");

                // Assert.AreEqual(putObjectRequest.ContentBody, content);
                // Assert.AreEqual(ServerSideEncryptionCustomerMethod.AES256,
getResponse.ServerSideEncryptionCustomerMethod);
            }
        }
        private static async Task GetObjectMetadataAsync(string base64Key)
        {
            GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest
            {
                BucketName = bucketName,
                Key = keyName,

                // The object stored in Amazon S3 is encrypted, so provide the necessary
encryption information.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };

            GetObjectMetadataResponse getObjectMetadataResponse = await
client.GetObjectMetadataAsync(getObjectMetadataRequest);
            Console.WriteLine("The object metadata show encryption method used is: {0}",
getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
            // Assert.AreEqual(ServerSideEncryptionCustomerMethod.AES256,
getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
        }
        private static async Task CopyObjectAsync(Aes aesEncryption, string base64Key)
        {
            aesEncryption.GenerateKey();
            string copyBase64Key = Convert.ToBase64String(aesEncryption.Key);

            CopyObjectRequest copyRequest = new CopyObjectRequest
            {
                SourceBucket = bucketName,
                SourceKey = keyName,
                DestinationBucket = bucketName,
                DestinationKey = copyTargetKeyName,
                // Information about the source object's encryption.
                CopySourceServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                CopySourceServerSideEncryptionCustomerProvidedKey = base64Key,
                // Information about the target object's encryption.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = copyBase64Key
            };
            await client.CopyObjectAsync(copyRequest);
```

```
        }
    }
}
```

Andere Amazon S3-Operationen und SSE-C

Das Beispiel im obigen Abschnitt zeigt, wie Sie eine serverseitige Verschlüsselung mit einem vom Kunden bereitgestellten Schlüssel (SSE-C) in den PUT-, GET-, Head- und Copy-Operationen anfordern. In diesem Abschnitt werden andere Amazon S3-APIs beschrieben, die SSE-C unterstützen.

Um große Objekte hochzuladen, können Sie die API für mehrteilige Uploads verwenden (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)). AWS SDK for .NET bietet sowohl High-Level- als auch Low-Level-APIs zum Hochladen großer Objekte. Diese APIs unterstützen verschlüsselungsbezogene Header in der Anforderung.

- Bei Verwendung der High-Level-`Transfer-Utility` -API stellen Sie wie im Folgenden dargestellt verschlüsselungsspezifische Header in der `TransferUtilityUploadRequest` bereit. Codebeispiele finden Sie unter Verwendung des AWS SDK für .NET für mehrteilige Uploads (High-Level-API) (p. 213).

```
TransferUtilityUploadRequest request = new TransferUtilityUploadRequest()
{
    FilePath = filePath,
    BucketName = existingBucketName,
    Key = keyName,
    // Provide encryption information.
    ServerSideEncryptionCustomerMethod = ServerSideEncryptionCustomerMethod.AES256,
    ServerSideEncryptionCustomerProvidedKey = base64Key,
};
```

- Beim Verwenden der Low-Level-API geben Sie verschlüsselungsbezogene Informationen in der Anforderung zum Starten des mehrteiligen Uploads gefolgt von identischen Verschlüsselungsinformationen in den nachfolgenden Anforderungen zum mehrteiligen Upload an. In der vollständigen Anforderung zum mehrteiligen Upload müssen Sie keine verschlüsselungsspezifischen Header angeben. Beispiele finden Sie unter Verwendung des AWS SDK für .NET für mehrteilige Uploads (Low-Level-API) (p. 219).

  Nachfolgend finden Sie ein Beispiel für einen mehrteiligen Low-Level-Upload, der eine Kopie eines vorhandenen großen Objekts erstellt. Im Beispiel wird das zu kopierende Objekt in Amazon S3 mittels SSE-C gespeichert und Sie möchten das Zielobjekt ebenfalls mittels SSE-C speichern. Im Beispiel führen Sie folgende Schritte aus:

  - Initiieren Sie eine mehrteilige Upload-Anfrage, indem Sie einen Verschlüsselungsschlüssel und zugehörige Informationen bereitstellen.
  - Stellen Sie Verschlüsselungsschlüssel für das Quell- und das Zielobjekt sowie die zugehörigen Informationen in der `CopyPartRequest` bereit.
  - Ermitteln Sie die Größe des zu kopierenden Quellobjekts, indem Sie die Objekt-Metadaten abrufen.
  - Laden Sie die Objekte in Teilen mit je 5 MB hoch.

  #### Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
```

```
{
    class SSECLowLevelMPUcopyObjectTest
    {
        private const string existingBucketName = "*** bucket name ***";
        private const string sourceKeyName      = "*** source object key name ***";
        private const string targetKeyName      = "*** key name for the target object
 ***";
        private const string filePath           = @"*** file path ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CopyObjClientEncryptionKeyAsync().Wait();
        }

        private static async Task CopyObjClientEncryptionKeyAsync()
        {
            Aes aesEncryption = Aes.Create();
            aesEncryption.KeySize = 256;
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);

            await CreateSampleObjUsingClientEncryptionKeyAsync(base64Key, s3Client);

            await CopyObjectAsync(s3Client, base64Key);
        }
        private static async Task CopyObjectAsync(IAmazonS3 s3Client, string base64Key)
        {
            List<CopyPartResponse> uploadResponses = new List<CopyPartResponse>();

            // 1. Initialize.
            InitiateMultipartUploadRequest initiateRequest = new
 InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = targetKeyName,
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // 2. Upload Parts.
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB
            long firstByte = 0;
            long lastByte = partSize;

            try
            {
                // First find source object size. Because object is stored encrypted with
                // customer provided key you need to provide encryption information in
 your request.
                GetObjectMetadataRequest getObjectMetadataRequest = new
 GetObjectMetadataRequest()
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                    ServerSideEncryptionCustomerProvidedKey = base64Key // " * **source
 object encryption key ***"
                };
```

```
                GetObjectMetadataResponse getObjectMetadataResponse = await
 s3Client.GetObjectMetadataAsync(getObjectMetadataRequest);

                long filePosition = 0;
                for (int i = 1; filePosition < getObjectMetadataResponse.ContentLength; i
++)
                {
                    CopyPartRequest copyPartRequest = new CopyPartRequest
                    {
                        UploadId = initResponse.UploadId,
                        // Source.
                        SourceBucket = existingBucketName,
                        SourceKey = sourceKeyName,
                        // Source object is stored using SSE-C. Provide encryption
 information.
                        CopySourceServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                        CopySourceServerSideEncryptionCustomerProvidedKey =
 base64Key, //"***source object encryption key ***",
                        FirstByte = firstByte,
                        // If the last part is smaller then our normal part size then use
 the remaining size.
                        LastByte = lastByte > getObjectMetadataResponse.ContentLength ?
                            getObjectMetadataResponse.ContentLength - 1 : lastByte,

                        // Target.
                        DestinationBucket = existingBucketName,
                        DestinationKey = targetKeyName,
                        PartNumber = i,
                        // Encryption information for the target object.
                        ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key
                    };
                    uploadResponses.Add(await s3Client.CopyPartAsync(copyPartRequest));
                    filePosition += partSize;
                    firstByte += partSize;
                    lastByte += partSize;
                }

                // Step 3: complete.
                CompleteMultipartUploadRequest completeRequest = new
 CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = targetKeyName,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (Exception exception)
            {
                Console.WriteLine("Exception occurred: {0}", exception.Message);
                AbortMultipartUploadRequest abortMPURequest = new
 AbortMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = targetKeyName,
                    UploadId = initResponse.UploadId
                };
                s3Client.AbortMultipartUpload(abortMPURequest);
            }
```

```
        }
        private static async Task CreateSampleObjUsingClientEncryptionKeyAsync(string
base64Key, IAmazonS3 s3Client)
        {
            // List to store upload part responses.
            List<UploadPartResponse> uploadResponses = new List<UploadPartResponse>();

            // 1. Initialize.
            InitiateMultipartUploadRequest initiateRequest = new
InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = sourceKeyName,
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };

            InitiateMultipartUploadResponse initResponse =
               await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // 2. Upload Parts.
            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                    {
                        BucketName = existingBucketName,
                        Key = sourceKeyName,
                        UploadId = initResponse.UploadId,
                        PartNumber = i,
                        PartSize = partSize,
                        FilePosition = filePosition,
                        FilePath = filePath,
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key
                    };

                    // Upload part and add response to our list.
                    uploadResponses.Add(await s3Client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                // Step 3: complete.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                    //PartETags = new List<PartETag>(uploadResponses)

                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);

            }
```

```
            catch (Exception exception)
            {
                Console.WriteLine("Exception occurred: {0}", exception.Message);
                AbortMultipartUploadRequest abortMPURequest = new
 AbortMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId
                };
                await s3Client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
    }
}
```

Angabe einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung der REST API

Zum Zeitpunkt der Objekterstellung über die REST-API können Sie die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) angeben. Wenn Sie SSE-C verwenden, müssen Sie mithilfe der folgenden Anforderungs-Header Informationen zum Verschlüsselungsschlüssel bereitstellen.

Name	Beschreibung
`x-amz-server-side-encryption-customer-algorithm`	Verwenden Sie diesen Header, um den Verschlüsselungsalgorithmus anzugeben. Der Header-Wert muss "AES256" sein.
`x-amz-server-side-encryption-customer-key`	Verwenden Sie diesen Header, um den base64-codierten 256-bit-Verschlüsselungsschlüssel für Amazon S3 bereitzustellen, mit dem Ihre Daten verschlüsselt und entschlüsselt werden.
`x-amz-server-side-encryption-customer-key-MD5`	Mittels dieses Headers können Sie den base64-kodierten 128-Bit-MD5-Digest des Verschlüsselungsschlüssels entsprechend RFC 1321 bereitstellen. Amazon S3 verwendet diesen Header für eine Integritätsprüfung der Nachricht, um sicherzustellen, dass der Verschlüsselungsschlüssel fehlerfrei übertragen wurde.

Sie können AWS SDK Wrapper-Bibliotheken verwenden, um Ihrer Anfrage diese Header hinzuzufügen. Falls nötig, können Sie auch die REST API-Aufrufe in Amazon S3 direkt von Ihrer Anwendung aus durchführen.

> **Note**
>
> Sie können die Amazon S3-Konsole nicht verwenden, um ein Objekt hochzuladen und SSE-C anzufordern. Sie können die Konsole auch nicht verwenden, um ein vorhandenes Objekt zu aktualisieren (beispielsweise durch Ändern der Speicherklasse oder Hinzufügen von Metadaten), das mittels SSE-C gespeichert wurde.

Amazon S3-Rest-APIs, die SSE-C unterstützen

Die folgenden Amazon S3-APIs unterstützen die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C).

- Operation GET: Sie können beim Abrufen von Objekten über die GET API (siehe GET Object) die Anforderungs-Header angeben. Für Objekte mit SSE-Verschlüsselung werden keine Torrents unterstützt.

- Operation HEAD: Um Objektmetadaten über die HEAD API abzurufen (siehe HEAD Object) können Sie diese Anforderungs-Header angeben.
- Operation PUT: Sie können beim Hochladen von Daten über die PUT Object API (siehe PUT Object) diese Anforderungs-Header angeben.
- Multipart Upload: Sie können beim Hochladen großer Objekte über die Multipart Upload API diese Header angeben. Sie geben diese Header in der Initiierungsanforderung (siehe Initiieren mehrteiliger Uploads) und in jeder nachfolgenden Anforderung für einen Teil-Upload an (siehe Hochladen von Teilen oder

  Hochladen von Teilen – Kopieren).

) enthalten. Für jede teilweise Upload-Anfrage muss dieselbe Verschlüsslungsinformation angegeben werden, wie diejenige, die Sie in der Initiierungsanfrage für den mehrteiligen Upload angegeben haben.
- Operation POST: Bei Verwendung einer POST-Operation zum Hochladen eines Objekts (siehe POST Object) geben Sie die Informationen in den Formularfeldern und nicht in den Anforderungs-Headern an.
- Kopieroperation: Wenn Sie ein Objekt kopieren (siehe PUT Object – Kopieren), erhalten Sie ein Quell- und ein Zielobjekt:
  - Wenn das Zielobjekt unter Verwendung von serverseitiger Verschlüsselung mit AWS-verwalteten Schlüsseln verschlüsselt werden soll, müssen Sie den Anfrage-Header `x-amz-server-side-encryption` bereitstellen.
  - Wenn das Zielobjekt unter Verwendung von SSE-C verschlüsselt werden soll, müssen Sie die Verschlüsselungsinformationen unter Verwendung der drei Header angeben, die in der obigen Tabelle beschrieben sind.
  - Wenn das Quellobjekt unter Verwendung von SSE-C verschlüsselt ist, müssen Sie die Verschlüsselungsschlüsselinformationen unter Verwendung der folgenden Header angeben, sodass Amazon S3 das Objekt zum Kopieren verschlüsseln kann.

Name	Beschreibung
`x-amz-copy-source-server-side-encryption-customer-algorithm`	Verwenden Sie diesen Header, um den Verschlüsselungsalgorithmus anzugeben, den Amazon S3 für die Entschlüsselung des Quellobjekts verwenden soll. Dieser Wert muss `AES256` lauten.
`x-amz-copy-source-server-side-encryption-customer-key`	Verwenden Sie diesen Header, um den base64-codierten 256-bit-Verschlüsselungsschlüssel für Amazon S3 bereitzustellen, mit dem das Quellobjekt entschlüsselt werden soll. Dieser Verschlüsselungsschlüssel muss derjenige sein, den Sie Amazon S3 beim Erstellen des Quellobjekts bereitgestellt haben. Andernfalls kann Amazon S3 das Objekt nicht entschlüsseln.
`x-amz-copy-source-server-side-encryption-customer-key-MD5`	Verwenden Sie diesen Header, um den base64-codierten 128-bit-MD5-Digest des Verschlüsselungsschlüssels gemäß RFC 1321 bereitzustellen.

Vorsignierte URLs und SSE-C

Sie können eine vorsignierte URL erstellen, die für Operationen wie das Hochladen eines neuen Objekts, das Abrufen eines vorhandenen Objekts oder von Objekt-Metadaten verwendet werden kann. Vorsignierte URLs unterstützen die SSE-C wie folgt:

- Beim Erstellen einer vorsignierten URL müssen Sie den Algorithmus unter Verwendung von `x-amz-server-side-encryption-customer-algorithm` in der Signaturberechnung angeben.

- Wenn Sie die vorsignierte URL verwenden, um ein neues Objekt hochzuladen, ein vorhandenes Objekt abzurufen oder nur Objekt-Metadaten abzurufen, müssen Sie in Ihrer Client-Anwendung alle Verschlüsselungs-Header angeben.

  Note

  Für andere Objekte als SSE-C-Objekte können Sie eine vorsignierte URL generieren und diese direkt in einen Browser kopieren, um beispielsweise auf die Daten zuzugreifen.
  Dies ist für SSE-C-Objekte jedoch nicht möglich, da Sie zusätzlich zur vorsignierten URL auch HTTP-Header einfügen müssen, die für SSE-C-Objekte spezifisch sind. Daher können Sie die vorsignierte URL für SSE-C-Objekte ausschließlich programmgesteuert verwenden.

Weitere Infos

- Festlegen einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung des AWS SDK für .NET (p. 313)
- Festlegen einer serverseitigen Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln unter Verwendung des AWS SDK for Java (p. 308)

Schützen von Daten mithilfe clientseitiger Verschlüsselung

Clientseitige Verschlüsselung bezieht sich auf die Verschlüsselung von Daten, bevor sie zu Amazon S3 gesendet werden. Zum Aktivieren der clientseitigen Verschlüsselung haben Sie die folgenden Möglichkeiten:

- Verwenden Sie einen Kundenstammschlüssel (CMK), der in AWS Key Management Service (AWS KMS) gespeichert ist.
- Verwenden Sie einen Masterschlüssel, den Sie innerhalb Ihrer Anwendung speichern.

Die folgenden AWS-SDKs unterstützen die clientseitige Verschlüsselung:

- AWS SDK für .NET
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK für PHP
- AWS SDK für Ruby
- AWS SDK für C++

Option 1: Verwenden eines in AWS KMS gespeicherten CMKs

- Wenn Sie ein Objekt hochladen — Unter Verwendung der CMK-ID (Customer Master Key) sendet der Client zunächst eine Anforderung an AWS KMS für ein CMK, mit der er Ihre Objektdaten verschlüsseln kann. AWS KMS gibt zwei Versionen eines zufällig erzeugten Datenschlüssels zurück:
  - Eine Klartextversion des Datenschlüssels, mit dem der Client die Objektdaten verschlüsselt.
  - Einen verschlüsselten Blob mit dem gleichen Datenschlüssel, den der Client als Objekt-Metadaten in Amazon S3 hochlädt.

    Note

    Der Client erhält für jedes Objekt, das er hochlädt, einen eindeutigen Datenschlüssel.
- Beim Herunterladen eines Objekts— Der Client lädt das verschlüsselte Objekt von Amazon S3 zusammen mit der verschlüsselten Blob-Version des als Objekt-Metadaten gespeicherten

Datenschlüssels herunter. Der Client sendet dann den verschlüsselten Blob an AWS KMS, um die Klartextversion des Datenschlüssels zu erhalten, sodass er die Objektdaten entschlüsseln kann.

Weitere Informationen zu AWS KMS finden Sie unter Was ist AWS Key Management Service? in der AWS Key Management Service Developer Guide.

Example

Im folgenden Beispiel wird ein Objekt zu Amazon S3 unter Verwendung von AWS KMS mit AWS SDK for Java hochgeladen. Das Beispiel verwendet einen von AWS verwaltetes CMK, um Daten clientseitig zu verschlüsseln, bevor es sie zu Amazon S3 hochlädt. Wenn Sie bereits über einen CMK verfügen, können Sie diesen verwenden, indem Sie den Wert der Variablen `kms_cmk_id` im Beispiel-Code angeben. Wenn Sie keinen CMK besitzen oder einen anderen benötigen, können Sie ihn über die Java-API generieren. Das Beispiel veranschaulicht, wie Sie einen CMK generieren.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.RegionUtils;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.kms.AWSKMS;
import com.amazonaws.services.kms.AWSKMSClientBuilder;
import com.amazonaws.services.kms.model.CreateKeyResult;
import com.amazonaws.services.s3.AmazonS3Encryption;
import com.amazonaws.services.s3.AmazonS3EncryptionClientBuilder;
import com.amazonaws.services.s3.model.CryptoConfiguration;
import com.amazonaws.services.s3.model.KMSEncryptionMaterialsProvider;
import com.amazonaws.services.s3.model.S3Object;
import com.amazonaws.services.s3.model.S3ObjectInputStream;

import java.io.ByteArrayOutputStream;
import java.io.IOException;

public class UploadObjectKMSKey {

    public static void main(String[] args) throws IOException {
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key name ***";
        Regions clientRegion = Regions.DEFAULT_REGION;
        String kms_cmk_id = "*** AWS KMS customer master key ID ***";
        int readChunkSize = 4096;

        try {
            // Optional: If you don't have a KMS key (or need another one),
            // create one. This example creates a key with AWS-created
            // key material.
            AWSKMS kmsClient = AWSKMSClientBuilder.standard()
                    .withRegion(clientRegion)
                    .build();
            CreateKeyResult keyResult = kmsClient.createKey();
            kms_cmk_id = keyResult.getKeyMetadata().getKeyId();

            // Create the encryption client.
            KMSEncryptionMaterialsProvider materialProvider = new
 KMSEncryptionMaterialsProvider(kms_cmk_id);
            CryptoConfiguration cryptoConfig = new CryptoConfiguration()
                    .withAwsKmsRegion(RegionUtils.getRegion(clientRegion.toString()));
            AmazonS3Encryption encryptionClient =
 AmazonS3EncryptionClientBuilder.standard()
```

```
                    .withCredentials(new ProfileCredentialsProvider())
                    .withEncryptionMaterials(materialProvider)
                    .withCryptoConfiguration(cryptoConfig)
                    .withRegion(clientRegion).build();

            // Upload an object using the encryption client.
            String origContent = "S3 Encrypted Object Using KMS-Managed Customer Master
 Key.";
            int origContentLength = origContent.length();
            encryptionClient.putObject(bucketName, keyName, origContent);

            // Download the object. The downloaded object is still encrypted.
            S3Object downloadedObject = encryptionClient.getObject(bucketName, keyName);
            S3ObjectInputStream input = downloadedObject.getObjectContent();

            // Decrypt and read the object and close the input stream.
            byte[] readBuffer = new byte[readChunkSize];
            ByteArrayOutputStream baos = new ByteArrayOutputStream(readChunkSize);
            int bytesRead = 0;
            int decryptedContentLength = 0;

            while ((bytesRead = input.read(readBuffer)) != -1) {
                baos.write(readBuffer, 0, bytesRead);
                decryptedContentLength += bytesRead;
            }
            input.close();

            // Verify that the original and decrypted contents are the same size.
            System.out.println("Original content length: " + origContentLength);
            System.out.println("Decrypted content length: " + decryptedContentLength);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Option 2: Verwenden eines innerhalb Ihrer Anwendung gespeicherten Masterschlüssels

Dieser Abschnitt veranschaulicht, wie Sie einen in Ihrer Anwendung gespeicherten Masterschlüssel für die clientseitige Datenverschlüsselung verwenden.

> **Important**
>
> Ihre clientseitigen Hauptschlüssel und Ihre unverschlüsselten Daten werden niemals an AWS gesendet. Es ist wichtig, dass Sie Ihre Verschlüsselungsschlüssel sicher verwalten. Wenn die Schlüssel verloren gehen, können Sie Ihre Daten nicht mehr entschlüsseln.

Es funktioniert so:

- Beim Hochladen eines Objekts – Sie stellen dem Amazon S3-Verschlüsselungsclient einen clientseitigen Masterschlüssel bereit. Der Client nutzt den Masterschlüssel nur zur Verschlüsselung des Datenverschlüsselungsschlüssels, den nach dem Zufallsprinzip generiert wird. Der Prozess funktioniert wie folgt:
  1. Der Amazon S3-Verschlüsselungs-Client generiert lokal einen einmaligen symmetrischen Schlüssel (auch als Datenverschlüsselungsschlüssel oder Datenschlüssel bezeichnet). Er verwendet eine

Klartextversion, um die Daten eines einzelnen Amazon S3-Objekts zu verschlüsseln. Der Client generiert einen separaten Datenschlüssel für jedes Objekt.

2. Der Client verschlüsselt den Datenverschlüsselungsschlüssel mithilfe des von Ihnen bereitgestellten Masterschlüssels. Der Client lädt den verschlüsselten Datenschlüssel und seine Beschreibung als Teil der Objekt-Metadaten hoch. Der Client bestimmt anhand der Materialbeschreibung, welcher clientseitige Masterschlüssel für die Entschlüsselung verwendet werden soll.
3. Der Client lädt die verschlüsselten Daten zu Amazon S3 hoch und speichert den verschlüsselten Datenschlüssel in Form von Objektmetadaten (`x-amz-meta-x-amz-key`) in Amazon S3.

- Beim Herunterladen eines Objekts – Der Client lädt das verschlüsselte Objekt von Amazon S3 herunter. Anhand der Materialbeschreibung aus den Metadaten des Objekts bestimmt der Client, welcher Masterschlüssel verwendet werden soll, um den Datenschlüssel zu entschlüsseln. Der Client verwendet diesen Masterschlüssel, um den Datenschlüssel zu entschlüsseln, und verwendet den Datenschlüssel dann zum Entschlüsseln des Objekts.

Der von Ihnen bereitgestellte clientseitige Masterschlüssel kann ein symmetrischer Schlüssel oder ein öffentliches/privates Schlüsselpaar sein. Die folgenden Abschnitte veranschaulichen, wie beide Arten von Schlüsseln verwendet werden.

Weitere Informationen finden Sie unter Clientseitige Datenverschlüsselung mit AWS SDK for Java und Amazon S3 .

> **Note**
>
> Wenn Sie während der erstmaligen Verwendung der Verschlüsselungs-API eine Verschlüsselungsfehlermeldung erhalten, enthält Ihre Version des JDK möglicherweise eine Java Cryptography Extension (JCE)-Jurisdiktionsrichtlinie, die die maximale Schlüssellänge für Verschlüsselungs- und Entschlüsselungstransformationen auf 128 Bits begrenzt. Das AWS-SDK benötigt eine maximale Schlüssellänge von 256 Bit. Um Ihre maximale Schlüssellänge zu überprüfen, rufen Sie die Methode`getMaxAllowedKeyLength()` der Klasse `javax.crypto.Cipher` auf. Um die Schlüssellängenbeschränkung zu entfernen, installieren Sie die Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy-Dateien von der Java SE-Downloadseite.

Example

Das folgende Beispiel veranschaulicht, wie Sie diese Aufgaben ausführen:

- Generieren eines 256-Bit-AES-Schlüssels
- Speichern und Laden des AES-Schlüssels zu und aus dem Dateisystem
- Verwenden des AES-Schlüssels zum clientseitigen Verschlüsseln von Daten, bevor sie an Amazon S3 gesendet werden
- Verwenden des AES-Schlüssels zum Entschlüsseln von Daten, die von Amazon S3 empfangen wurden
- Überprüfen, ob die entschlüsselten Daten mit den Originaldaten übereinstimmen

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3EncryptionClientBuilder;
import com.amazonaws.services.s3.model.*;
```

```
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.io.*;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.X509EncodedKeySpec;

public class S3ClientSideEncryptionSymMasterKey {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String objectKeyName = "*** Object key name ***";
        String masterKeyDir = System.getProperty("java.io.tmpdir");
        String masterKeyName = "secret.key";

        // Generate a symmetric 256-bit AES key.
        KeyGenerator symKeyGenerator = KeyGenerator.getInstance("AES");
        symKeyGenerator.init(256);
        SecretKey symKey = symKeyGenerator.generateKey();

        // To see how it works, save and load the key to and from the file system.
        saveSymmetricKey(masterKeyDir, masterKeyName, symKey);
        symKey = loadSymmetricAESKey(masterKeyDir, masterKeyName, "AES");

        try {
            // Create the Amazon S3 encryption client.
            EncryptionMaterials encryptionMaterials = new EncryptionMaterials(symKey);
            AmazonS3 s3EncryptionClient = AmazonS3EncryptionClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withEncryptionMaterials(new
 StaticEncryptionMaterialsProvider(encryptionMaterials))
                    .withRegion(clientRegion)
                    .build();

            // Upload a new object. The encryption client automatically encrypts it.
            byte[] plaintext = "S3 Object Encrypted Using Client-Side Symmetric Master
 Key.".getBytes();
            s3EncryptionClient.putObject(new PutObjectRequest(bucketName,
                    objectKeyName,
                    new ByteArrayInputStream(plaintext),
                    new ObjectMetadata()));

            // Download and decrypt the object.
            S3Object downloadedObject = s3EncryptionClient.getObject(bucketName,
 objectKeyName);
            byte[] decrypted =
 com.amazonaws.util.IOUtils.toByteArray(downloadedObject.getObjectContent());

            // Verify that the data that you downloaded is the same as the original data.
            System.out.println("Plaintext: " + new String(plaintext));
            System.out.println("Decrypted text: " + new String(decrypted));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
    private static void saveSymmetricKey(String masterKeyDir, String masterKeyName,
 SecretKey secretKey) throws IOException {
        X509EncodedKeySpec x509EncodedKeySpec = new
 X509EncodedKeySpec(secretKey.getEncoded());
        FileOutputStream keyOutputStream = new FileOutputStream(masterKeyDir +
 File.separator + masterKeyName);
        keyOutputStream.write(x509EncodedKeySpec.getEncoded());
        keyOutputStream.close();
    }

    private static SecretKey loadSymmetricAESKey(String masterKeyDir, String masterKeyName,
 String algorithm)
            throws IOException, NoSuchAlgorithmException, InvalidKeySpecException,
 InvalidKeyException {
        // Read the key from the specified file.
        File keyFile = new File(masterKeyDir + File.separator + masterKeyName);
        FileInputStream keyInputStream = new FileInputStream(keyFile);
        byte[] encodedPrivateKey = new byte[(int) keyFile.length()];
        keyInputStream.read(encodedPrivateKey);
        keyInputStream.close();

        // Reconstruct and return the master key.
        return new SecretKeySpec(encodedPrivateKey, "AES");
    }
}
```

Das folgende Beispiel veranschaulicht, wie Sie diese Aufgaben ausführen:

- Generieren eines 1024-Bit-RSA-Schlüsselpaars
- Speichern und Laden der RSA-Schlüssel zu und aus dem Dateisystem
- Verwenden der RSA-Schlüssel zum clientseitigen Verschlüsseln von Daten, bevor sie an Amazon S3 gesendet werden
- Verwenden der RSA-Schlüssel zum Entschlüsseln von Daten, die von Amazon S3 empfangen wurden
- Überprüfen, ob die entschlüsselten Daten mit den Originaldaten übereinstimmen

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3EncryptionClientBuilder;
import com.amazonaws.services.s3.model.*;
import com.amazonaws.util.IOUtils;

import java.io.*;
import java.security.*;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;

public class S3ClientSideEncryptionAsymmetricMasterKey {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String objectKeyName = "*** Key name ***";
```

```
        String rsaKeyDir = System.getProperty("java.io.tmpdir");
        String publicKeyName = "public.key";
        String privateKeyName = "private.key";

        // Generate a 1024-bit RSA key pair.
        KeyPairGenerator keyGenerator = KeyPairGenerator.getInstance("RSA");
        keyGenerator.initialize(1024, new SecureRandom());
        KeyPair origKeyPair = keyGenerator.generateKeyPair();

        // To see how it works, save and load the key pair to and from the file system.
        saveKeyPair(rsaKeyDir, publicKeyName, privateKeyName, origKeyPair);
        KeyPair keyPair = loadKeyPair(rsaKeyDir, publicKeyName, privateKeyName, "RSA");

        try {
            // Create the encryption client.
            EncryptionMaterials encryptionMaterials = new EncryptionMaterials(keyPair);
            AmazonS3 s3EncryptionClient = AmazonS3EncryptionClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withEncryptionMaterials(new
StaticEncryptionMaterialsProvider(encryptionMaterials))
                    .withRegion(clientRegion)
                    .build();

            // Create a new object.
            byte[] plaintext = "S3 Object Encrypted Using Client-Side Asymmetric Master
Key.".getBytes();
            S3Object object = new S3Object();
            object.setKey(objectKeyName);
            object.setObjectContent(new ByteArrayInputStream(plaintext));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentLength(plaintext.length);

            // Upload the object. The encryption client automatically encrypts it.
            PutObjectRequest putRequest = new PutObjectRequest(bucketName,
                    object.getKey(),
                    object.getObjectContent(),
                    metadata);
            s3EncryptionClient.putObject(putRequest);

            // Download and decrypt the object.
            S3Object downloadedObject = s3EncryptionClient.getObject(bucketName,
object.getKey());
            byte[] decrypted = IOUtils.toByteArray(downloadedObject.getObjectContent());

            // Verify that the data that you downloaded is the same as the original data.
            System.out.println("Plaintext: " + new String(plaintext));
            System.out.println("Decrypted text: " + new String(decrypted));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void saveKeyPair(String dir,
                                    String publicKeyName,
                                    String privateKeyName,
                                    KeyPair keyPair) throws IOException {
        PrivateKey privateKey = keyPair.getPrivate();
        PublicKey publicKey = keyPair.getPublic();

        // Write the public key to the specified file.
```

```
        X509EncodedKeySpec x509EncodedKeySpec = new
 X509EncodedKeySpec(publicKey.getEncoded());
        FileOutputStream publicKeyOutputStream = new FileOutputStream(dir + File.separator
 + publicKeyName);
        publicKeyOutputStream.write(x509EncodedKeySpec.getEncoded());
        publicKeyOutputStream.close();

        // Write the private key to the specified file.
        PKCS8EncodedKeySpec pkcs8EncodedKeySpec = new
 PKCS8EncodedKeySpec(privateKey.getEncoded());
        FileOutputStream privateKeyOutputStream = new FileOutputStream(dir + File.separator
 + privateKeyName);
        privateKeyOutputStream.write(pkcs8EncodedKeySpec.getEncoded());
        privateKeyOutputStream.close();
    }

    private static KeyPair loadKeyPair(String dir,
                                       String publicKeyName,
                                       String privateKeyName,
                                       String algorithm)
            throws IOException, NoSuchAlgorithmException, InvalidKeySpecException {
        // Read the public key from the specified file.
        File publicKeyFile = new File(dir + File.separator + publicKeyName);
        FileInputStream publicKeyInputStream = new FileInputStream(publicKeyFile);
        byte[] encodedPublicKey = new byte[(int) publicKeyFile.length()];
        publicKeyInputStream.read(encodedPublicKey);
        publicKeyInputStream.close();

        // Read the private key from the specified file.
        File privateKeyFile = new File(dir + File.separator + privateKeyName);
        FileInputStream privateKeyInputStream = new FileInputStream(privateKeyFile);
        byte[] encodedPrivateKey = new byte[(int) privateKeyFile.length()];
        privateKeyInputStream.read(encodedPrivateKey);
        privateKeyInputStream.close();

        // Convert the keys into a key pair.
        KeyFactory keyFactory = KeyFactory.getInstance(algorithm);
        X509EncodedKeySpec publicKeySpec = new X509EncodedKeySpec(encodedPublicKey);
        PublicKey publicKey = keyFactory.generatePublic(publicKeySpec);

        PKCS8EncodedKeySpec privateKeySpec = new PKCS8EncodedKeySpec(encodedPrivateKey);
        PrivateKey privateKey = keyFactory.generatePrivate(privateKeySpec);

        return new KeyPair(publicKey, privateKey);
    }
}
```

Identity and Access Management in Amazon S3

Standardmäßig sind alle Amazon S3-Ressourcen – Buckets, Objekte und zugehörige Unterressourcen (z. B. `lifecycle`-Konfiguration und `website`-Konfiguration) – privat, sodass nur der Ressourceneigentümer mit einem AWS-Konto, über die sie erstellt wurden, auf die Ressourcen zugreifen kann. Der Ressourceneigentümer kann optional anderen Zugriffsberechtigungen gewähren, indem er eine Zugriffsrichtlinie schreibt.

Amazon S3 unterstützt Optionen für Zugriffsrichtlinien, die ganz allgemein als ressourcenbasierte Richtlinien und Benutzerrichtlinien unterteilt werden können. Zugriffsrichtlinien, die Sie Ihren Ressourcen hinzufügen (Buckets und Ordnern) werden als ressourcenbasierte Richtlinien bezeichnet. Beispielsweise sind Bucket-Richtlinien und Zugriffskontrolllisten (ACLs) ressourcenbasierte Richtlinien. Sie können auch Benutzern in Ihrem Konto Richtlinien hinzufügen. Diese werden als Benutzerrichtlinien bezeichnet. Sie

können ressourcenbasierte Richtlinien, Benutzerrichtlinien oder eine Kombination daraus verwenden, um Ihre Berechtigungen für Ihre Amazon S3-Ressourcen zu verwalten. Die einführenden Themen bieten allgemeine Orientierungshilfe für die Verwaltung von Berechtigungen.

Einführung in das Verwalten des Zugriffs auf Amazon S3-Ressourcen

Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die Optionen zum Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen erläutert werden:

- Übersicht über die Verwaltung von Zugriffsberechtigungen (p. 330)
- Wie Amazon S3 eine Anforderung autorisiert (p. 336)
- Orientierungshilfen für die Verwendung der unterstützten Zugriffsrichtlinienoptionen (p. 344)
- Beispielhafte Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen (p. 347)

Mehrere bewährte Sicherheitsmethoden umfassen ebenfalls die Zugriffskontrolle, z. B.:

- Vergewissern Sie sich, dass Amazon S3-Buckets nicht öffentlich zugänglich sind.
- Implementieren der geringstmöglichen Zugriffsrechte
- Verwenden Sie IAM-Rollen
- Aktivieren der Multifaktor-Authentifizierung (MFA)-Löschfunktion
- Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets
- Überwachen von AWS-Sicherheitshinweisen

Amazon S3-Ressourcenzugriffsoptionen

Nachdem Sie einleitende Themen zum Verwalten des Zugriffs auf Amazon S3-Ressourcen überprüft haben, können Sie dann die folgenden Themen verwenden, um weitere Informationen zu bestimmten Zugriffsrichtlinienoptionen zu erhalten:

- Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375)
- Zugriffsverwaltung mit ACLs (p. 479)
- Verwenden von Amazon S3 Block Public Access (p. 493)

Übersicht über die Verwaltung von Zugriffsberechtigungen

Beim Erteilen von Berechtigungen entscheiden Sie, wer die Berechtigungen erhält, für welche Amazon S3-Ressourcen die Berechtigungen gelten und welche Aktionen an diesen Ressourcen gestattet werden sollen.

Themen

- Amazon S3-Ressourcen: Buckets und Objekte (p. 331)
- Eigentümerschaft an Amazon S3 Buckets und Objekten (p. 331)
- Ressourcenoperationen (p. 332)
- Verwalten des Zugriffs auf Ressourcen (p. 332)
- Welche Zugriffskontrollmethode sollte ich verwenden? (p. 336)
- Weitere Infos (p. 336)

Amazon S3-Ressourcen: Buckets und Objekte

In Amazon Web Services (AWS) ist eine Ressource eine Entität, mit der Sie arbeiten können. In Amazon S3 sind Buckets und Objekte die Ressourcen und beiden sind Subressourcen zugeordnet. Nachfolgend sind einige der Bucket-Subressourcen beschrieben:

- `lifecycle` – speichert Lebenszyklus-Konfigurationsinformationen (siehe Verwaltung des Objektlebenszyklus (p. 139)).
- `website` – speichert Website-Konfigurationsinformationen, wenn Sie Ihren Bucket für das Website-Hosting konfigurieren (siehe Hosten einer statischen Website auf Amazon S3 (p. 602)).
- `versioning` – speichert die Versioning-Konfiguration (siehe PUT Bucket-Versioning).
- `policy` und `acl` (Access Control List) – speichern Zugriffsberechtigungsinformationen für den Bucket.
- `cors` (Cross-Origin Resource Sharing) – unterstützt die Konfiguration Ihres Buckets so, dass ursprungsübergreifende Anforderungen möglich sind (siehe Cross-Origin Resource Sharing (CORS) (p. 174)).
- `logging` – ermöglicht es Ihnen, Amazon S3 aufzufordern, Bucket-Zugriffsprotokolle zu speichern.

Zu den Objekt-Subressourcen gehören unter anderem:

- `acl` – speichert eine Liste von Zugriffsberechtigungen für das Objekt. Dieses Thema beschreibt, wie diese Subressource verwendet wird, um Objektberechtigungen zu verwalten (siehe Zugriffsverwaltung mit ACLs (p. 479)).
- `restore` – unterstützt die temporäre Wiederherstellung eines archivierten Objekts (siehe POST Object restore). Ein Objekt in der Speicherklasse S3 Glacier ist ein archiviertes Objekt. Für den Zugriff auf das Objekt müssen Sie zuerst eine Anforderung zur Wiederherstellung initiieren, die eine Kopie des archivierten Objekts wiederherstellt. Geben Sie in der Anforderung die Anzahl der Tage an, wie lang die wiederhergestellte Kopie existieren soll. Weitere Informationen über das Archivieren von Objekten finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Eigentümerschaft an Amazon S3 Buckets und Objekten

Buckets und Objekte sind Amazon S3-Ressourcen. Standardmäßig kann nur der jeweilige Eigentümer auf diese Ressourcen zugreifen. Der Ressourceneigentümer bezieht sich auf das AWS-Konto, das die Ressource erstellt. Beispiel:

- Das AWS-Konto, mit dem Sie Buckets und Objekte erstellen und hochladen, ist Eigentümer dieser Ressourcen.

- Wenn Sie mit den Anmeldeinformationen eines AWS Identity and Access Management(IAM)-Benutzers ein Objekt hochladen, ist das AWS-Konto, zu dem der Benutzer oder die Rolle gehört, Eigentümer des Objekts.

- Ein Bucket-Eigentümer kann einem anderen AWS-Konto (oder Benutzern in einem anderen Konto) kontenübergreifende Berechtigungen erteilen, um Objekte hochzuladen. In diesem Fall gehören dem AWS-Konto, das diese Objekte hochlädt, diese Objekte. Der Bucket-Eigentümer besitzt keine Berechtigungen für die Objekte, die anderen Konten gehören, mit den folgenden Ausnahmen:
  - Der Bucket-Eigentümer zahlt die Rechnungen. Ein Bucket-Eigentümer kann explizit den Zugriff auf Objekte verweigern oder Objekte im Bucket löschen, unabhängig davon, wem sie gehören.
  - Ein Bucket-Eigentümer kann Objekte archivieren oder archivierte Objekte wiederherstellen, unabhängig davon, wem sie gehören. Die Archivierung bezieht sich auf die Speicherklasse, die beim

Speichern der Objekte verwendet wurde. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Eigentümerschaft und Anforderungsauthentifizierung

Alle Anforderungen nach einem Bucket sind authentifiziert oder nicht authentifiziert. Authentifizierte Anforderungen müssen einen Signaturwert enthalten, der den Absender der Anforderung authentifiziert, für nicht authentifizierte Anforderungen gilt dies nicht. Weitere Informationen zur Anforderungsauthentifizierung finden Sie unter Senden von Anforderungen (p. 11).

Der Eigentümer eines Buckets kann nicht authentifizierte Anforderungen zulassen. So sind beispielsweise nicht authentifizierte PUT Object-Anforderungen erlaubt, wenn für ein Bucket eine öffentliche Bucketrichtlinie gilt, oder wenn ein Bucket-ACL `WRITE`- oder `FULL_CONTROL`-Zugriff für die Gruppe „All Users (Alle Benutzer) oder für anonyme Benutzer gewährt. Weitere Informationen zu öffentlichen Bucketrichtlinien und öffentlichen ACLs finden Sie unter Die Bedeutung von „öffentlich“ (p. 496).

Alle nicht authentifizierten Anforderungen werden vom anonymen Benutzer erstellt. Dieser Benutzer ist in Zugriffssteuerungslisten (ACL) durch die spezifische kanonische Benutzer-ID `65a011a29cdf8ec533ec3d1ccaae921c` repräsentiert. Wenn ein Objekt mit einer nicht authentifizierten Anforderung zu einem Bucket hochgeladen wird, ist der anonyme Benutzer Eigentümer des Objekts. Die Standard-Objekt-ACL gewährt dem anonymen Benutzer als dem Eigentümer des Objekts `FULL_CONTROL`. Daher erlaubt Amazon S3, dass nicht authentifizierte Anforderungen das Objekt abrufen oder seine ACL modifizieren.

Um zu verhindern, dass Objekte von dem anonymen Benutzer modifiziert werden, empfehlen wir, keine Bucketrichtlinien zu implementieren, die anonyme öffentliche Schreibvorgänge für Ihren Bucket erlauben, oder die ACLs verwenden, den dem anonymen Benutzer Schreibzugriff auf Ihren Bucket gewähren. Sie können diese empfohlene Verhaltensweise durch die Verwendung von Amazon S3 Block Public Access erzwingen.

Weitere Informationen zum Blockieren des öffentlichen Zugriffs finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493). Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht (p. 480).

> **Important**
>
> AWS rät davon ab, die Root-Anmeldeinformationen Ihres AWS-Kontos für authentifizierte Anforderungen zu verwenden. Erstellen Sie stattdessen einen IAM-Benutzer, dem Sie vollständigen Zugriff gewähren. Wir bezeichnen diese Benutzer als Administratorbenutzer. Anstelle der Root-Benutzer-Anmeldeinformationen des AWS-Kontos können Sie die des Administratorbenutzers für Aufgaben in AWS verwenden, z. B. um einen Bucket und Benutzer zu erstellen sowie diesen Berechtigungen zu gewähren. Weitere Informationen finden Sie unter AWS-Konto-Root-Benutzer-Anmeldeinformationen vs. IAM-Benutzeranmeldeinformationen in der AWS General Reference sowie unter Bewährte Methoden für IAM im IAM-Benutzerhandbuch.

Ressourcenoperationen

Amazon S3 bietet zum Arbeiten eine Reihe von Operationen mit den Amazon S3-Ressourcen. Eine Liste der verfügbaren Operationen finden Sie unter Operationen für Buckets und Operationen für Objekte in der Amazon Simple Storage Service API Reference.

Verwalten des Zugriffs auf Ressourcen

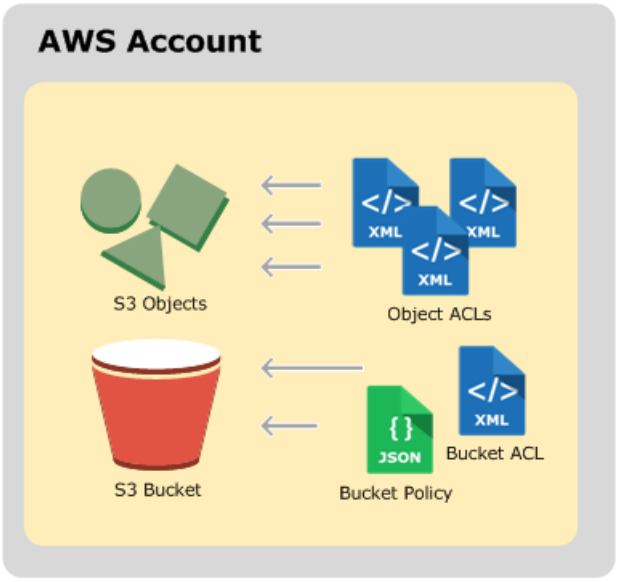
Die Zugriffsverwaltung bezieht sich darauf, anderen (AWS-Konten und Benutzern) durch Schreiben einer Zugriffsrichtlinie Berechtigungen für die Durchführung von Ressourcenoperationen zu gewähren. Sie können beispielsweise die `PUT Object`-Berechtigung einem Benutzer in einem AWS-Konto erteilen, sodass der Benutzer Objekte in Ihren Bucket hochladen kann. Neben der Erteilung von Berechtigungen für einzelne Benutzer und Konten können Sie jedem Berechtigungen erteilen (auch als anonymer Zugriff

bezeichnet), oder allen authentifizierten Benutzern (Benutzern mit AWS-Zugriffsrechten). Wenn Sie beispielsweise Ihren Bucket als Website konfigurieren, könnten Sie Objekt öffentlich machen, indem Sie jedem die `GET Object`-Berechtigung erteilen.

Zugriffsrichtlinienoptionen

Eine Zugriffsrichtlinie beschreibt, wer Zugriff auf welche Objekte hat. Sie können eine Zugriffsrichtlinie einer Ressource (Bucket und Objekt) oder einem Benutzer zuordnen. Dementsprechend können Sie die verfügbaren Amazon S3-Zugriffsrichtlinien wie folgt kategorisieren:

- Auf Ressourcen basierende Richtlinien – Bucket-Richtlinien und Zugriffskontrolllisten (ACLs) sind ressourcenbasiert, weil Sie sie Ihren Amazon S3-Ressourcen zuordnen.



- ACL – Jedem Bucket und Objekt ist eine Zugriffskontrollliste zugeordnet. Eine ACL listet die erteilten Berechtigungen auf, die den Berechtigungsempfänger und die erteilte Berechtigung identifizieren. Sie können ACLs verwenden, um grundlegende Lese-/Schreibberechtigungen für andere AWS-Konten zu erteilen. ACLs verwenden ein Amazon S3-spezifisches XML-Schema.

  Es folgt ein Beispiel für eine Bucket-ACL. Das Recht der ACL zeigt einen Bucket-Eigentümer, der volle Kontrolle besitzt.

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>*** Owner-Canonical-User-ID ***</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>*** Owner-Canonical-User-ID ***</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

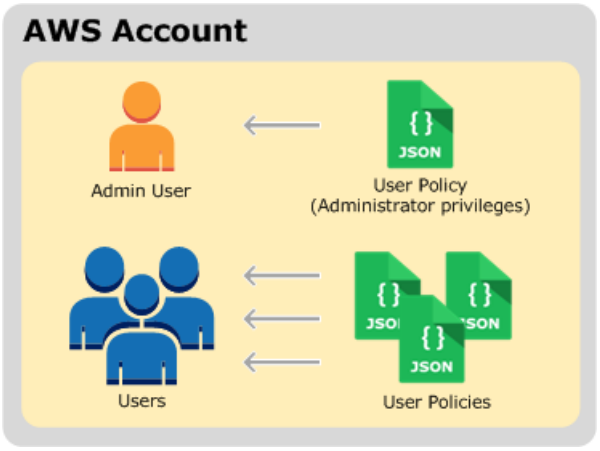
  Bucket- und Objekt-ACLs verwenden dasselbe XML-Schema.

- Bucketrichtlinie – Sie können Ihrem Bucket eine Bucketrichtlinie hinzufügen, um anderen AWS-Konten oder IAM-Benutzern die Berechtigungen für den Bucket und die darin enthaltenen Objekte zu erteilen. Objektberechtigungen gelten nur für die Objekte, die der Bucket-Eigentümer erstellt. Bucket-Richtlinien ergänzen ACL-basierte Zugriffsrichtlinien, und in vielen Fällen ersetzen sie sie.

  Hier finden Sie ein Beispiel für eine Bucket-Richtlinie. Bucket-Richtlinien (und Benutzerrichtlinien) werden unter Verwendung einer JSON-Datei dargestellt. Die Richtlinie gewährt anonyme Berechtigung zum Lesen aller Objekte in einem Bucket. Die Bucket-Richtlinie enthält eine Anweisung, die die `s3:GetObject`-Aktion (Leseberechtigung) für Buckets im Bucket `examplebucket` erlaubt. Durch die Angabe von `principal` mit einem Platzhalterzeichen (*) erteilt die Richtlinie anonymen Zugriff und ist mit Vorsicht zu verwenden. Die folgende Bucket-Richtlinie würde beispielsweise Objekte öffentlich zugänglich machen.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect":"Allow",
            "Principal": "*",
            "Action":["s3:GetObject"],
            "Resource":["arn:aws:s3:::examplebucket/*"]
        }
    ]
}
```

- Benutzerrichtlinien – Sie können IAM für die Verwaltung des Zugriffs auf Ihre Amazon S3-Ressourcen verwenden. Sie können IAM-Benutzer, Gruppen und Rollen in Ihrem Konto erstellen und ihnen Zugriffsrichtlinien zuordnen, die ihnen Zugriff auf die AWS-Ressourcen erteilen, einschließlich Amazon S3.



Weitere Informationen zu IAM finden Sie auf der Produktdetailseite zum AWS Identity and Access Management (IAM).

Nachstehend finden Sie ein Beispiel für eine Benutzerrichtlinie. In einer IAM-Benutzerrichtlinie können Sie keine anonymen Berechtigungen gewähren, weil die Richtlinie mit einem Benutzer verknüpft ist. Die Beispielrichtlinie gestattet dem entsprechenden Benutzer, dass er sechs verschiedene Amazon S3-Aktionen für einen Bucket und die darin enthaltenen Objekte ausführt. Sie können diese Richtlinie einem spezifischen IAM-Benutzer, einer Gruppe oder einer Rolle zuordnen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "ExampleStatement1",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                 "arn:aws:s3:::examplebucket/*",
                 "arn:aws:s3:::examplebucket"
            ]
        },
        {
            "Sid": "ExampleStatement2",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        }
    ]
}
```

Wenn Amazon S3 eine Anforderung erhält, muss es alle Zugriffsrichtlinien auswerten, um festzustellen, ob es die Anforderung genehmigen oder verweigern soll. Weitere Informationen dazu, wie Amazon S3 diese Richtlinien auswertet, finden Sie unter Wie Amazon S3 eine Anforderung autorisiert (p. 336).

Access Analyzer for S3

Auf der Amazon S3-Konsole können Sie mit Access Analyzer for S3 alle Buckets überprüfen, die über Bucket-Zugriffskontrolllisten (ACLs), Bucket-Richtlinien oder Zugriffspunkt-Richtlinien verfügen, die öffentlichen oder freigegebenen Zugriff gewähren. Access Analyzer for S3 macht Sie auf Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen AWS-Konten, einschließlich AWS-Konten außerhalb Ihrer Organisation, Zugriff gewährt wird. Für jeden öffentlichen oder freigegebenen Bucket erhalten Sie Ergebnisse, die die Quelle und die Ebene des öffentlichen oder freigegebenen Zugriffs melden.

In Access Analyzer for S3 können Sie den gesamten öffentlichen Zugriff auf einen Bucket mit einem einzigen Klick blockieren. Wir empfehlen Ihnen, den gesamten Zugriff auf Ihre Buckets zu blockieren, es sei denn, Sie benötigen öffentlichen Zugriff, um einen bestimmten Anwendungsfall zu unterstützen. Bevor Sie den gesamten öffentlichen Zugriff blockieren, stellen Sie sicher, dass Ihre Anwendungen ohne öffentlichen Zugriff weiterhin ordnungsgemäß funktionieren. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493).

Sie können auch einen Drilldown in die Berechtigungseinstellungen auf Bucket-Ebene ausführen, um detaillierte Zugriffsebenen zu konfigurieren. Für bestimmte und geprüfte Anwendungsfälle, die öffentlichen oder freigegebenen Zugriff erfordern, können Sie Ihre Absicht bestätigen und aufzeichnen, dass der Bucket öffentlich oder freigegeben bleibt, indem Sie die Ergebnisse für den Bucket archivieren. Diese Bucket-Konfigurationen sind jederzeit aufrufbar und änderbar. Sie können Ihre Ergebnisse auch als CSV-Bericht zu Auditing-Zwecken herunterladen.

Access Analyzer for S3 ist kostenlos in der Amazon S3-Konsole verfügbar. Access Analyzer for S3 wird von AWS Identity and Access Management (IAM) Access Analyzer betrieben. Um Access Analyzer for S3 auf der Amazon S3-Konsole verwenden zu können, müssen Sie die IAM-Konsole aufrufen und einen Analysator auf Kontenebene in IAM Access Analyzer auf regionaler Basis erstellen.

Weitere Informationen zu Access Analyzer for S3 finden Sie unter Verwenden von Access Analyzer for S3 im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Welche Zugriffskontrollmethode sollte ich verwenden?

Angesichts der verschiedenen Optionen zum Schreiben einer Zugriffsrichtlinie ergeben sich die folgenden Fragen:

- Wann sollte ich welche Zugriffskontrollmethode verwenden? Sollte ich beispielsweise für das Erteilen von Bucket-Berechtigungen eine Bucket-Richtlinie oder eine Bucket-ACL verwenden? Ich besitze einen Bucket und die im Bucket enthaltenen Objekte. Sollte ich eine auf Ressourcen basierende Zugriffsrichtlinie oder eine IAM-Benutzerrichtlinie verwenden? Wenn ich eine auf Ressourcen basierende Zugriffsrichtlinie verwende, sollte ich eine Bucket-Richtlinie oder eine Objekt-ACL verwenden, um Objektberechtigungen zu verwalten?
- Ich besitze einen Bucket, aber nicht alle darin enthaltenen Objekte. Wie werden Zugriffsberechtigungen für die Objekte verwaltet, die jemandem anderen gehören?
- Wenn ich den Zugriff über eine Kombination aus diesen Zugriffsrichtlinienoptionen erteile, wie kann Amazon S3 feststellen, ob ein Benutzer berechtigt ist, eine angefragte Operation auszuführen?

Die folgenden Abschnitte erklären diese Zugriffskontrollalternativen, wie Amazon S3 einen Zugriffskontrollmechanismus auswertet und wann welche Zugriffskontrollmethode zu verwenden ist. Sie bieten auch Beispiel-Anleitungen.

- Wie Amazon S3 eine Anforderung autorisiert (p. 336)
- Orientierungshilfen für die Verwendung der unterstützten Zugriffsrichtlinienoptionen (p. 344)
- Beispielhafte Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen (p. 347)

Weitere Infos

Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die Grundkonzepte und verfügbaren Optionen zum Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen erläutert werden. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3 (p. 329). Die folgenden Themen enthalten weitere Informationen über spezifische Zugriffsrichtlinienoptionen.

- Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375)
- Zugriffsverwaltung mit ACLs (p. 479)

Wie Amazon S3 eine Anforderung autorisiert

Themen

Wenn Amazon S3 eine Anforderung erhält – z. B. ein Bucket oder eine Objektoperation –, überprüft es zuerst, ob der Auftraggeber die erforderlichen Berechtigungen besitzt. Amazon S3 wertet alle relevanten Zugriffsrichtlinien, Benutzerrichtlinien und ressourcenbasierten Richtlinien (Bucket-Richtlinie, Bucket-ACL, Objekt-ACL) aus, um entscheiden zu können, ob die Anforderung autorisiert werden soll. Es folgen typische Beispielszenarien:

- Wenn der Anforderer ein IAM-Prinzipal ist, muss Amazon S3 bestimmen, ob das übergeordnete AWS-Konto, zu dem der Prinzipal gehört, dem Prinzipal die erforderliche Berechtigung zum Ausführen der Operation erteilt hat. Erfolgt die Anforderung darüber hinaus für eine Bucket-Operation, wie beispielsweise eine Anforderung, den Bucket-Inhalt aufzulisten, muss Amazon S3 prüfen, ob der Bucket-Eigentümer dem Auftraggeber die Berechtigung erteilt hat, die Operation auszuführen.

Note

Um eine bestimmte Operation für eine Ressource auszuführen, muss einem IAM-Benutzer sowohl die Berechtigung von dem übergeordneten AWS-Konto, zu dem er gehört, als auch von dem AWS-Konto, dem die Ressource gehört, erteilt werden.

- Erfolgt die Anforderung für eine Operation für ein Objekt, das nicht dem Bucket-Eigentümer gehört, muss Amazon S3 sicherstellen, dass der Auftraggeber die Berechtigungen von dem Objekteigentümer hat, und außerdem die Bucket-Richtlinie prüfen, um sicherzustellen, dass der Bucket-Eigentümer keine explizite Zugriffsverweigerung für das Objekt festgelegt hat.

  Note

  Ein Bucket-Eigentümer (der die Rechnung zahlt) kann explizit den Zugriff auf Objekte im Bucket verweigern, unabhängig davon, wem dieser gehört. Der Bucket-Eigentümer kann auch ein beliebiges Objekt im Bucket löschen

Um zu ermitteln, ob der Auftraggeber die Berechtigung hat, die spezifische Operation auszuführen, geht Amazon S3 wie folgt vor, wenn es eine Anforderung erhält:

1. Es wandelt alle relevanten Zugriffsrichtlinien (Benutzerrichtlinie, Bucket-Richtlinie, ACLs) zur Laufzeit in eine Richtlinienmenge zur Auswertung um.
2. Es wertet in den folgenden Schritten die resultierende Richtlinienmenge aus. Amazon S3 wertet in jedem Schritt eine Untermenge der Richtlinien in einem spezifischen Kontext aus, basierend auf der Kontextautorität.
   a. Benutzerkontext – Im Benutzerkontext ist das übergeordnete Konto, zu dem der Benutzer gehört, die Kontextautorität.

      Amazon S3 wertet eine Untermenge der Richtlinien aus, die dem übergeordneten Konto gehören. Diese Untermenge beinhaltet die Benutzerrichtlinie, die das übergeordnete Konto dem Benutzer zuordnet. Wenn dem übergeordneten Konto auch die Ressource in der Anforderung gehört (Bucket, Objekt), wertet Amazon S3 gleichzeitig auch die entsprechenden Ressourcenrichtlinien aus (Bucket-Richtlinie, Bucket-ACL und Objekt-ACL).

      Ein Benutzer benötigt die Berechtigung von dem übergeordneten Konto, um die Operation auszuführen.

      Dieser Schritt wird nur angewendet, wenn die Anforderung von einem Benutzer in einem AWS-Konto gestellt wurde. Wenn die Anforderung unter Verwendung von Root-Anmeldeinformationen eines AWS-Kontos erfolgt, überspringt Amazon S3 diesen Schritt.

   b. Bucket-Kontext – Im Bucket-Kontext wertet Amazon S3 Richtlinien aus, die dem AWS-Konto gehören, dem der Bucket gehört.

      Erfolgt die Anforderung für eine Bucket-Operation, muss der Auftraggeber die Berechtigung vom Bucket-Eigentümer besitzen. Erfolgt die Anforderung für ein Objekt, wertet Amazon S3 alle Richtlinien aus, die dem Bucket-Eigentümer gehören, um zu überprüfen, ob der Bucket-Eigentümer für das Objekt eine explizite Zugriffsverweigerung festgelegt hat. Wurde eine explizite Zugriffsverweigerung festgelegt, autorisiert Amazon S3 die Anforderung nicht.

   c. Objektkontext – Erfolgt die Anforderung für ein Objekt, wertet Amazon S3 die Untermenge der Richtlinien aus, die dem Objekteigentümer gehören.

In den folgenden Abschnitten wird dies anhand von Beispielen ausführlich beschrieben:

- Wie Amazon S3 eine Anforderung für eine Bucket-Operation autorisiert (p. 338)
- Wie Amazon S3 eine Anforderung für eine Objekt-Operation autorisiert (p. 341)

Verwandte Themen

Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die Optionen zum Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen erläutert werden. Weitere Informationen finden Sie unter Einführung in das Verwalten des Zugriffs auf Amazon S3-Ressourcen (p. 330). Die folgenden Themen enthalten weitere Informationen über spezifische Zugriffsrichtlinienoptionen.

- Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375)
- Zugriffsverwaltung mit ACLs (p. 479)

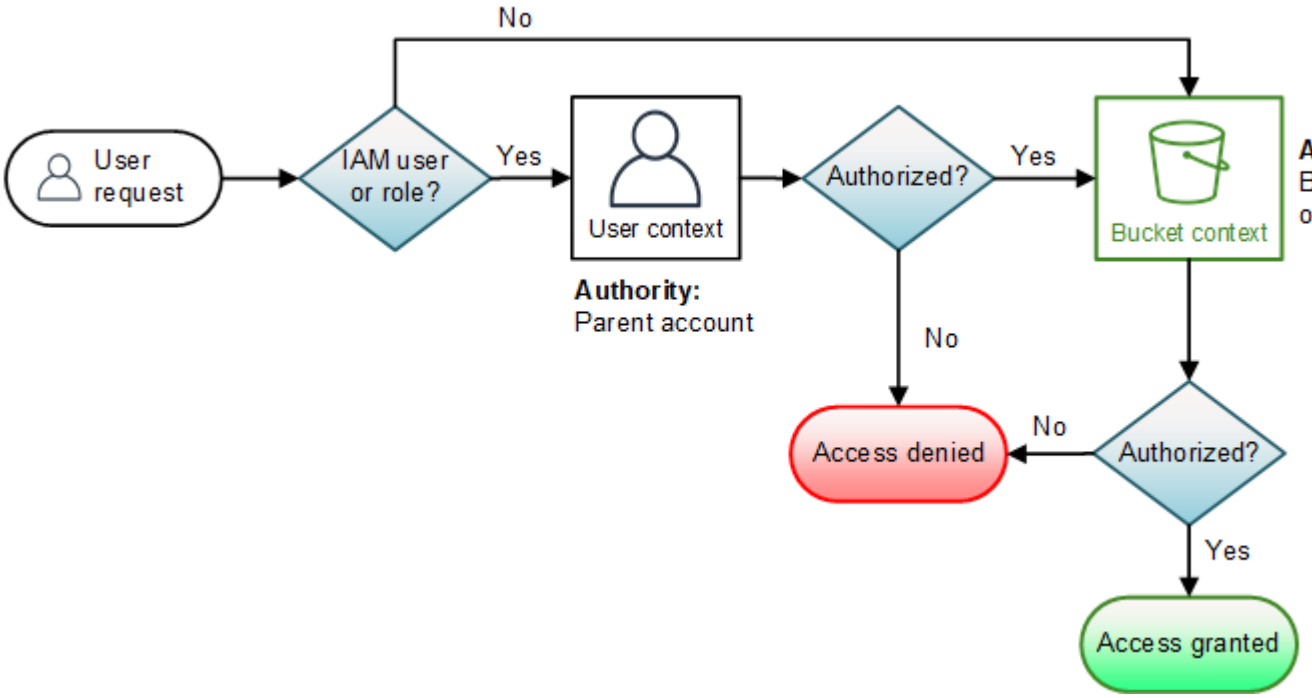
Wie Amazon S3 eine Anforderung für eine Bucket-Operation autorisiert

Wenn Amazon S3 eine Anforderung für eine Bucket-Operation erhält, wandelt Amazon S3 alle relevanten Berechtigungen – ressourcenbasierte -Berechtigungen (Bucket-Richtlinie, Bucket-Zugriffskontrollliste (ACL)) und IAM-Benutzerrichtlinie, falls die Anforderung von einem IAM-Prinzipal stammt – in eine Reihe von Richtlinien um, die es zur Laufzeit auswertet. Anschließend wertet es die resultierende Richtlinienmenge in mehreren Schritten in Übereinstimmung mit einem spezifischen Kontext aus – Benutzerkontext oder Bucket-Kontext.

1. Benutzerkontext – Wenn der Anforderer ein IAM-Prinzipal ist, muss der Prinzipal die Berechtigung von dem übergeordneten AWS-Konto erhalten, zu dem er gehört. In diesem Schritt wertet Amazon S3 eine Untermenge der Richtlinien aus, die dem übergeordneten Konto gehören (auch als Kontextautorität bezeichnet). Diese Richtlinienuntermenge beinhaltet die Benutzerrichtlinie, die das übergeordnete Konto dem Prinzipal zuordnet. Wenn dem übergeordneten Konto auch die Ressource in der Anforderung gehört (in diesem Fall der Bucket), wertet Amazon S3 gleichzeitig auch die entsprechenden Ressourcenrichtlinien aus (Bucket-Richtlinie und Bucket-ACL). Immer wenn eine Anforderung für eine Bucket-Operation gemacht wird, zeichnen die Server-Zugriffsprotokolle die kanonische Benutzer-ID des Anforderers auf. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).
2. Bucket-Kontext – Der Auftraggeber muss die Berechtigung vom Bucket-Eigentümer besitzen, eine spezifische Bucket-Operation auszuführen. In diesem Schritt wertet Amazon S3 eine Untermenge der Richtlinien aus, die dem AWS-Konto gehören, dem der Bucket gehört.

   Der Bucket-Eigentümer kann Berechtigungen unter Verwendung einer Bucket-Richtlinie oder Bucket-ACL erteilen. Wenn das AWS-Konto, das Eigentümer des Buckets ist, zudem auch das übergeordnete Konto eines IAM-Prinzipals ist, ist zu beachten, das mit ihm Bucket-Berechtigungen in einer Benutzerrichtlinie konfiguriert werden können.

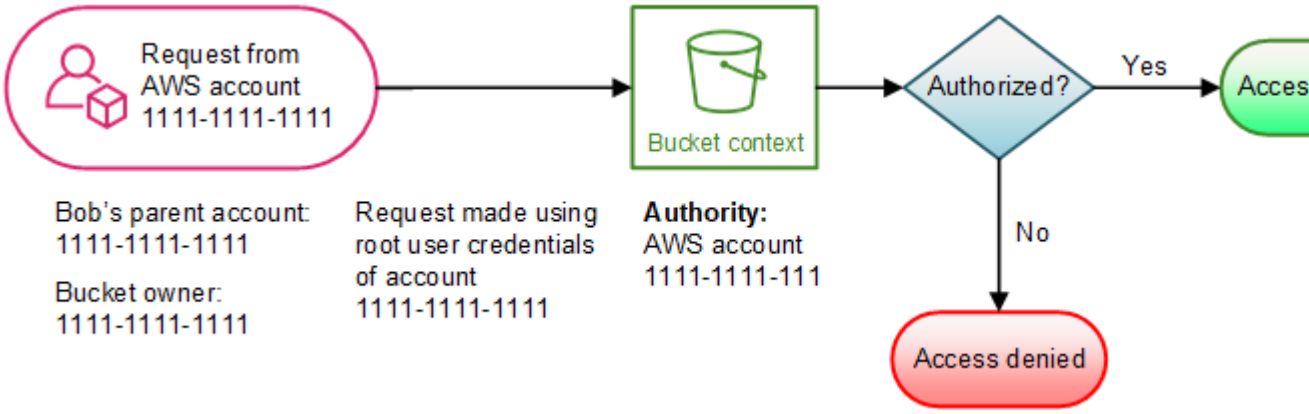
Nachfolgend sehen Sie eine grafische Darstellung der kontextbasierten Auswertung für Bucket-Operationen.



Die folgenden Beispiele veranschaulichen die Auswertungslogik.

Beispiel 1: Vom Bucket-Eigentümer angeforderte Bucket-Operation

In diesem Beispiel sendet der Bucket-Eigentümer eine Anforderung einer Bucket-Operation unter Verwendung der Root-Anmeldeinformationen des AWS-Kontos.

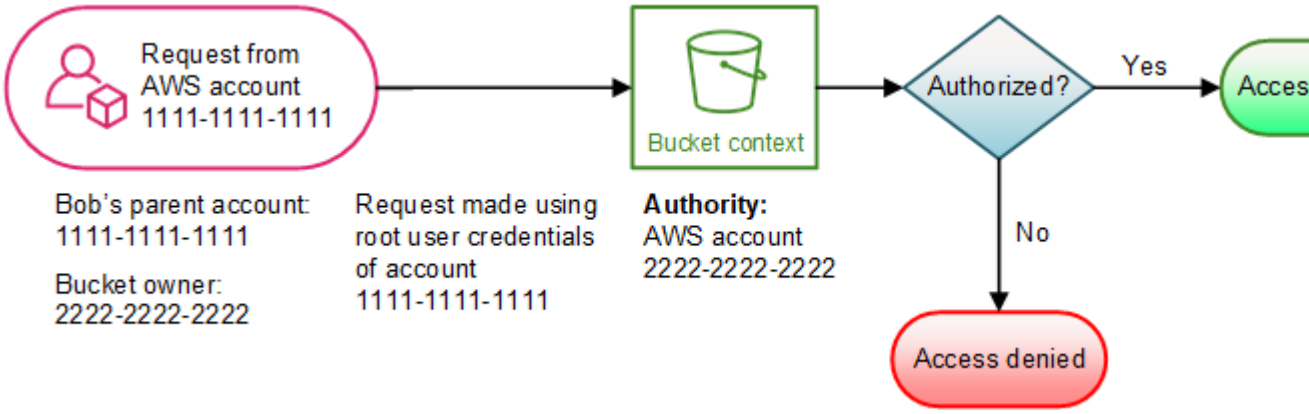


Amazon S3 führt die Kontextauswertung wie folgt durch:

1. Da die Anforderung unter Verwendung von Root-Anmeldeinformationen eines AWS-Kontos erfolgt, wird der Benutzerkontext nicht ausgewertet.
2. Im Bucket-Kontext überprüft Amazon S3 die Bucket-Richtlinie, um festzustellen, ob der Auftraggeber die Berechtigung besitzt, die Operation auszuführen. Amazon S3 autorisiert die Anforderung.

Beispiel 2: Von einem AWS-Konto, das nicht der Bucket-Eigentümer ist, angeforderte Bucket-Operation

In diesem Beispiel wird eine Anforderung unter Verwendung der Root-Anmeldeinformationen des AWS-Kontos 1111-1111-1111 für eine Bucket-Operation gestellt, die dem AWS-Konto 2222-2222-2222 gehört. An dieser Anforderung sind keine IAM-Benutzer beteiligt.

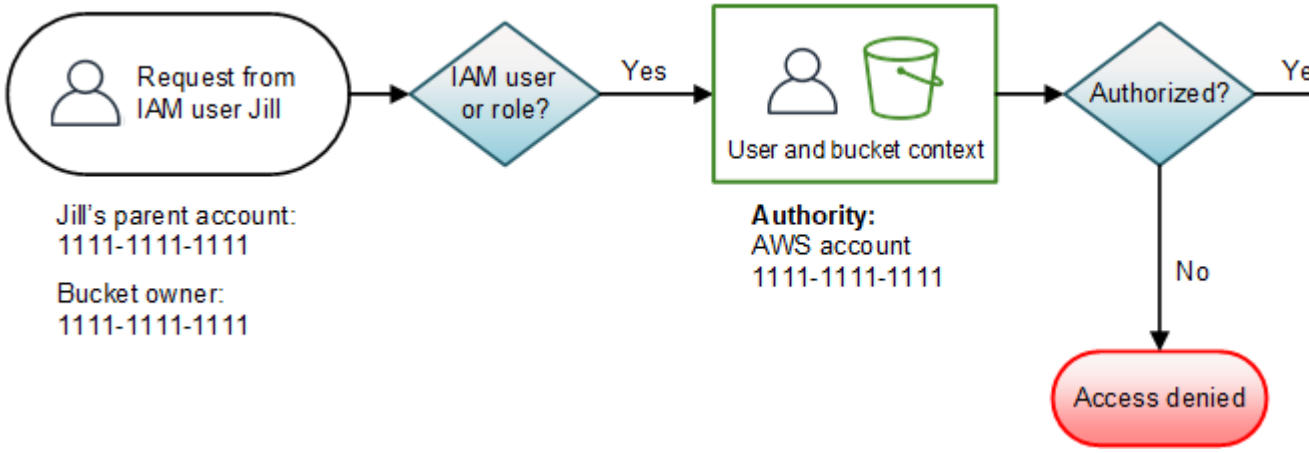


In diesem Fall wertet Amazon S3 den Kontext wie folgt aus:

1. Da die Anforderung unter Verwendung von Root-Anmeldeinformationen eines AWS-Kontos erfolgt, wird der Benutzerkontext nicht ausgewertet.
2. Im Bucket-Kontext wertet Amazon S3 die Bucket-Richtlinie aus. Wenn der Bucket-Eigentümer (AWS account 2222-2222-2222) das AWS-Konto 1111-1111-1111 nicht autorisiert hat, die angeforderte Operation auszuführen, verweigert Amazon S3 die Anforderung. Andernfalls genehmigt Amazon S3 die Anforderung und führt die Operation aus.

Beispiel 3: Von einem IAM-Prinzipal, dessen übergeordnetes AWS-Konto auch der Bucket-Eigentümer ist, angeforderte Bucket-Operation

In dem Beispiel wird die Anforderung von Jill gesendet, einer IAM-Benutzerin im AWS-Konto 1111-1111-1111, dem auch der Bucket gehört.



Amazon S3 führt die folgende Kontextauswertung durch:

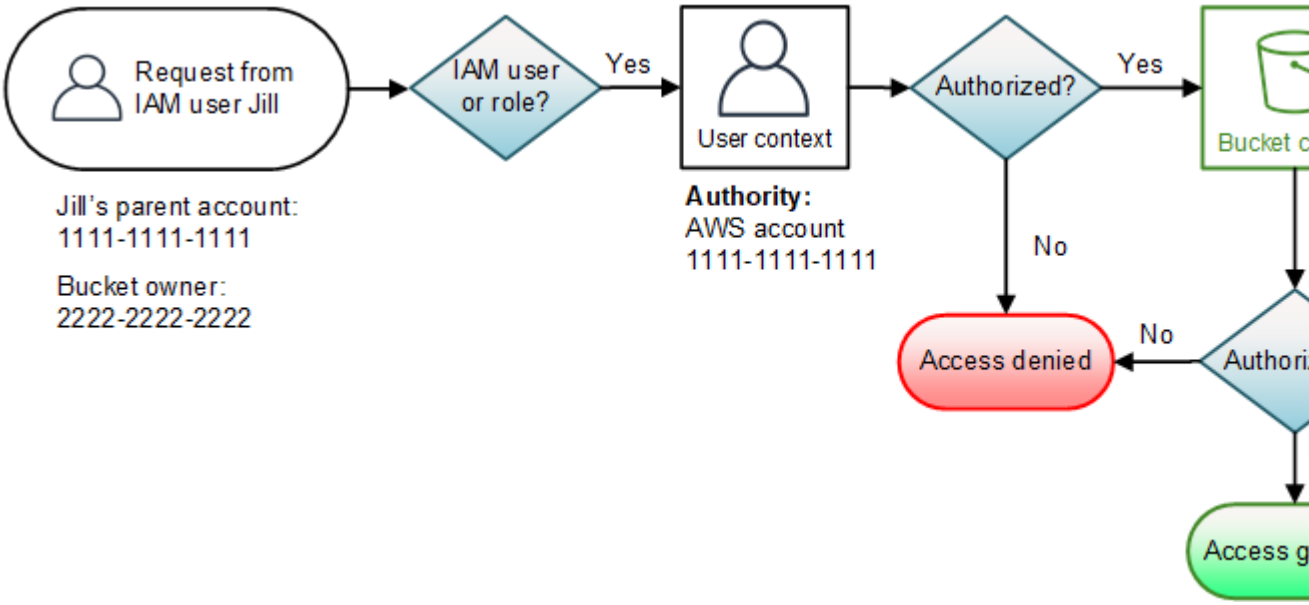
1. Da die Anfrage von einem IAM-Prinzipal stammt, wertet Amazon S3 im Benutzerkontext alle Richtlinien aus, die zu dem übergeordneten AWS-Konto gehören, um festzustellen, ob Jill zum Ausführen der Operation berechtigt ist.

In diesem Beispiel ist das übergeordnete AWS-Konto 1111-1111-1111, zu dem der Prinzipal gehört, auch gleichzeitig der Bucket-Eigentümer. Somit wertet Amazon S3 zusätzlich zu der Benutzerrichtlinie auch die Bucket-Richtlinie und Bucket-ACL im selben Kontext aus, weil sie zum selben Konto gehören.

2. Amazon S3 hat die Bucket-Richtlinie und die Bucket-ACL als Teil des Benutzerkontexts ausgewertet, deshalb wertet es den Bucket-Kontext nicht aus.

Beispiel 4: Von einem IAM-Prinzipal, dessen übergeordnetes AWS-Konto nicht der Bucket-Eigentümer ist, angeforderte Bucket-Operation

In diesem Beispiel wird die Anforderung von Jill gesendet, einer IAM-Benutzerin, deren übergeordnetes AWS-Konto 1111-1111-1111 ist, aber der Bucket gehört einem anderen AWS-Konto, 2222-2222-2222.



Jill benötigt Berechtigungen sowohl vom übergeordneten AWS-Konto, als auch vom Bucket-Eigentümer. Amazon S3 wertet den Kontext wie folgt aus:

1. Da die Anforderung von einem IAM-Prinzipal stammt, wertet Amazon S3 den Benutzerkontext aus, indem es die Richtlinien überprüft, die vom Konto autorisiert wurden, um sicherzustellen, dass Jill über die erforderlichen Berechtigungen verfügt. Wenn Jill die Berechtigung besitzt, wertet Amazon S3 den Bucket-Kontext aus, andernfalls weist es die Anforderung ab.
2. Im Bucket-Kontext überprüft Amazon S3, ob der Bucket-Eigentümer 2222-2222-2222 Jill (oder ihrem übergeordneten AWS-Konto) die Berechtigung erteilt hat, die angeforderte Operation auszuführen. Wenn sie diese Berechtigung besitzt, genehmigt Amazon S3 die Anforderung und führt die Operation aus. Andernfalls lehnt Amazon S3 die Anforderung ab.

Wie Amazon S3 eine Anforderung für eine Objekt-Operation autorisiert

Wenn Amazon S3 eine Anforderung für eine Bucket-Operation erhält, wandelt es alle relevanten Berechtigungen – ressourcenbasierte Berechtigungen (Objekt-Zugriffskontrollliste (ACL), Bucket-Richtlinie, Bucket-ACL) und IAM-Benutzerrichtlinien – in eine Reihe von Richtlinien um, die zur Laufzeit ausgewertet werden sollen. Anschließend wertet es in mehreren Schritten die resultierende Richtlinienmenge aus. Es wertet in jedem Schritt eine Untermenge der Richtlinien in drei spezifischen Kontexten aus – Benutzerkontext, Bucket-Kontext und Objektkontext.

1. Benutzerkontext – Wenn der Anforderer ein IAM-Prinzipal ist, muss der Prinzipal die Berechtigung von dem übergeordneten AWS-Konto erhalten, zu dem er gehört. In diesem Schritt wertet Amazon S3 eine Untermenge der Richtlinien aus, die dem übergeordneten Konto gehören (auch als Kontextautorität bezeichnet). Diese Richtlinienuntermenge beinhaltet die Benutzerrichtlinie, die das übergeordnete Konto dem Prinzipal zuordnet. Wenn dem übergeordneten Konto auch die Ressource in der Anforderung gehört (Bucket, Objekt), wertet Amazon S3 gleichzeitig die entsprechenden Ressourcenrichtlinien aus (Bucket-Richtlinie, Bucket-ACL und Objekt-ACL).

   Note

   Wenn das übergeordnete AWS-Konto Eigentümer der Ressource (Bucket oder Objekt) ist, kann es seinem IAM-Prinzipal unter Verwendung der Benutzerrichtlinie oder der Ressourcenrichtlinie Ressourcenberechtigungen erteilen.

2. Bucket-Kontext – In diesem Kontext wertet Amazon S3 Richtlinien aus, die dem AWS-Konto gehören, dem der Bucket gehört.

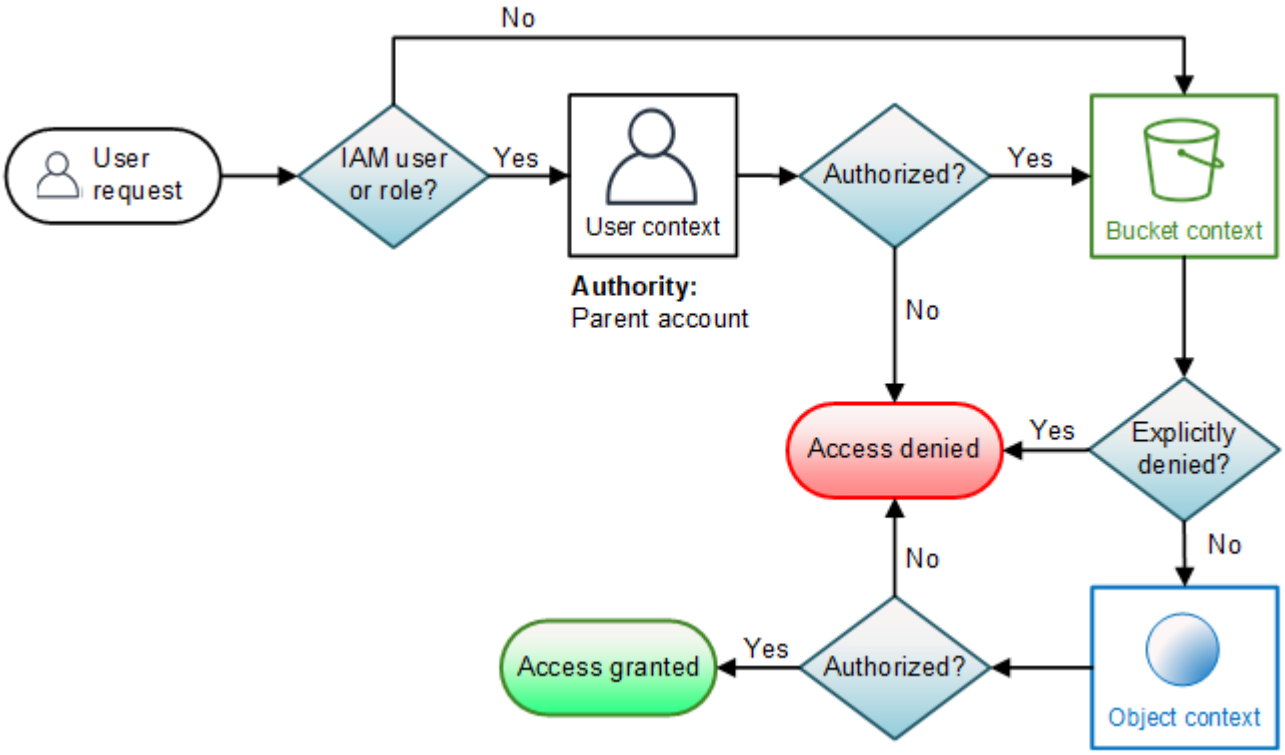
   Wenn das AWS-Konto, dem das Objekt in der Anforderung gehört, nicht gleich dem Bucket-Eigentümer ist, überprüft Amazon S3 im Bucket-Kontext die Richtlinien daraufhin, ob der Bucket-Eigentümer für das Objekt eine explizite Zugriffsverweigerung festgelegt hat. Wurde eine explizite Zugriffsverweigerung für das Objekt festgelegt, autorisiert Amazon S3 die Anforderung nicht.

3. Objekt-Kontext – Der Auftraggeber muss die Berechtigung vom Objekt-Eigentümer besitzen, eine spezifische Objekt-Operation auszuführen. In diesem Schritt wertet Amazon S3 die Objekt-ACL aus.

   Note

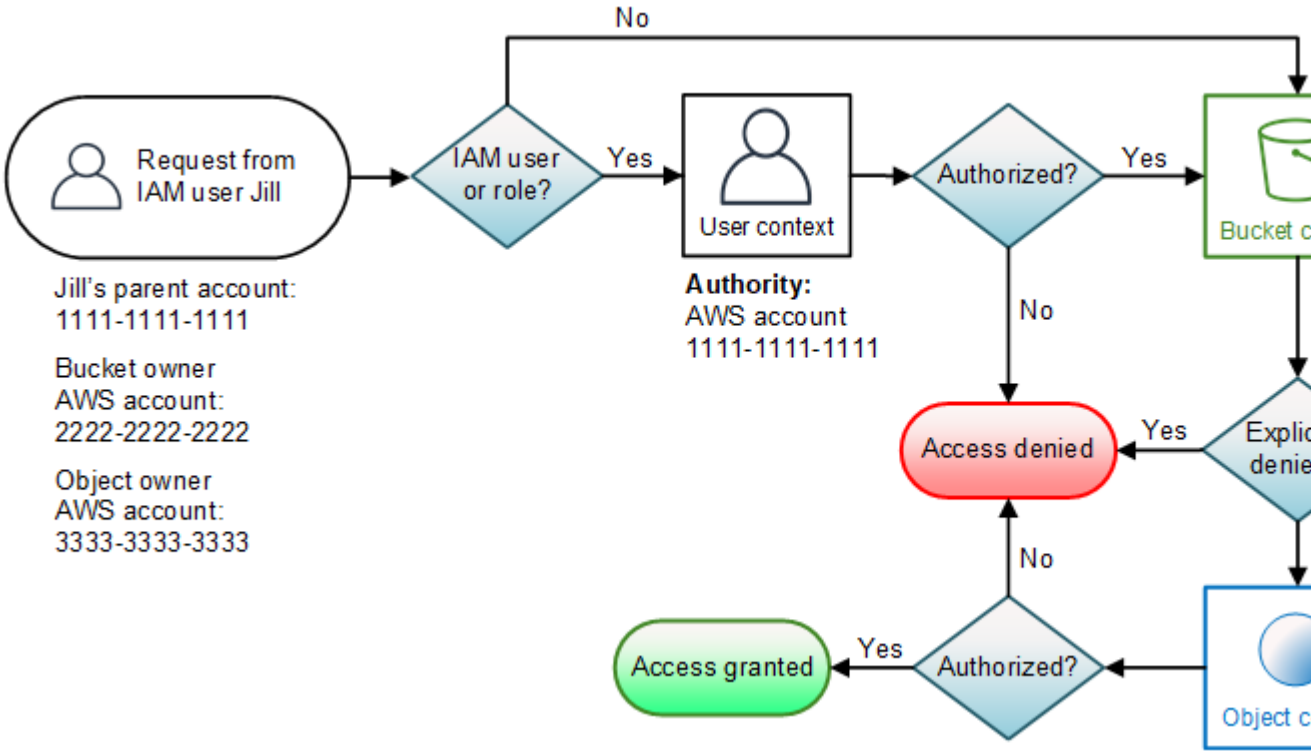
   Ist der Bucket-Eigentümer gleich dem Objekt-Eigentümer ist, kann der Zugriff auf das Objekt in der Bucket-Richtlinie erteilt werden, die im Bucket-Kontext ausgewertet wird. Unterscheiden sich die Eigentümer, müssen die Objekt-Eigentümer eine Objekt-ACL verwenden, um die Berechtigungen zu erteilen. Wenn das AWS-Konto, dem das Objekt gehört, auch das übergeordnete Konto des IAM-Prinzipals ist, kann es Objekt-Berechtigungen in einer Benutzerrichtlinie konfigurieren, die im Benutzerkontext ausgewertet wird. Weitere Informationen über die Verwendung dieser Alternativen zu Zugriffsrichtlinien finden Sie unter Orientierungshilfen für die Verwendung der unterstützten Zugriffsrichtlinienoptionen (p. 344).

Nachfolgend sehen Sie eine grafische Darstellung der kontextbasierten Auswertung für eine Objekt-Operation.



Beispiel 1: Anforderung einer Objekt-Operation

In diesem Beispiel sendet IAM-Benutzerin Jill, deren übergeordnetes AWS-Konto 1111-1111-1111 ist, eine Anforderung für eine Objekt-Operation (z. B. Get object) für ein Objekt, das dem AWS-Konto 3333-3333-3333 gehört, in einem Bucket, das dem AWS-Konto 2222-2222-2222 gehört.



Jill benötigt die Berechtigung vom übergeordneten AWS-Konto, dem Bucket-Eigentümer und dem Objekteigentümer. Amazon S3 wertet den Kontext wie folgt aus:

1. Da die Anforderung von einem IAM-Prinzipal stammt, überprüft Amazon S3 den Benutzerkontext daraufhin, ob das übergeordnete AWS-Konto 1111-1111-1111 Jill zum Ausführen der angeforderten Operation berechtigt hat. Wenn sie die Berechtigung besitzt, wertet Amazon S3 den Bucket-Kontext aus. Andernfalls lehnt Amazon S3 die Anforderung ab.
2. Im Bucket-Kontext ist der Bucket-Eigentümer, das AWS-Konto 2222-2222-2222, die Kontextautorität. Amazon S3 wertet die Bucket-Richtlinie aus, um festzustellen, ob der Bucket-Eigentümer Jill explizit die Berechtigung entzogen hat, auf das Objekt zuzugreifen.
3. Im Objektkontext ist die Kontextautorität das AWS-Konto 3333-3333-3333, der Objekteigentümer. Amazon S3 wertet die Objekt-ACL aus, um festzustellen, ob Jill die Berechtigung besitzt, auf das Objekt zuzugreifen. Ist dies der Fall, autorisiert Amazon S3 die Anforderung.

Orientierungshilfen für die Verwendung der unterstützten Zugriffsrichtlinienoptionen

Amazon S3 unterstützt ressourcenbasierte Richtlinien und Benutzerrichtlinien für die Verwaltung des Zugriffs auf Ihre Amazon S3-Ressourcen (siehe Verwalten des Zugriffs auf Ressourcen (p. 332)). Ressourcenbasierte Richtlinien umfassen Bucket-Richtlinien, Bucket-ACLs und Objekt-ACLs. Dieser Abschnitt beschreibt spezifische Szenarien für die Verwendung ressourcenbasierter Zugriffsrichtlinien für die Verwaltung des Zugriffs auf Ihre Amazon S3-Ressourcen.

Wann eine ACL-basierte Zugriffsrichtlinie verwendet wird (Bucket- und Objekt-ACLs)

Buckets und Objekten sind ACLs zugeordnet, die Sie verwenden können, um Berechtigungen zu erteilen. Die folgenden Abschnitte beschreiben Szenarien, in denen Objekt-ACLs und Bucket-ACLs verwendet werden.

Wann Objekt-ACLs verwendet werden

Neben einer Objekt-ACL gibt es noch andere Methoden, wie ein Objekt-Eigentümer Objekt-Berechtigungen verwalten kann. Beispiel:

- Wenn das AWS-Konto, dem das Objekt gehört, auch Eigentümer des Buckets ist, kann es eine Bucket-Richtlinie schreiben, um die Objektberechtigungen zu verwalten.
- Wenn das AWS-Konto, das Eigentümer des Objekts ist, einem Benutzer in seinem Konto Berechtigungen erteilen will, kann es eine Benutzerrichtlinie verwenden.

Wann verwenden Sie Objekt-ACLs zur Verwaltung von Objektberechtigungen? Nachfolgend finden Sie die Szenarien, in denen Sie Objekt-ACLs zur Verwaltung von Objektberechtigungen verwenden sollten.

- Eine Objekt-ACL ist die einzige Möglichkeit, den Zugriff auf Objekte zu verwalten, die nicht dem Bucket-Eigentümer gehören – Ein AWS-Konto, dem der Bucket gehört, kann einem anderen AWS-Konto die Berechtigung erteilen, Objekte hochzuladen. Dem Bucket-Eigentümer gehören diese Objekte nicht. Das AWS-Konto, das das Objekt erstellt hat, muss Berechtigungen unter Verwendung der Objekt-ACLs erteilen.

  > Note
  >
  > Ein Bucket-Eigentümer kann keine Berechtigungen für Objekte erteilen, die ihm nicht gehören. Beispielsweise gilt eine Bucket-Richtlinie, die Objektberechtigungen erteilt, nur für Objekte, die dem Bucket-Eigentümer gehören. Ein Bucket-Eigentümer, der die Rechnung zahlt, kann jedoch eine Bucket-Richtlinie schreiben, die den Zugriff auf Objekte im Bucket verweigern, unabhängig davon, wem dieser gehört. Der Bucket-Eigentümer kann auch beliebige Objekte im Bucket löschen

- Berechtigungen variieren zwischen den Objekten, und Sie müssen Berechtigungen auf Objektebene verwalten – Sie können eine einzige Richtlinienanweisung schreiben, die einem AWS-Konto Leseberechtigung für Millionen von Objekten mit einem bestimmten Schlüsselnamenspräfix erteilt. Beispielsweise könnten Sie Leseberechtigung für Objekte erteilen, die mit dem Schlüsselnamepräfix "logs" beginnen. Wenn Ihre Zugriffsberechtigungen jedoch zwischen den Objekten variieren, ist die Erteilung von Berechtigungen für einzelne Objekte unter Verwendung einer Bucket-Richtlinie möglicherweise nicht praktisch. Außerdem sind Bucket-Richtlinien auf eine Größe von 20 KB beschränkt.

  In diesem Fall ist die Verwendung von Objekt-ACLs möglicherweise eine geeignetere Alternative. Das gilt auch, obwohl eine Objekt-ACL ebenfalls auf maximal 100 Rechte beschränkt ist (siehe Zugriffskontrolllisten (ACL) – Übersicht (p. 480)).

- Objekt-ACLs steuern nur Berechtigungen auf Objektebene – Es gibt eine Bucket-Richtlinie für den gesamten Bucket, Objekt-ACLs werden jedoch pro Objekt eingerichtet.

  Ein AWS-Konto, dem ein Bucket gehört, kann einem anderen AWS-Konto die Berechtigung erteilen, die Zugriffsrichtlinie zu verwalten. Es gestattet diesem Konto, die Richtlinie beliebig zu verändern. Um die Berechtigungen besser verwalten zu können, sollten Sie keine so allgemeine Berechtigung erteilen, sondern besser nur READ-ACP - und WRITE-ACP-Berechtigungen für Untermengen von Objekten. Damit kann das Konto nur für bestimmte Objekte Berechtigungen verwalten, indem es die ACLs einzelner Objekte aktualisiert.

Verwendung einer Bucket-ACL

Der einzige empfohlene Anwendungsfall für die Bucket-ACL ist, der Amazon S3 Log Delivery-Gruppe Schreibberechtigung zu erteilen, damit sie Zugriffsprotokollobjekte in Ihren Bucket schreiben kann (siehe Amazon S3-Serverzugriffsprotokollierung (p. 777)). Wenn Amazon S3 Zugriffsprotokolle in Ihren Bucket schreiben soll, müssen sie der Log Delivery-Gruppe Schreibberechtigung für den Bucket erteilen. Die einzige Möglichkeit, wie Sie der Log Delivery-Gruppe die erforderlichen Berechtigungen erteilen können, ist eine Bucket-ACL, wie im folgenden Auszug aus einer Bucket-ACL gezeigt.

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
   ...
  </Owner>
  <AccessControlList>
    <Grant>
        ...
    </Grant>
    <Grant>
       <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
       </Grantee>
       <Permission>WRITE</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Verwendung einer Bucket-Richtlinie

Wenn ein AWS-Konto, dem ein Bucket gehört, Benutzern Berechtigungen in seinem Konto erteilen will, kann es eine Bucket-Richtlinie oder eine Benutzerrichtlinie verwenden. In den folgenden Szenarien müssen Sie eine Bucket-Richtlinie verwenden.

- Sie möchten kontenübergreifende Berechtigungen für alle Amazon S3-Berechtigungen verwenden – Sie können ACLs verwenden, um kontenübergreifende Berechtigungen für andere Konten zu erteilen, aber ACLs unterstützen nur eine begrenzte Menge an Berechtigungen (Welche Berechtigungen kann ich erteilen? (p. 483)), nicht alle Amazon S3-Berechtigungen. Sie können beispielsweise mit einer ACL keine Berechtigungen für Bucket-Subressourcen erteilen (siehe Identity and Access Management in Amazon S3 (p. 329)).

  Obwohl sowohl Bucket- als auch Benutzerrichtlinien die Erteilung von Berechtigungen für alle Amazon S3-Operationen unterstützen (siehe Amazon S3-Aktionen (p. 380)), sind die Benutzerrichtlinien dafür vorgesehen, die Berechtigungen für Benutzer in Ihrem Konto zu verwalten. Für kontenübergreifende Berechtigungen für andere AWS-Konten oder Benutzer in einem anderen Konto müssen Sie eine Bucket-Richtlinie verwenden.

Verwendung einer Benutzerrichtlinie

Im Allgemeinen können Sie eine Benutzerrichtlinie oder eine Bucket-Richtlinie verwenden, um Berechtigungen zu verwalten. Sie können Berechtigungen verwalten, indem Sie Benutzer erstellen und die Berechtigungen einzeln verwalten, indem Sie den Benutzern (oder Benutzergruppen) Richtlinien hinzufügen, oder Sie können ressourcenbasierte Richtlinien verwenden, wie beispielsweise eine Bucket-Richtlinie, wenn dies für Ihr Szenario besser geeignet ist.

Beachten Sie, dass AWS Identity and Access Management (IAM) ermöglicht, mehrere Benutzer innerhalb Ihres AWS-Kontos zu erstellen und ihre Berechtigungen über Benutzerrichtlinien zu verwalten. Ein IAM-Benutzer muss Berechtigungen von dem übergeordneten AWS-Konto erhalten, zu dem er gehört, und von

dem AWS-Konto, dem die Ressource gehört, auf die der Benutzer Zugriff erhalten will. Die Berechtigungen können folgt erteilt werden:

- Berechtigung vom übergeordneten Konto – Das übergeordnete Konto kann seinem Benutzer Berechtigungen erteilen, indem es ihm eine Benutzerrichtlinie zuordnet.
- Berechtigung vom Ressourceneigentümer – Der Ressourceneigentümer kann dem IAM-Benutzer Berechtigungen erteilen (unter Verwendung einer Bucket-Richtlinie) oder dem übergeordneten Konto (unter Verwendung einer Bucket-Richtlinie, Bucket-ACL oder Objekt-ACL).

Das ist vergleichbar mit einem Kind, das mit einem Spielzeug spielen möchte, das jemandem anderen gehört. In diesem Fall muss das Kind die Berechtigung von einem Elternteil erhalten, mit dem Spielzeug zu spielen, als auch eine Berechtigung vom Eigentümer des Spielzeugs.

Berechtigungsdelegation

Wenn einem AWS-Konto eine Ressource gehört, kann es diese Berechtigungen einem anderen AWS-Konto erteilen. Dieses Konto kann diese Berechtigungen oder eine Untermenge davon an Benutzer in dem Konto delegieren. Dies wird auch als Berechtigungsdelegation bezeichnet. Ein Konto, das Berechtigungen von einem anderen Konto erhält, kann die Berechtigungen nicht kontenübergreifend an ein anderes AWS-Konto delegieren.

Verwandte Themen

Wir empfehlen Ihnen, zunächst alle einführenden Themen zu lesen, die erklären, wie Sie den Zugriff auf Ihre Amazon S3-Ressourcen verwalten, sowie alle zugehörigen Anleitungen. Weitere Informationen finden Sie unter Einführung in das Verwalten des Zugriffs auf Amazon S3-Ressourcen (p. 330). Die folgenden Themen enthalten weitere Informationen über spezifische Zugriffsrichtlinienoptionen.

- Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375)
- Zugriffskontrolllisten (ACL) – Übersicht (p. 480)
- Zugriffsverwaltung mit ACLs (p. 479)

Beispielhafte Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen

In diesem Thema werden die folgenden einführenden Anleitungsbeispiele für das Gewähren des Zugriffs aufs Amazon S3-Ressourcen bereitgestellt. Diese Beispiele verwenden die AWS Management Console, um Ressourcen (Buckets, Objekte, Benutzer) zu erstellen und diesen Berechtigungen zu erteilen. Anschließend zeigen die Beispiele auf, wie Sie Berechtigungen mithilfe der Befehlszeilen-Tools überprüfen können, sodass Sie keinen Code schreiben müssen. Wir stellen Befehle bereit, die sowohl die AWS-Befehlszeilenschnittstelle (Command Line Interface, CLI) und die AWS-Tools für Windows PowerShell verwenden.

- Beispiel 1: Bucket-Eigentümer erteilt seinen Benutzern Bucket-Berechtigungen (p. 351)

  Die in Ihrem Konto erstellten IAM-Benutzer verfügen standardmäßig über keine Berechtigungen. In dieser Übung erteilen Sie einem Benutzer die Berechtigung, Bucket- und Objektoperationen auszuführen.
- Beispiel 2: Bucket-Eigentümer erteilt kontoübergreifende Bucket-Berechtigungen (p. 355)

  In dieser Übung gewährt ein Bucket-Eigentümer, Konto A, einem anderen AWS-Konto, Konto B, kontoübergreifende Berechtigungen. Konto B delegiert diese Berechtigungen anschließend an Benutzer in seinem Konto.
- Verwalten von Objektberechtigungen, wenn der Objekt- und der Bucket-Eigentümer nicht identisch sind

Bei den Beispielszenarien in diesem Fall geht es um einen Bucket-Eigentümer, der anderen Objektberechtigungen erteilt, es gehören jedoch nicht alle Objekte im Bucket dem Bucket-Eigentümer. Welche Berechtigungen benötigt der Bucket-Eigentümer und wie kann er diese Berechtigungen delegieren?

Das AWS-Konto, das einen Bucket erstellt, wird als Bucket-Eigentümer bezeichnet. Der Eigentümer kann anderen AWS-Konten die Berechtigung zum Hochladen von Objekten erteilen und die AWS-Konten, die Objekte erstellen, besitzen diese. Der Bucket-Eigentümer hat keine Berechtigungen für diese Objekte, die von anderen AWS-Konten erstellt wurden. Wenn der Bucket-Eigentümer eine Bucket-Richtlinie erstellt, die Zugriff auf Objekte erteilt, gilt diese Richtlinie nicht für Objekte, die sich im Besitz von anderen Konten befinden.

In diesem Fall muss der Objekteigentümer zuerst dem Bucket-Eigentümer über eine Objekt-ACL Berechtigungen erteilen. Der Bucket-Eigentümer kann diese Objektberechtigungen anschließend an andere delegieren – an Benutzer in seinem eigenen Konto oder an ein anderes AWS-Konto wie in den folgenden Beispielen dargestellt.

- Beispiel 3: Bucket-Eigentümer, der seinen Benutzern Berechtigungen für Objekte erteilt, die ihm nicht gehören (p. 361)

  In dieser Übung erhält der Bucket-Eigentümer zuerst Berechtigungen von dem Objekteigentümer. Der Bucket-Eigentümer delegiert diese Berechtigungen anschließend an Benutzer in seinem eigenen Konto.
- Beispiel 4: Der Bucket-Eigentümer erteilt eine kontenübergreifende Berechtigung für Objekte, die ihm nicht gehören (p. 366)

  Wenn der Bucket-Eigentümer Berechtigungen von dem Objekteigentümer erhalten hat, kann er keine Berechtigung an andere AWS-Konten delegieren, da die kontoübergreifende Delegation nicht unterstützt wird (siehe Berechtigungsdelegation (p. 347)). Stattdessen kann der Bucket-Eigentümer eine IAM-Rolle mit Berechtigungen erstellen, um bestimmte Operationen (wie beispielsweise das Abrufen von Objekten) auszuführen, und einem anderen AWS-Konto erlauben, diese Rolle anzunehmen. Jeder, der diese Rolle annimmt, kann anschließend auf Objekte zugreifen. Dieses Beispiel zeigt, wie ein Bucket-Eigentümer diese kontoübergreifende Delegation mithilfe einer IAM-Rolle aktivieren kann.

Bevor Sie die beispiehalften Walkthroughs ausprobieren

Diese Beispiele verwenden die AWS Management Console, um Ressourcen zu erstellen und Berechtigungen zu erteilen. Zum Testen der Berechtigungen verwenden die Beispiele die Befehlszeilen-Tools, AWS Command Line Interface (CLI) und AWS-Tools für Windows PowerShell, sodass Sie keinen Code schreiben müssen. Zum Testen der Berechtigungen müssen Sie eins dieser Tools einrichten. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

Darüber hinaus verwenden diese Beispiele beim Erstellen von Ressourcen keine Root-Anmeldeinformationen von AWS-Konten. Sie erstellen stattdessen einen Administratorbenutzer in diesen Konten, um diese Aufgaben auszuführen.

Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen

AWS Identity and Access Management (IAM) rät davon ab, die Root-Anmeldeinformationen Ihres AWS-Kontos für Anfragen zu nutzen. Erstellen Sie stattdessen einen IAM-Benutzer, gewähren Sie diesem vollständigen Zugriff und verwenden Sie anschließend die Anmeldeinformationen dieses Benutzers für die Interaktion mit AWS. Wir bezeichnen diesen Benutzer als Administratorbenutzer. Weitere Informationen finden Sie unter Root-Konto-Anmeldeinformationen vs. IAM-Benutzeranmeldeinformationen in der AWS General Reference sowie unter Bewährte Methoden für IAM im IAM-Benutzerhandbuch.

Alle Beispielanleitungen in diesem Abschnitt verwenden die Anmeldeinformationen des Administratorbenutzers. Wenn Sie noch keinen Administratorbenutzer für Ihr AWS-Konto erstellt haben, erfahren Sie in diesen Themen, wie dies geht.

Beachten Sie, dass Sie die URL für die Anmeldung des IAM-Benutzers verwenden müssen, um sich mithilfe der Anmeldeinformationen bei der AWS Management Console anzumelden. Die IAM-Konsole stellt diese URL für Ihr AWS-Konto bereit. In diesen Themen erfahren Sie, wie Sie die URL abrufen können.

Einrichten der Tools für die beispielhaften Walkthroughs

Die einführenden Beispiele (siehe Beispielhafte Walkthroughs: Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen (p. 347)) verwenden die AWS Management Console, um Ressourcen zu erstellen und Berechtigungen zu erteilen. Zum Testen der Berechtigungen verwenden die Beispiele die Befehlszeilen-Tools, AWS Command Line Interface (CLI) und AWS-Tools für Windows PowerShell, sodass Sie keinen Code schreiben müssen. Zum Testen der Berechtigungen müssen Sie eins dieser Tools einrichten.

Einrichten von AWS CLI

1. Herunterladen und Konfigurieren der AWS CLI. Anleitungen können Sie den folgenden Themen im Benutzerhandbuch für AWS Command Line Interface entnehmen.

   Einrichten mit der AWS-Befehlszeilenschnittstelle

   Installieren der AWS-Befehlszeilenschnittstelle

   Configuring the AWS Command Line Interface
2. Richten Sie das Standardprofil ein.

   Sie speichern Ihre Anmeldeinformationen in der AWS CLI-Config-Datei. Erstellen Sie mit den Anmeldeinformationen für Ihr AWS-Konto ein Standardprofil in der Config-Datei. Unter Konfigurations- und Anmeldeinformationsdateien finden Sie eine Anleitung dazu, wie Sie die AWS CLI-Konfigurationsdatei finden und bearbeiten können.

   ```
   [default]
   aws_access_key_id = access key ID
   aws_secret_access_key = secret access key
   region = us-west-2
   ```
3. Überprüfen Sie die Einrichtung, indem Sie den folgenden Befehl in die Befehlszeile eingeben. Beide Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Probieren Sie den Hilfebefehl aus.

     ```
     aws help
     ```
   - Verwenden Sie `aws s3 ls`, um eine Liste der Buckets auf dem konfigurierten Konto abzurufen.

     ```
     aws s3 ls
     ```

Im Verlauf dieser Beispiel-Anleitungen erstellen Sie Benutzer und speichern Anmeldeinformationen in den Config-Dateien, indem Sie Profile erstellen. Dies wird im folgenden Beispiel dargestellt. Beachten Sie, dass diese Profile Namen haben (AccountAadmin and AccountBadmin):

```
[profile AccountAadmin]
aws_access_key_id = User AccountAadmin access key ID
aws_secret_access_key = User AccountAadmin secret access key
```

```
region = us-west-2

[profile AccountBadmin]
aws_access_key_id = Account B access key ID
aws_secret_access_key = Account B secret access key
region = us-east-1
```

Um mit diesen Anmeldeinformationen einen Befehl auszuführen, fügen Sie den Parameter `--profile` hinzu, um den Profilnamen festzulegen. Der folgende AWS CLI-Befehl ruft eine Auflistung von Objekten in `examplebucket` ab und legt das Profil `AccountBadmin` fest.

```
aws s3 ls s3://examplebucket --profile AccountBadmin
```

Alternativ können Sie eine Reihe von Anmeldeinformationen als Standardprofil konfigurieren, indem Sie die Umgebungsvariable `AWS_DEFAULT_PROFILE` von der Befehlszeile aus ändern. Wenn Sie dies vorgenommen haben, verwendet die AWS CLI jedes Mal, wenn Sie AWS CLI-Befehle ohne den Parameter `--profile` ausführen, das Profil, das Sie in der Umgebungsvariablen als Standardprofil festgelegt haben.

```
$ export AWS_DEFAULT_PROFILE=AccountAadmin
```

So richten Sie AWS-Tools for Windows PowerShell ein

1. Laden Sie die AWS-Tools für Windows PowerShell herunter und konfigurieren Sie diese. Anleitungen finden Sie unter Download and Install the AWS Tools for Windows PowerShell im AWS-Tools für Windows PowerShell-Benutzerhandbuch.

   Note

   Um das Modul AWS-Tools für Windows PowerShell laden zu können, muss die Ausführung von PowerShell-Skripts aktiviert sein. Weitere Informationen finden Sie unter Enable Script Execution im AWS-Tools für Windows PowerShell-Benutzerhandbuch.

2. Für diese Übungen legen Sie AWS-Benutzerinformationen pro Sitzung mit dem Befehl `Set-AWSCredentials` fest. Der Befehl speichert die Anmeldeinformationen in einem persistenten Speicher (Parameter `-StoreAs`).

   ```
   Set-AWSCredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas string
   ```

3. Überprüfen Sie die Einrichtung.

   - Führen Sie den Befehl `Get-Command` aus, um eine Liste der verfügbaren Befehle für Amazon S3-Operationen abzurufen.

     ```
     Get-Command -module awspowershell -noun s3* -StoredCredentials string
     ```

   - Führen Sie den Befehl `Get-S3Object` aus, um eine Liste von Objekten in einem Bucket abzurufen.

     ```
     Get-S3Object -BucketName bucketname -StoredCredentials string
     ```

Eine Liste mit Befehlen finden Sie unter Amazon Simple Storage Service Cmdlets.

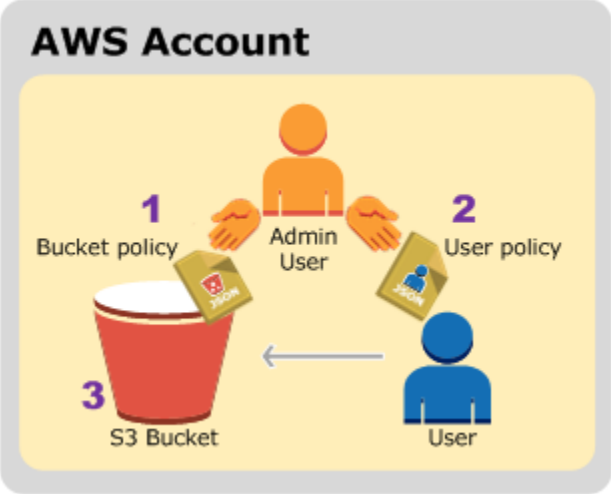
Jetzt können Sie die Übungen ausprobieren. Folgen Sie den Links am Anfang des Abschnitts.

Beispiel 1: Bucket-Eigentümer erteilt seinen Benutzern Bucket-Berechtigungen

Themen

In dieser Übung besitzt ein AWS-Konto einen Bucket, und es hat einen IAM-Benutzer im Konto. Standardmäßig hat der Benutzer keine Berechtigungen. Damit der Benutzer alle Aufgaben ausführen kann, muss das übergeordnete Konto ihm Berechtigungen erteilen. Der Bucket-Eigentümer und das übergeordnete Konto sind identisch. Um den Benutzern Berechtigungen für den Bucket zu gewähren, kann das AWS-Konto daher eine Bucket-Richtlinie, eine Benutzerrichtlinie oder beides verwenden. Der Kontobesitzer gewährt einige Berechtigungen unter Verwendung einer Bucket-Richtlinie und andere Berechtigungen unter Verwendung einer Benutzerrichtlinie.

Die folgenden Schritte fasen das detaillierte Beispiel zusammen:



1. Der Kontoadministrator erstellt eine Bucket-Richtlinie, die dem Benutzer verschiedene Berechtigungen erteilt.
2. Der Kontoadministrator weist dem Benutzer eine Benutzerrichtlinie zu, die ihm zusätzliche Berechtigungen erteilt.
3. Anschließend probiert der Benutzer Berechtigungen aus, die über die Bucket-Richtlinie und die Benutzerrichtlinie erteilt wurden.

Für dieses Beispiel benötigen Sie ein AWS-Konto. Statt die Root-Anmeldeinformationen für Konto zu verwenden, erstellen Sie einen Administrator-Benutzer (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen (p. 348)). Wir verweisen wie folgt auf das AWS-Konto und den Administrator-Benutzer:

Konto-ID	Konto bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin

Alle Aufgaben in Verbindung mit dem Erstellen von Benutzern und Gewähren von Berechtigungen werden in der AWS Management Console ausgeführt. Um die Berechtigungen zu überprüfen, verwendet das detaillierte Beispiel die Befehlszeilen-Tools, AWS Command Line Interface (CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code zu schreiben brauchen.

Schritt 0: Vorbereitung auf den Walkthrough

1. Stellen Sie sicher, dass Sie ein AWS-Konto haben, und dass es einen Benutzer mit Administratorrechten enthält.

   a. Melden Sie sich nach Bedarf für ein Konto an. Wir bezeichnen dieses Konto als Konto A.

      i. Öffnen Sie https://aws.amazon.com/s3 und klicken Sie auf Sign Up (Anmelden).

      ii. Folgen Sie den Anweisungen auf dem Bildschirm.

      Sie werden per E-Mail von AWS benachrichtigt, sobald Ihr Konto aktiv ist und von Ihnen verwendet werden kann.

   b. Erstellen Sie in Konto A einen Administratorbenutzers AccountAadmin. Melden Sie sich mit den Anmeldeinformationen von Konto A in der IAM-Konsole an und machen Sie Folgendes:

      i. Erstellen Sie den Benutzer AccountAadmin und schreiben Sie sich die Sicherheitsanmeldeinformationen für den Benutzer auf.

      Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

      ii. Erteilen Sie dem Administrator AccountAadmin Berechtigungen, indem Sie ihm eine Benutzerrichtlinie zuordnen, die ihm vollen Zugriff bietet.

      Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.

      iii. Schreiben Sie sich die URL für die Anmeldung des IAM-Benutzers für AccountAadmin auf. Sie brauchen diese URL, wenn Sie sich bei der AWS Management Console anmelden. Weitere Informationen darüber, wo Sie sie finden, erhalten Sie unter Wie sich Benutzer in Ihrem Konto anmelden in IAM-Benutzerhandbuch. Notieren Sie die URLs für alle Konten.

2. Richten Sie die AWS CLI (Command Line Interface) oder die AWS Tools for Windows PowerShell ein. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

   - Wenn Sie die AWS-CLI verwenden, erstellen Sie das Profil "AccountAadmin" in der Konfigurationsdatei.
   - Wenn Sie die AWS-Tools für Windows PowerShell verwenden, stellen Sie sicher, dass sie die Anmeldeinformationen für die Sitzung als AccountAadmin speichern.

   Detaillierte Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

Schritt 1: Erstellen von Ressourcen (einen Bucket und einen IAM-Benutzer) in Konto A und Erteilen von Berechtigungen

Melden Sie sich mit den Anmeldeinformationen von Benutzer AccountAadmin in Konto A und der speziellen URL für die Anmeldung eines IAM-Benutzers in der AWS Management Console an und machen Sie Folgendes:

1. Ressourcen erstellen (einen Bucket und einen IAM-Benutzer)

a. Erstellen Sie in der Amazon S3-Konsole einen Bucket. Schreiben Sie sich die AWS-Region auf, in der Sie ihn erstellt haben. Weitere Informationen finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

b. Führen Sie in der IAM-Konsole die folgenden Schritte aus:

   i. Erstellen Sie einen Benutzer, Dave.

      Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

   ii. Schreiben Sie sich die Anmeldeinformationen für UserDave auf.

   iii. Notieren Sie den Amazon-Ressourcennamen (ARN) für den Benutzer Dave. Wählen Sie in der IAM-Konsole den Benutzer aus. Sie finden den ARN des Benutzers auf der Registerkarte Summary.

2. Erteilen Sie Berechtigungen.

   Der Bucket-Eigentümer und das übergeordnete Konto, zu dem der Benutzer gehört, sind identisch, das AWS-Konto kann deshalb Benutzerberechtigungen unter Verwendung einer Bucket-Richtlinie, einer Benutzerrichtlinie oder beidem erteilen. In diesem Beispiel machen Sie beides. Wenn das Objekt auch demselben Konto gehört, kann der Bucket-Eigentümer in der Bucket-Richtlinie (oder einer IAM-Richtlinie) Objektberechtigungen erteilen.

   a. Weisen Sie in der Amazon S3-Konsole die folgende Bucket-Richtlinie *examplebucket* zu.

      Die Richtlinie enthält zwei Anweisungen.

      - Die erste Anweisung erteilt Dave Berechtigungen für die Bucket-Operationen `s3:GetBucketLocation` und `s3:ListBucket`.
      - Die zweite Anweisung erteilt die `s3:GetObject`-Berechtigung. Konto A gehört auch das Objekt, deshalb kann der Kontoadministrator die `s3:GetObject`-Berechtigung erteilen.

      In der `Principal`-Anweisung wird Dave durch seinen Benutzer-ARN identifiziert. Weitere Informationen zu Richtlinienelementen finden Sie unter Richtlinien und Berechtigungen in Amazon S3 (p. 376).

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "statement1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
            "s3:GetBucketLocation",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket"
         ]
      },
      {
         "Sid": "statement2",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
```

```
            "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket/*"
         ]
      }
   ]
}
```

b. Erstellen Sie mithilfe der folgenden Richtlinie eine Inlinerichtlinie für den Benutzer Dave. Die Richtlinie erteilt dem Benutzer Dave die `s3:PutObject`-Berechtigung. Sie müssen die Richtlinie aktualisieren, indem Sie Ihren Bucket-Namen angeben.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PermissionForObjectOperations",
         "Effect": "Allow",
         "Action": [
            "s3:PutObject"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket/*"
         ]
      }
   ]
}
```

Weitere Informationen finden Sie unter Verwenden von eingebundenen Richtlinien im IAM-Benutzerhandbuch. Beachten Sie, dass Sie sich mit den Anmeldeinformationen von Konto A an der Konsole anmelden müssen.

Schritt 2: Testen der Berechtigungen

Überprüfen Sie unter Verwendung der Anmeldeinformationen von Dave, ob die Berechtigungen funktionieren. Sie haben die Wahl zwischen den folgenden beiden Verfahren.

Test mit der AWS CLI

1. Aktualisieren Sie die AWS CLI-Konfigurationsdatei, indem Sie das folgende Profil UserDaveAccountA hinzufügen. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

```
[profile UserDaveAccountA]
aws_access_key_id = access-key
aws_secret_access_key = secret-access-key
region = us-east-1
```

2. Überprüfen Sie, ob Dave Operationen ausführen kann, für die ihm in der Benutzerrichtlinie Berechtigungen erteilt wurden. Laden Sie ein Beispielobjekt unter Verwendung des folgenden AWS CLI `put-object`-Befehls hoch.

   Der Parameter `--body` im Befehl identifiziert die hochzuladende Quelldatei. Befindet sich die Datei beispielsweise im Stammverzeichnis auf dem Laufwerk C: eines Windows-Computers, geben Sie `c:\HappyFace.jpg` an. Der Parameter `--key` gibt den Schlüsselnamen für das Objekt an.

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body HappyFace.jpg --
profile UserDaveAccountA
```

Führen Sie den folgenden AWS CLI-Befehl aus, um das Objekt zu erhalten.

```
aws s3api get-object --bucket examplebucket --key HappyFace.jpg OutputFile.jpg --
profile UserDaveAccountA
```

Testen mit den AWS Tools für Windows PowerShell

1. Speichern Sie die Anmeldeinformationen von Dave als AccountADave. Anschließend verwenden Sie diese Anmeldeinformationen für ein PUT und ein GET für ein Objekt.

```
set-awscredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas
 AccountADave
```

2. Laden Sie ein Beispielobjekt mit dem Befehl `Write-S3Object` der AWS-Tools für Windows PowerShell hoch. Verwenden Sie dazu die gespeicherten Anmeldeinformationen des Benutzers Dave.

```
Write-S3Object -bucketname examplebucket -key HappyFace.jpg -file HappyFace.jpg -
StoredCredentials AccountADave
```

Laden Sie das zuvor hochgeladene Objekt herunter.

```
Read-S3Object -bucketname examplebucket -key HappyFace.jpg -file Output.jpg -
StoredCredentials AccountADave
```

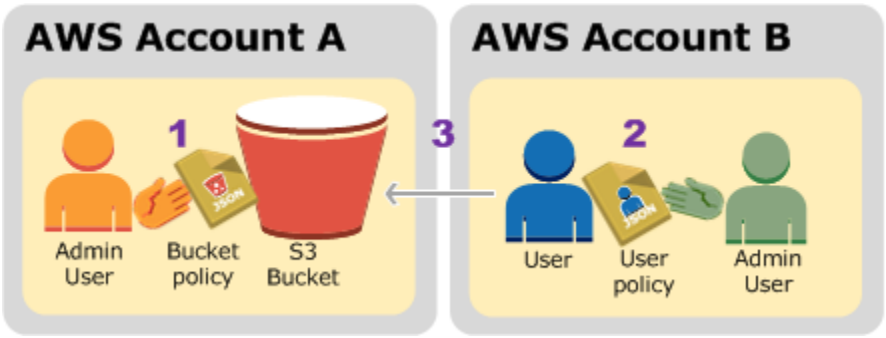
Beispiel 2: Bucket-Eigentümer erteilt kontoübergreifende Bucket-Berechtigungen

Themen

Ein AWS-Konto – z. B. Konto A – kann einem anderen AWS-Konto, Konto B, die Berechtigung erteilen, auf seine Ressourcen zuzugreifen, wie beispielsweise Buckets und Objekte. Konto B kann diese Berechtigungen an Benutzer in seinem Konto delegieren. In diesem Beispielszenario erteilt ein Bucket-Eigentümer einem anderen Konto eine kontenübergreifende Berechtigung, um bestimmte Bucket-Operationen auszuführen.

> **Note**
>
> Konto A kann einem Benutzer in Konto B unter Verwendung einer Bucket-Richtlinie auch direkt Berechtigungen erteilen. Der Benutzer braucht dennoch eine Berechtigung von seinem übergeordneten Konto, Konto B, zu dem der Benutzer gehört, auch wenn Konto B keine Berechtigungen von Konto A erhalten hat. Solange der Benutzer die Berechtigung von dem Ressourcen-Eigentümer und dem übergeordneten Konto hat, kann der Benutzer auf die Ressource zugreifen.

Nachfolgend finden Sie eine kurze Zusammenfassung der wichtigsten Details:



1. Der Administrator-Benutzer von Konto A ordnet eine Bucket-Richtlinie zu, die Konto B kontenübergreifende Berechtigungen erteilt, um bestimmte Bucket-Operationen auszuführen.

   Beachten Sie, dass der Administrator-Benutzer in Konto B die Berechtigungen automatisch erbt.
2. Der Administrator-Benutzer von Konto B ordnet dem Benutzer eine Benutzerrichtlinie zu, die die Berechtigungen an den Benutzer delegiert, die er von Konto A erhalten hat.
3. Der Benutzer in Konto B überprüft die Berechtigungen, indem er auf ein Objekt in dem Bucket zugreift, das Konto A gehört.

Für dieses Beispiel benötigen Sie zwei Konten. Die folgende Tabelle zeigt, wie wir auf diese Konten und die Administrator-Benutzer darin verweisen. Laut der IAM-Richtlinien (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen (p. 348)) verwenden wir in dieser schrittweisen Anleitung nicht die Root-Anmeldeinformationen für das Konto. Stattdessen erstellen Sie einen Administrator-Benutzer in jedem Konto, und verwenden dessen Anmeldeinformationen, um Ressourcen zu erstellen und ihnen Berechtigungen zu erteilen.

AWS-Konto-ID	Konto bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin
2222-2222-2222	Konto B	AccountBadmin

Alle Aufgaben in Verbindung mit dem Erstellen von Benutzern und Gewähren von Berechtigungen werden in der AWS Management Console ausgeführt. Um die Berechtigungen zu überprüfen, verwendet das detaillierte Beispiel die Befehlszeilen-Tools, AWS Command Line Interface (CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code zu schreiben brauchen.

Schritt 0: Vorbereitung auf den Walkthrough

1. Stellen Sie sicher, dass Sie zwei AWS-Konten haben, und dass jedes Konto einen Administrator hat, wie in der Tabelle im vorigen Abschnitt gezeigt.

   a. Melden Sie sich nach Bedarf für ein AWS-Konto an.

      i. Öffnen Sie https://aws.amazon.com/s3/ und wählen Sie Create an AWS Account (Erstellen eines AWS-Kontos) aus.

      ii. Folgen Sie den Anweisungen auf dem Bildschirm.

      Sie werden per E-Mail von AWS benachrichtigt, sobald Ihr Konto aktiv ist und von Ihnen verwendet werden kann.

   b. Melden Sie sich mit den Anmeldeinformationen für Konto A an der IAM-Konsole an und erstellen Sie wie folgt den Administrator-Benutzer:

i. Erstellen Sie den Benutzer AccountAadmin und schreiben Sie sich die Sicherheitsanmeldeinformationen auf. Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

ii. Erteilen Sie dem Administrator AccountAadmin Berechtigungen, indem Sie ihm eine Benutzerrichtlinie zuordnen, die ihm vollen Zugriff bietet. Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.

c. Während Sie in der IAM-Konsole arbeiten, schreiben Sie sich die URL für die Anmeldung des IAM-Benutzers auf dem Dashboard auf. Alle Benutzer in diesem Konto müssen diese URL für die Anmeldung an der AWS Management Console verwenden.

Weitere Informationen finden Sie unter Wie sich Benutzer bei Ihrem Konto anmelden im IAM-Benutzerhandbuch.

d. Wiederholen Sie den obigen Schritt unter Verwendung der Anmeldeinformationen von Konto B und erstellen Sie den Administrator-Benutzer AccountBadmin.

2. Richten Sie die AWS CLI (Command Line Interface) oder die AWS Tools for Windows PowerShell ein. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

- Wenn Sie die AWS CLI verwenden, erstellen Sie zwei Profile, AccountAadmin und AccountBadmin, in der Konfigurationsdatei.
- Wenn Sie die AWS Tools for Windows PowerShell verwenden, stellen Sie sicher, dass sie die Anmeldeinformationen für die Sitzung als AccountAadmin und AccountBadmin speichern.

Detaillierte Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

3. Speichern Sie die Anmeldeinformationen des Administrator-Benutzers, auch als Profile bezeichnet. Sie können den Profilnamen verwenden, statt für jeden eingegebenen Befehl Anmeldeinformationen anzugeben. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

a. Fügen Sie der AWS-CLI-Anmeldeinformationsdatei für jeden der Administrator-Benutzer in den beiden Konten Profile hinzu.

```
[AccountAadmin]
aws_access_key_id = access-key-ID
aws_secret_access_key = secret-access-key
region = us-east-1

[AccountBadmin]
aws_access_key_id = access-key-ID
aws_secret_access_key = secret-access-key
region = us-east-1
```

b. Wenn Sie die AWS Tools für Windows PowerShell verwenden

```
set-awscredentials -AccessKey AcctA-access-key-ID -SecretKey AcctA-secret-access-
key -storeas AccountAadmin
set-awscredentials -AccessKey AcctB-access-key-ID -SecretKey AcctB-secret-access-
key -storeas AccountBadmin
```

Schritt 1: Erledigen der Aufgaben von Konto A

Schritt 1.1: Anmelden bei der AWS-Managementkonsole

Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto A zuerst als Benutzer AccountAadmin bei der AWS Management Console an. Dieser Benutzer erstellt einen Bucket und ordnet ihm eine Richtlinie zu.

Schritt 1.2: Erstellen eines Buckets

1. Erstellen Sie in der Amazon S3-Konsole einen Bucket. Diese Übung geht davon aus, dass der Bucket in der Region USA Ost (Nord-Virginia) erstellt wurde und `examplebucket` heißt.

   Weitere Informationen finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
2. Hochladen eines Beispielobjekts in den Bucket.

   Weitere Informationen finden Sie unter Objekte in einen Bucket einfügen im Amazon Simple Storage Service Handbuch Erste Schritte.

Schritt 1.3: Zuordnen einer Bucket-Richtlinie, um Konto B kontoübergreifende Berechtigungen zu erteilen

Die Bucket-Richtlinie erteilt Konto B die Berechtigungen `s3:GetBucketLocation` und `s3:ListBucket`. Es wird angenommen, dass Sie noch mit den Benutzeranmeldeinformationen von AccountAadmin an der Konsole angemeldet sind.

1. Weisen Sie `examplebucket` die folgende Bucket-Richtlinie zu. Die Richtlinie erteilt Konto B die Berechtigung für die Aktionen `s3:GetBucketLocation` und `s3:ListBucket`.

   Weitere Informationen finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example permissions",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:GetBucketLocation",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket"
         ]
      }
   ]
}
```

2. Überprüfen, ob Konto B (und damit der Administrator-Benutzer) die Operationen ausführen kann.

   - Verwenden der AWS CLI

```
aws s3 ls s3://examplebucket --profile AccountBadmin
aws s3api get-bucket-location --bucket examplebucket --profile AccountBadmin
```

- Verwendung der AWS Tools für Windows PowerShell

```
get-s3object -BucketName example2bucket -StoredCredentials AccountBadmin
get-s3bucketlocation -BucketName example2bucket -StoredCredentials AccountBadmin
```

Schritt 2: Erledigen der Aufgaben von Konto B

Jetzt erstellt der Administrator von Konto B einen Benutzer Dave und delegiert an Dave die Berechtigungen, die er von Konto A erhalten hat.

Schritt 2.1: Anmelden bei der AWS-Managementkonsole

Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto B zuerst als Benutzer AccountBadmin bei der AWS Management Console an.

Schritt 2.2: Erstellen des Benutzers Dave in Konto B

Erstellen Sie in der IAM-Konsole einen Benutzer Dave.

Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.

Schritt 2.3: Delegieren von Berechtigungen an den Benutzer Dave

Erstellen Sie mithilfe der folgenden Richtlinie eine Inlinerichtlinie für den Benutzer Dave. Sie müssen die Richtlinie aktualisieren, indem Sie Ihren Bucket-Namen angeben.

Es wird angenommen, dass Sie mit den Anmeldeinformationen von AccountBadmin an der Konsole angemeldet sind.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example",
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket"
         ]
      }
   ]
}
```

Weitere Informationen finden Sie unter Verwenden von eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Schritt 2.4: Testen der Berechtigungen

Jetzt kann Dave in Konto B den Inhalt von `examplebucket` auflisten, der Konto A gehört. Sie können die Berechtigungen mit einem der folgenden Verfahren überprüfen.

Test mit der AWS CLI

1. Fügen Sie der AWS CLI-Konfigurationsdatei das Profil UserDave hinzu. Weitere Informationen zur Konfigurationsdatei finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349)

```
[profile UserDave]
```

```
aws_access_key_id = access-key
aws_secret_access_key = secret-access-key
region = us-east-1
```

2. Geben Sie an der Eingabeaufforderung den folgenden AWS CLI-Befehl ein, um zu überprüfen, ob Dave jetzt eine Objektliste aus dem `examplebucket` erhält, der Konto A gehört. Beachten Sie, dass der Befehl das Profil UserDave angibt.

   ```
   aws s3 ls s3://examplebucket --profile UserDave
   ```

   Dave besitzt keine anderen Berechtigungen. Wenn er also eine andere Operation ausprobiert – z. B. die folgende Bestimmung des Bucket-Speicherorts –, gibt Amazon S3 "Berechtigung verweigert" zurück.

   ```
   aws s3api get-bucket-location --bucket examplebucket --profile UserDave
   ```

Testen mit den AWS Tools für Windows PowerShell

1. Speichern Sie die Anmeldeinformationen von Dave als AccountBDave.

   ```
   set-awscredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas
    AccountBDave
   ```

2. Probieren Sie den Befehl List Bucket aus.

   ```
   get-s3object -BucketName example2bucket -StoredCredentials AccountBDave
   ```

   Dave besitzt keine anderen Berechtigungen. Wenn er also eine andere Operation ausprobiert – z. B. die folgende Bestimmung des Bucket-Speicherorts –, gibt Amazon S3 "Berechtigung verweigert" zurück.

   ```
   get-s3bucketlocation -BucketName example2bucket -StoredCredentials AccountBDave
   ```

Schritt 3: Zusatzpunkt: Ausprobieren einer expliziten Zugriffsverweigerung

Sie können Berechtigungen über eine ACL, eine Bucket-Richtlinie oder eine Benutzerrichtlinie erhalten. Wenn es jedoch eine explizite Zugriffsverweigerung gibt, die entweder über eine Bucket-Richtlinie oder ein Benutzerprofil festgelegt wurde, hat die explizite Zugriffsverweigerung Vorrang gegenüber allen anderen Berechtigungen. Für den Test aktualisieren wir die Bucket-Richtlinie und verweigern Konto B explizit die `s3:ListBucket`-Berechtigung. Die Richtlinie erteilt auch die `s3:ListBucket`-Berechtigung, aber die explizite Zugriffsverweigerung hat Vorragen, und Konto B oder die Benutzer in Konto B können keine Objekte in `examplebucket` auflisten.

1. Ersetzen Sie unter Verwendung der Anmeldeinformationen von Benutzer AccountAadmin in Konto A die Bucket-Richtlinie durch Folgendes.

   ```
   {
      "Version": "2012-10-17",
      "Statement": [
         {
            "Sid": "Example permissions",
            "Effect": "Allow",
            "Principal": {
               "AWS": "arn:aws:iam::AccountB-ID:root"
            },
   ```

```
         "Action": [
            "s3:GetBucketLocation",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket"
         ]
      },
      {
         "Sid": "Deny permission",
         "Effect": "Deny",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket"
         ]
      }
   ]
}
```

2. Wenn Sie jetzt versuchen, über die Anmeldeinformationen von AccountBadmin eine Bucket-Liste zu erhalten, wird Ihnen eine Zugriffsverweigerung gemeldet.
   - Verwenden der AWS-CLI:

     ```
     aws s3 ls s3://examplebucket --profile AccountBadmin
     ```

   - Verwendung der AWS Tools für Windows PowerShell:

     ```
     get-s3object -BucketName example2bucket -StoredCredentials AccountBDave
     ```

Schritt 4: Bereinigen

1. Nachdem Sie mit den Tests fertig sind, räumen Sie wie folgt auf.
   - Melden Sie sich mit den Anmeldeinformationen von Konto A an der AWS Management Console (AWS Management Console) an und machen Sie Folgendes:
     - Entfernen Sie in der Amazon S3-Konsole die *examplebucket* zugeordnete Bucket-Richtlinie. Löschen Sie in den Bucket Properties die Richtlinie im Abschnitt Permissions.
     - Wenn der Bucket für diese Übung erstellt wurde, löschen Sie in der Amazon S3-Konsole die Objekte und löschen dann den Bucket.
     - Entfernen Sie in der IAM-Konsole den Benutzer AccountAadmin.
2. Melden Sie sich mit den Anmeldeinformationen von Konto B an der AWS Management Console (AWS Management Console) an. Löschen Sie in der IAM-Konsole den Benutzer AccountBadmin.

Beispiel 3: Bucket-Eigentümer, der seinen Benutzern Berechtigungen für Objekte erteilt, die ihm nicht gehören

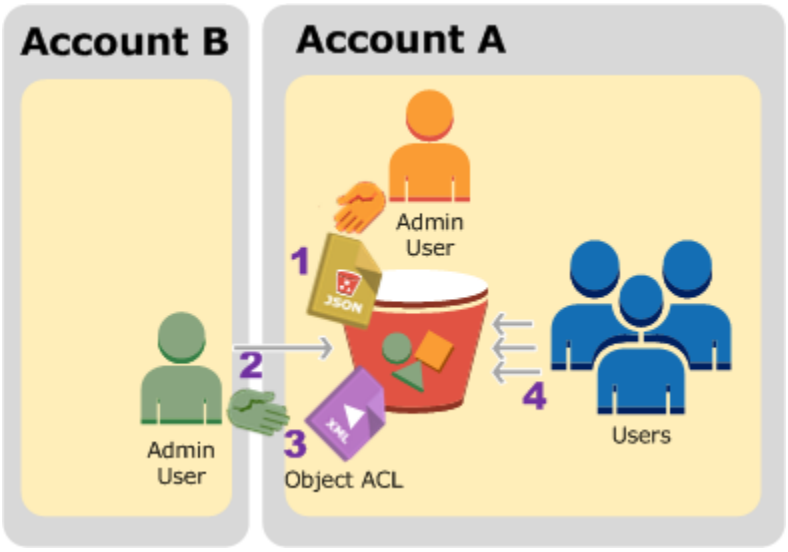
Themen

- Schritt 2: Erledigen der Aufgaben von Konto B (p. 365)
- Schritt 3: Testen der Berechtigungen (p. 365)
- Schritt 4: Bereinigen (p. 366)

In dem Szenario für dieses Beispiel will ein Bucket-Eigentümer Berechtigungen für den Zugriff auf Objekte erteilen, aber nicht alle Objekte im Bucket gehören dem Bucket-Eigentümer. Wie kann ein Bucket-Eigentümer Berechtigungen für Objekte erteilen, die ihm nicht gehören? In diesem Beispiel versucht der Bucket-Eigentümer, Benutzern in seinem eigenen Konto eine Berechtigung zu erteilen.

Ein Bucket-Eigentümer kann anderen AWS-Konten ermöglichen, Objekte hochzuladen. Diese Objekte gehören den Konten, die sie erstellt haben. Der Bucket-Eigentümer hat keine eigenen Objekte, die nicht vom Bucket-Eigentümer erstellt wurden. Aus diesem Grund muss der Objekteigentümer, damit der Bucket-Eigentümer Zugriff auf diese Objekte erteilen kann, zuerst dem Bucket-Eigentümer über eine Objekt-ACL eine Berechtigung erteilen. Weitere Informationen finden Sie unter Eigentümerschaft an Amazon S3 Buckets und Objekten (p. 331).

In diesem Beispiel delegiert der Bucket-Eigentümer Benutzern in seinem eigenen Konto eine Berechtigung. Nachfolgend finden Sie eine kurze Zusammenfassung der wichtigsten Details:



1. Der Administrator-Benutzer von Konto A ordnet eine Bucket-Richtlinie mit zwei Anweisungen zu.
   - Kontoübergreifende Berechtigung für Konto B, um Objekte hochzuladen.
   - Berechtigung für einen Benutzer in seinem eigenen Konto, auf Objekte im Bucket zuzugreifen.
2. Der Administrator-Benutzer für Konto B lädt Objekte in den Bucket hoch, der Konto A gehört.
3. Der Administrator von Konto B aktualisiert die Objekt-ACL, indem er die Berechtigung hinzufügt, die dem Bucket-Eigentümer vollständige Berechtigungen für das Objekt erteilt.
4. Der Benutzer in Konto A überprüft dies durch Zugriff auf Objekte im Bucket, unabhängig davon, wem diese gehören.

Für dieses Beispiel benötigen Sie zwei Konten. Die folgende Tabelle zeigt, wie wir auf diese Konten und die Administrator-Benutzer in diesen Konten verweisen. Laut der IAM-Richtlinien (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen (p. 348)) verwenden wir in dieser schrittweisen Anleitung nicht die Root-Anmeldeinformationen für das Konto. Stattdessen erstellen Sie einen Administrator-Benutzer in jedem Konto, und verwenden dessen Anmeldeinformationen, um Ressourcen zu erstellen und ihnen Berechtigungen zu erteilen.

AWS-Konto-ID	Konto wird bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin
2222-2222-2222	Konto B	AccountBadmin

Alle Aufgaben in Verbindung mit dem Erstellen von Benutzern und Gewähren von Berechtigungen werden in der AWS Management Console ausgeführt. Um die Berechtigungen zu überprüfen, verwendet das detaillierte Beispiel die Befehlszeilen-Tools, AWS Command Line Interface (CLI) und AWS Tools for Windows PowerShell, sodass Sie keinen Code zu schreiben brauchen.

Schritt 0: Vorbereitung auf den Walkthrough

1. Stellen Sie sicher, dass Sie zwei AWS-Konten haben, und dass jedes Konto einen Administrator hat, wie in der Tabelle im vorigen Abschnitt gezeigt.

   a. Melden Sie sich nach Bedarf für ein AWS-Konto an.

      i. Öffnen Sie https://aws.amazon.com/s3/ und wählen Sie Create an AWS Account (Erstellen eines AWS-Kontos) aus.

      ii. Folgen Sie den Anweisungen auf dem Bildschirm. Sie werden per E-Mail von AWS benachrichtigt, sobald Ihr Konto aktiv ist und von Ihnen verwendet werden kann.

   b. Melden Sie sich mit den Anmeldeinformationen für Konto A an der IAM-Konsole an und erstellen Sie wie folgt einen Administrator-Benutzer:

      - Erstellen Sie den Benutzer AccountAadmin und schreiben Sie sich die Sicherheitsanmeldeinformationen auf. Weitere Informationen zum Hinzufügen von Benutzern finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.
      - Erteilen Sie dem Administrator AccountAadmin Berechtigungen, indem Sie ihm eine Benutzerrichtlinie zuordnen, die ihm vollen Zugriff bietet. Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.
      - Schreiben Sie sich auf dem Dashboard der IAM-Konsole die URL für die Anmeldung des IAM-Benutzers auf. Benutzer in diesem Konto müssen diese URL für die Anmeldung an der AWS Management Console verwenden. Weitere Informationen finden Sie unter Wie sich Benutzer bei Ihrem Konto anmelden im IAM-Benutzerhandbuch.

   c. Wiederholen Sie den obigen Schritt unter Verwendung der Anmeldeinformationen von Konto B und erstellen Sie den Administrator-Benutzer AccountBadmin.

2. Richten Sie die AWS CLI (Command Line Interface) oder die AWS Tools for Windows PowerShell ein. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

   - Wenn Sie die AWS CLI verwenden, erstellen Sie zwei Profile, AccountAadmin und AccountBadmin, in der Konfigurationsdatei.
   - Wenn Sie die AWS Tools for Windows PowerShell verwenden, stellen Sie sicher, dass sie die Anmeldeinformationen für die Sitzung als AccountAadmin und AccountBadmin speichern.

   Detaillierte Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

Schritt 1: Erledigen der Aufgaben von Konto A

Schritt 1.1: Anmelden bei der AWS-Managementkonsole

Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto A zuerst als Benutzer AccountAadmin bei der AWS Management Console an. Dieser Benutzer erstellt einen Bucket und ordnet ihm eine Richtlinie zu.

Schritt 1.2: Erstellen eines Buckets, eines Benutzers und Hinzufügen einer Bucket-Richtlinie, die Benutzerberechtigungen erteilt

1. Erstellen Sie in der Amazon S3-Konsole einen Bucket. Diese Übung geht davon aus, dass der Bucket in der Region USA Ost (Nord-Virginia) erstellt wurde und `examplebucket` heißt.

   Weitere Informationen finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
2. Erstellen Sie in der IAM-Konsole einen Benutzer Dave.

   Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.
3. Schreiben Sie sich die Anmeldeinformationen für Dave auf.
4. Ordnen Sie in der Amazon S3-Konsole dem `examplebucket`-Bucket die folgende Bucket-Richtlinie zu. Weitere Informationen finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service. Folgen Sie den Schritten, um eine Bucket-Richtlinie hinzuzufügen. Informationen darüber, wie Sie Konto-IDs finden, finden Sie unter Finden Ihrer AWS-Konto-ID.

   Die Richtlinie erteilt Konto B die Berechtigung `s3:PutObject` und `s3:ListBucket`. Die Richtlinie erteilt außerdem dem Benutzer Dave die `s3:GetObject`-Berechtigung.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Statement1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:PutObject",
            "s3:ListBucket"
         ],
         "Resource": [
         "arn:aws:s3:::examplebucket/*"
   "arn:aws:s3:::examplebucket"
         ]
      },
      {
         "Sid": "Statement3",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
            "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::examplebucket/*"
         ]
```

```
      }
   ]
}
```

Schritt 2: Erledigen der Aufgaben von Konto B

Nachdem Konto A die Berechtigung besitzt, Operationen für den Bucket von Konto A auszuführen, erledigt der Administrator von Konto B Folgendes:

- Hochladen eines Objekts in den Bucket von Konto A.
- Fügen Sie eine Berechtigung in der Objekt-ACL hinzu, um Account A, dem Bucket-Eigentümer, die volle Kontrolle zu ermöglichen.

Verwenden der AWS CLI

1. Hochladen eines Objekts mit dem AWS CLI-Befehl `put-object`. Der Parameter `-body` im Befehl identifiziert die hochzuladende Quelldatei. Befindet sich die Datei beispielsweise auf dem Laufwerk `C:` eines Windows-Computers, geben Sie `c:\HappyFace.jpg` an. Der Parameter `--key` gibt den Schlüsselnamen für das Objekt an.

   ```
   aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body HappyFace.jpg --
   profile AccountBadmin
   ```

2. Erteilung einer Berechtigung in der Objekt-ACL, um dem Bucket-Eigentümer volle Kontrolle über das Objekt zu erteilen. Informationen darüber, wie Sie eine kanonische Benutzer-ID finden, finden Sie unter Finden Ihrer kanonischen Benutzer-ID für das Konto.

   ```
   aws s3api put-object-acl --bucket examplebucket --key HappyFace.jpg --grant-full-
   control id="AccountA-CanonicalUserID" --profile AccountBadmin
   ```

Verwendung der AWS Tools für Windows PowerShell

1. Verwendung des Befehls `Write-S3Object` der AWS-Tools für Windows PowerShell, um ein Objekt hochzuladen.

   ```
   Write-S3Object -BucketName examplebucket -key HappyFace.jpg -file HappyFace.jpg -
   StoredCredentials AccountBadmin
   ```

2. Erteilung einer Berechtigung in der Objekt-ACL, um dem Bucket-Eigentümer volle Kontrolle über das Objekt zu erteilen.

   ```
   Set-S3ACL -BucketName examplebucket -Key HappyFace.jpg -CannedACLName "bucket-owner-
   full-control" -StoredCreden
   ```

Schritt 3: Testen der Berechtigungen

Jetzt kann der Benutzer Dave in Konto A überprüfen, ob er Zugriff auf das Objekt hat, das Konto B gehört.

Verwenden der AWS CLI

1. Fügen Sie die Anmeldeinformationen von Benutzer Dave der AWS CLI-Konfigurationsdatei hinzu, und erstellen Sie ein neues Profil, `UserDaveAccountA`. Weitere Informationen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

```
[profile UserDaveAccountA]
aws_access_key_id = access-key
aws_secret_access_key = secret-access-key
region = us-east-1
```

2. Führen Sie den AWS CLI-Befehl `get-object` aus, um `HappyFace.jpg` herunterzuladen und es lokal zu speichern. Sie stellen dem Benutzer Dave Anmeldeinformationen bereit, indem Sie den Parameter `--profile` hinzufügen.

```
aws s3api get-object --bucket examplebucket --key HappyFace.jpg Outputfile.jpg --
profile UserDaveAccountA
```

Verwendung der AWS Tools für Windows PowerShell

1. Speichern Sie die AWS-Anmeldeinformationen von Benutzer Dave als UserDaveAccountA im persistenten Speicher.

```
Set-AWSCredentials -AccessKey UserDave-AccessKey -SecretKey UserDave-SecretAccessKey -
storeas UserDaveAccountA
```

2. Führen Sie den Befehl Read-S3Object aus, um das Objekt `HappyFace.jpg` herunterzuladen und es lokal zu speichern. Sie stellen dem Benutzer Dave Anmeldeinformationen bereit, indem Sie den Parameter `-StoredCredentials` hinzufügen.

```
Read-S3Object -BucketName examplebucket -Key HappyFace.jpg -file HappyFace.jpg  -
StoredCredentials UserDaveAccountA
```

Schritt 4: Bereinigen

1. Nachdem Sie mit den Tests fertig sind, räumen Sie wie folgt auf.
   - Melden Sie sich mit den Anmeldeinformationen von Konto A an der AWS Management Console (AWS Management Console) an und machen Sie Folgendes:
     - Entfernen Sie in der Amazon S3-Konsole die *examplebucket* zugeordnete Bucket-Richtlinie. Löschen Sie in den Bucket Properties die Richtlinie im Abschnitt Permissions.
     - Wenn der Bucket für diese Übung erstellt wurde, löschen Sie in der Amazon S3-Konsole die Objekte und löschen dann den Bucket.
     - Entfernen Sie in der IAM-Konsole den Benutzer AccountAadmin.
2. Melden Sie sich mit den Anmeldeinformationen von Konto B an der AWS Management Console (AWS Management Console) an. Löschen Sie in der IAM-Konsole den Benutzer AccountBadmin.

Beispiel 4: Der Bucket-Eigentümer erteilt eine kontenübergreifende Berechtigung für Objekte, die ihm nicht gehören

Themen

In diesem Beispielszenario sind Sie Eigentümer eines Buckets und haben andere AWS-Konten aktiviert, um Objekte hochzuladen. Das bedeutet, Ihr Bucket kann Objekte enthalten, die anderen AWS-Konten gehören.

Angenommen, Sie müssen als Bucket-Eigentümer einem Benutzer in einem anderen Konto eine kontenübergreifende Berechtigung für Objekte erteilen, unabhängig davon, wer der Eigentümer ist. Dieser Benutzer könnte beispielsweise eine Buchhaltungsanwendung sein, die Zugriff auf Objekt-Metadaten benötigt. Es gibt zwei Kernprobleme:

- Der Bucket-Eigentümer hat keine Berechtigungen für diese Objekte, die von anderen AWS-Konten erstellt wurden. Damit der Bucket-Eigentümer Berechtigungen für Objekte erteilen kann, die ihm nicht gehören, muss der Objekteigentümer, nämlich das AWS-Konto, das die Objekte erstellt hat, zuerst dem Bucket-Eigentümer die Berechtigung erteilen. Der Bucket-Eigentümer kann diese Berechtigungen delegieren.
- Das Konto des Bucket-Eigentümers kann Berechtigungen an Benutzer in seinem eigenen Konto delegieren (siehe Beispiel 3: Bucket-Eigentümer, der seinen Benutzern Berechtigungen für Objekte erteilt, die ihm nicht gehören (p. 361)), aber es kann keine Berechtigungen für andere AWS-Konten delegieren, weil keine kontenübergreifende Delegation unterstützt wird.

In diesem Szenario kann der Bucket-Eigentümer eine AWS Identity and Access Management (IAM)-Rolle erstellen, mit der Berechtigung zum Zugriff auf Objekte, und einem anderen AWS-Konto die Berechtigung erteilen, die Rolle vorübergehend zu übernehmen, sodass es Zugriff auf Objekte in dem Bucket erhält.

Hintergrund: Kontoübergreifende Berechtigungen und die Verwendung von IAM-Rollen

IAM-Rollen unterstützen verschiedene Szenarien, den Zugriff auf Ihre Ressourcen zu definieren. Der kontenübergreifende Zugriff ist eines der Schlüsselszenarien. In diesem Beispiel verwendet der Bucket-Eigentümer, Konto A, eine IAM-Rolle, um den Objektzugriff temporär kontenübergreifend an Benutzer in einem anderen AWS-Konto, Konto C, zu delegieren. Jeder von Ihnen erstellten IAM-Rolle sind zwei Richtlinien zugeordnet:

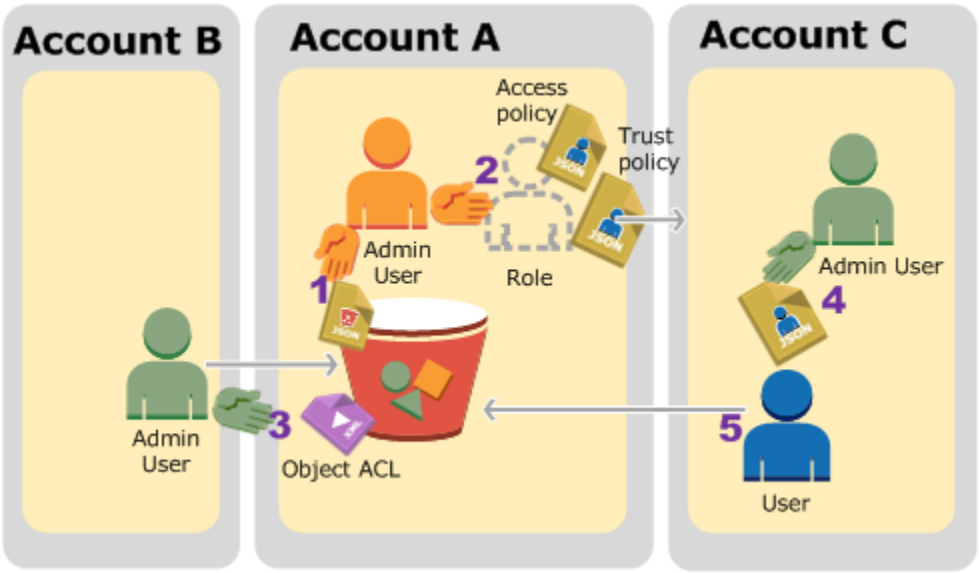
- Eine Vertrauensrichtlinie, die ein anderes AWS-Konto identifiziert, das die Rolle einnehmen darf.
- Eine Zugriffsrichtlinie, die definiert, welche Berechtigungen – z. B. `s3:GetObject` – zulässig sind, wenn jemand die Rolle einnimmt. Eine Liste aller Berechtigungen, die Sie in einer Richtlinie angeben können, finden Sie unter Amazon S3-Aktionen (p. 380).

Das in der Vertrauensrichtlinie identifizierte AWS-Konto erteilt dann seinem Benutzer die Berechtigung, die Rolle zu übernehmen. Der Benutzer kann dann wie folgt auf Objekte zugreifen:

- Die Rolle einnehmen und daraufhin temporäre Sicherheitsanmeldeinformationen erhalten.
- Verwenden der temporären Sicherheitsanmeldeinformationen, um auf die Objekte im Bucket zuzugreifen.

Weitere Informationen zu IAM-Rollen finden Sie unter IAM Roles im IAM-Benutzerhandbuch.

Nachfolgend finden Sie eine kurze Zusammenfassung der wichtigsten Details:



1. Der Administrator-Benutzer von Konto A ordnet eine Bucket-Richtlinie zu, die Konto B die bedingte Berechtigung erteilt, Objekte hochzuladen.
2. Der Administrator von Konto A erstellt eine IAM-Rolle, die eine Vertrauensbeziehung zu Konto C einrichtet, sodass Benutzer in diesem Konto auf Konto A zugreifen können. Die der Rolle zugeordnete Zugriffsrichtlinie beschränkt die Aktionen, die der Benutzer in Konto C machen kann, wenn er auf Konto A zugreift.
3. Der Administrator von Konto B lädt ein Objekt in den Bucket hoch, der Konto A gehört, und erteilt dem Bucket-Eigentümer vollständige Berechtigungen.
4. Der Administrator von Konto C erstellt einen Benutzer und ordnet ihm eine Benutzerrichtlinie zu, die dem Benutzer gestattet, die Rolle zu übernehmen.
5. Der Benutzer in Konto C übernimmt zuerst die Rolle, womit er temporäre Sicherheitsanmeldeinformationen erhält. Unter Verwendung dieser temporären Sicherheitsanmeldeinformationen greift der Benutzer dann auf die Objekte im Bucket zu.

Für dieses Beispiel benötigen Sie drei Konten. Die folgende Tabelle zeigt, wie wir auf diese Konten und die Administrator-Benutzer in diesen Konten verweisen. Laut der IAM-Richtlinien (siehe Informationen zur Verwendung eines Administratorbenutzers zum Erstellen von Ressourcen und Erteilen von Berechtigungen (p. 348)) verwenden wir in dieser schrittweisen Anleitung nicht die Root-Anmeldeinformationen für das Konto. Stattdessen erstellen Sie einen Administrator-Benutzer in jedem Konto, und verwenden dessen Anmeldeinformationen, um Ressourcen zu erstellen und ihnen Berechtigungen zu erteilen.

AWS-Konto-ID	Konto bezeichnet als	Administratorbenutzer im Konto
1111-1111-1111	Konto A	AccountAadmin
2222-2222-2222	Konto B	AccountBadmin
3333-3333-3333	Konto C	AccountCadmin

Schritt 0: Vorbereitung auf den Walkthrough

Note

Öffnen Sie gegebenenfalls einen Texteditor und schreiben Sie sich die Informationen innerhalb der einzelnen Schritte auf. Insbesondere brauchen Sie Konten-IDs, kanonische Benutzer-IDs, URLs für die Anmeldung von IAM-Benutzern für jedes Konto, das mit der Konsole verbunden wird, sowie die ARNs (Amazon Resource Names) der IAM-Benutzer und -Rollen.

1. Stellen Sie sicher, dass Sie drei AWS-Konten haben, und dass jedes Konto einen Administrator hat, wie in der Tabelle im vorigen Abschnitt gezeigt.

   a. Melden Sie sich nach Bedarf an den AWS-Konten an. Wir bezeichnen diese Konten als Konto A, Konto B und Konto C.

      i. Öffnen Sie https://aws.amazon.com/s3/ und wählen Sie Create an AWS Account (Erstellen eines AWS-Kontos) aus.

      ii. Folgen Sie den Anweisungen auf dem Bildschirm.

      Sie werden per E-Mail von AWS benachrichtigt, sobald Ihr Konto aktiv ist und von Ihnen verwendet werden kann.

   b. Melden Sie sich mit den Anmeldeinformationen für Konto A an der IAM-Konsole an und erstellen Sie wie folgt einen Administrator-Benutzer:

      - Erstellen Sie den Benutzer AccountAadmin und schreiben Sie sich die Sicherheitsanmeldeinformationen auf. Weitere Informationen zum Hinzufügen von Benutzern finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.
      - Erteilen Sie dem Administrator AccountAadmin Berechtigungen, indem Sie ihm eine Benutzerrichtlinie zuordnen, die ihm vollen Zugriff bietet. Weitere Informationen finden Sie unter Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.
      - Schreiben Sie sich auf dem Dashboard der IAM-Konsole die URL für die Anmeldung des IAM-Benutzers auf. Benutzer in diesem Konto müssen diese URL für die Anmeldung an der AWS Management Console verwenden. Weitere Informationen finden Sie unter Wie sich Benutzer bei Ihrem Konto anmelden im IAM-Benutzerhandbuch.

   c. Wiederholen Sie den obigen Schritt, um Administrator-Benutzer in Konto B und Konto C zu erstellen.

2. Notieren Sie für Konto C die kanonische Benutzer-ID.

   Wenn Sie eine IAM-Rolle in Konto A erstellen, erteilt die Vertrauensrichtlinie Konto C die Berechtigung, die Rolle durch Angabe der Konto-ID zu übernehmen. Sie finden die Konteninformationen wie folgt:

   a. Verwenden Sie Ihre AWS-Konto-ID oder Ihren Kontoalias, Ihren IAM-Benutzernamen und Ihr Passwort, um sich bei der Amazon S3-Konsole anzumelden.

   b. Wählen Sie den Namen eines Amazon S3-Buckets aus, um die Details zu diesem Bucket anzuzeigen.

   c. Klicken Sie auf die Registerkarte Berechtigungen und anschließend auf Zugriffskontrollliste.

   d. Im Abschnitt Zugriff für Ihr AWS-Konto in der Spalte Konto befindet sich eine lange Kennung, z. B. `c1daexampleaaf850ea79cf0430f33d72579fd1611c97f7ded193374c0b163b6`. Dies ist Ihre kanonische Benutzer-ID.

3. Für das Erstellen einer Bucket-Richtlinie benötigen Sie die folgenden Informationen. Schreiben Sie sich diese Werte auf:

   - Kanonische Benutzer-ID von Konto A – Wenn der Administrator von Konto A dem Administrator von Konto B die bedingte Berechtigung erteilt, Objekte hochzuladen, gibt die Bedingung die kanonische Benutzer-ID des Benutzers von Konto A an, der vollständige Kontrolle über die Objekte benötigt.

Note

Die kanonische Benutzer-ID ist das Konzept nur für Amazon S3. Es handelt sich dabei um eine 64 Zeichen lange verschleierte Version der Konto-ID.

- Benutzer-ARN für den Administrator von Konto B – Die Benutzer-ARN finden Sie in der IAM-Konsole. Wählen Sie den Benutzer aus. Sie finden die ARN des Benutzers auf der Registerkarte Summary.

  In der Bucket-Richtlinie erteilen Sie AccountBadmin die Berechtigung, Objekte hochzuladen, und Sie geben unter Verwendung des ARN den Benutzer an. Ein Beispiel für einen ARN-Wert:

```
arn:aws:iam::AccountB-ID:user/AccountBadmin
```

4. Richten Sie die AWS CLI (Command Line Interface) oder die AWS Tools for Windows PowerShell ein. Stellen Sie sicher, dass Sie die Anmeldeinformationen speichern, wie folgt:

   - Wenn Sie die AWS CLI verwenden, erstellen Sie die Profile, AccountAadmin und AccountBadmin, in der Konfigurationsdatei.
   - Wenn Sie die AWS Tools for Windows PowerShell verwenden, stellen Sie sicher, dass sie die Anmeldeinformationen für die Sitzung als AccountAadmin und AccountBadmin speichern.

   Detaillierte Anweisungen finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

Schritt 1: Erledigen der Aufgaben von Konto A

In diesem Beispiel ist Konto A der Bucket-Eigentümer. Der Benutzer AccountAadmin in Konto A erstellt also einen Bucket, ordnet eine Bucket-Richtlinie zu, die dem Administrator von Konto B die Berechtigung zum Hochladen von Objekten erteilt, erstellt eine IAM-Rolle, die Konto C die Berechtigung erteilt, die Rolle zu übernehmen, so dass es auf Objekte im Bucket zugreifen kann.

Schritt 1.1: Anmelden bei der AWS-Managementkonsole

Melden Sie sich unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto A zuerst als Benutzer AccountAadmin bei der AWS Management Console an. Dieser Benutzer erstellt einen Bucket und ordnet ihm eine Richtlinie zu.

Schritt 1.2: Erstellen eines Buckets und Anfügen einer Richtlinie

Führen Sie in der Amazon S3-Konsole die folgenden Schritte aus:

1. Erstellen Sie einen Bucket. Diese Übung setzt voraus, dass der Bucket-Name `examplebucket` ist.

   Weitere Informationen finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
2. Ordnen Sie die folgende Bucket-Richtlinie zu, die dem Administrator von Konto B die bedingte Berechtigung erteilt, Objekte hochzuladen.

   Sie müssen die Richtlinie aktualisieren, indem Sie Ihre eigenen Werte für *examplebucket*, *AccountB-ID* und *CanonicalUserId-of-AWSaccountA-BucketOwner* angeben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "111",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::examplebucket/*"
        },
        {
            "Sid": "112",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::examplebucket/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-grant-full-control": "id=CanonicalUserId-of-AWSaccountA-
BucketOwner"
                }
            }
        }
    ]
}
```

Schritt 1.3: Erstellen einer IAM-Rolle, um Konto C kontenübergreifenden Zugriff in Konto A zu erteilen

Erstellen Sie in der IAM-Konsole eine IAM-Rolle ("examplerole"), die Konto C die Berechtigung erteilt, die Rolle zu übernehmen. Stellen Sie sicher, dass sie noch als Administrator von Konto A angemeldet sind, weil die Rolle in Konto A erstellt werden muss.

1. Bevor Sie die Rolle erstellen, richten Sie die verwaltete Richtlinie ein, die die von der Rolle benötigten Berechtigungen definiert. Diese Richtlinie fügen Sie zu einem späteren Zeitpunkt der Rolle an.

   a. Klicken Sie im Navigationsbereich links auf Policies und dann auf Create Policy.

   b. Klicken Sie neben Create Your Own Policy auf Select.

   c. Geben Sie `access-accountA-bucket` in das Feld Policy Name ein.

   d. Kopieren Sie die folgende Zugriffsrichtlinie und fügen Sie sie in das Feld Policy Document ein. Die Zugriffsrichtlinie erteilt der Rolle die Berechtigung `s3:GetObject`, wenn also der Benutzer von Konto C die Rolle übernimmt, kann er nur die Operation `s3:GetObject` ausführen.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::examplebucket/*"
       }
     ]
   }
   ```

   e. Klicken Sie auf Create Policy.

   Die neue Richtlinie wird in der Liste der verwalteten Richtlinien angezeigt.

2. Klicken Sie im Navigationsbereich links auf Roles und dann auf Create New Role.
3. Wählen Sie unter Select Role Type Role for Cross-Account Access aus und klicken Sie auf die Schaltfläche Select neben Provide access between AWS accounts you own.
4. Geben Sie die Konto-ID von Konto C ein.

In dieser Schritt-für-Schritt-Anleitung müssen Sie noch nicht die Multi-Factor Authentication (MFA) von den Benutzern verlangen, um die Rolle zu übernehmen. Aktivieren Sie daher diese Option noch nicht.

5. Klicken Sie auf Next Step, um die mit der Rolle verknüpften Berechtigungen einzurichten.
6. Aktivieren Sie das Kontrollkästchen neben der zuvor erstellten Richtlinie `access-accountA-bucket` und klicken Sie auf Next Step.

   Die Prüfseite wird angezeigt, sodass Sie die Einstellungen bestätigen können, bevor Sie die Rolle erstellen. Ein sehr wichtiges, auf dieser Seite zu beachtendes Element ist der Link, den Sie an die Benutzer senden können, die die Rolle verwenden müssen. Die Benutzer, die auf den Link klicken, werden direkt zur Seite Switch Role geleitet, in der die Felder Account ID und Role Name bereits ausgefüllt sind. Dieser Link wird auch auf der Seite Role Summary für beliebige kontoübergreifende Rollen angezeigt.
7. Geben Sie `examplerole` für den Rollennamen ein und klicken Sie dann auf Next Step.
8. Nachdem Sie die Rolle überprüft haben, klicken Sie auf Create Role.

   Die Rolle `examplerole` wird in der Liste der Rollen angezeigt.
9. Klicken Sie auf den Rollennamen `examplerole`.
10. Wählen Sie die Registerkarte Trust Relationships.
11. Klicken Sie auf Show policy document und überprüfen Sie, ob die angezeigte Vertrauensrichtlinie mit der folgenden Richtlinie übereinstimmt.

    Die folgende Vertrauensrichtlinie richtet eine Vertrauensbeziehung zu Konto C ein und gestattet ihm die Aktion `sts:AssumeRole`. Weitere Informationen finden Sie unter AssumeRole in der AWS Security Token Service API Reference.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountC-ID:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

12. Schreiben Sie sich den Amazon-Ressourcennamen (ARN) der von Ihnen erstellten Rolle `examplerole` auf.

    In den nachfolgenden Schritten ordnen Sie eine Benutzerrichtlinie ein, um einem IAM-Benutzer zu erlauben, diese Rolle einzunehmen. Sie identifizieren die Rolle über den ARN-Wert.

Schritt 2: Erledigen der Aufgaben von Konto B

Der Beispiel-Bucket, der Konto A gehört, benötigt Objekte, die anderen Konten gehören. In diesem Schritt lädt der Administrator von Konto B unter Verwendung der Befehlszeilen-Tools ein Objekt hoch.

- Laden Sie unter Verwendung des AWS CLI-Befehls put-object ein Objekt in den `examplebucket` hoch.

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body HappyFace.jpg --
grant-full-control id="canonicalUserId-ofTheBucketOwner" --profile AccountBadmin
```

Beachten Sie Folgendes:

- Der Parameter `--Profile` gibt das Profil AccountBadmin an, das Objekt gehört also Konto B.
- Der Parameter `grant-full-control` erteilt dem Bucket-Eigentümer vollständige Berechtigungen für das Objekt, wie für die Bucket-Richtlinie erforderlich.
- Der Parameter `--body` identifiziert die hochzuladende Quelldatei. Befindet sich die Datei beispielsweise auf dem Laufwerk C: eines Windows-Computers, geben Sie `c:\HappyFace.jpg` an.

Schritt 3: Erledigen der Aufgaben von Konto C

In den vorigen Schritten hat Konto A bereits eine Rolle erstellt, `examplerole`, die eine Vertrauensbeziehung zu Konto C einrichtet. Aus diesem Grund können die Benutzer in Konto C auf Konto A zugreifen. In diesem Schritt erstellt der Administrator von Konto C einen Benutzer (Dave) und delegiert die Berechtigung `sts:AssumeRole`, die er von Konto A erhalten hat, an ihn. Damit kann Dave `examplerole` übernehmen und erhält temporären Zugriff auf Konto A. Die Zugriffsrichtlinie, die Konto A der Rolle zugeordnet hat, beschränkt die Aktionen, die Dave ausführen kann, wenn er auf Konto A zugreift – insbesondere, Objekte in `examplebucket` hochzuladen.

Schritt 3.1: Erstellen eines Benutzers in Konto C und Delegieren der Berechtigung, `examplerole` anzunehmen

1. Melden Sie sich zuerst unter Verwendung der URL für die Anmeldung als IAM-Benutzer für Konto C als Benutzer AccountCadmin bei der AWS Management Console an.
2. Erstellen Sie in der IAM-Konsole einen Benutzer Dave.

   Detaillierte Anweisungen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch.
3. Schreiben Sie sich die Anmeldeinformationen für Dave auf. Dave benötigt diese Anmeldeinformationen, um die Rolle `examplerole` zu übernehmen.
4. Erstellen Sie eine interne Richtlinie für den IAM-Benutzer Dave, um die Berechtigung `sts:AssumeRole` für Dave für die Rolle `examplerole` in Konto A zu delegieren.

   a. Klicken Sie im linken Navigationsbereich auf Users.
   b. Klicken Sie auf den Benutzernamen Dave.
   c. Wählen Sie auf der Seite mit den Benutzerinformationen die Registerkarte Permissions aus und erweitern Sie den Abschnitt Inline Policies.
   d. Wählen Sie hier klicken (oder Create User Policy).
   e. Klicken Sie auf Custom Policy und dann auf Select.
   f. Geben Sie einen Namen für die Richtlinie in das Feld Policy Name ein.
   g. Kopieren Sie die folgende Richtlinie in das Feld Policy Document:.

      Sie müssen die Richtlinie aktualisieren, indem Sie die ID von Konto A angeben.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole"],
      "Resource": "arn:aws:iam::AccountA-ID:role/examplerole"
    }
  ]
}
```

h. Klicken Sie auf Apply Policy.

5. Speichern Sie die Anmeldeinformationen von Dave in der Konfigurationsdatei der AWS CLI, indem Sie ein weiteres Profil hinzufügen, AccountCDave.

```
[profile AccountCDave]
aws_access_key_id = UserDaveAccessKeyID
aws_secret_access_key = UserDaveSecretAccessKey
region = us-west-2
```

Schritt 3.2: Übernehmen der Rolle (examplerole) und Zugreifen auf Objekte

Jetzt kann Dave auf Objekte in dem Bucket zugreifen, der Konto A gehört, nämlich wie folgt:

- Dave übernimmt zuerst die `examplerole` unter Verwendung seiner eigenen Anmeldeinformationen. Damit werden temporäre Anmeldeinformationen zurückgegeben.
- Unter Verwendung der temporären Anmeldeinformationen greift Dave dann auf die Objekte im Bucket von Konto A zu.

1. Führen Sie in der Befehlszeile den folgenden AWS CLI-Befehl `assume-role` aus. Verwenden Sie dazu das Profil AccountCDave.

   Sie müssen in dem Befehl den ARN-Wert aktualisieren, indem Sie die ID von Konto A angeben, in dem `examplerole` definiert ist.

```
aws sts assume-role --role-arn arn:aws:iam::accountA-ID:role/examplerole --profile
 AccountCDave --role-session-name test
```

   Der AWS STS (Security Token Service) wiederum gibt temporäre Sicherheits-Anmeldeinformationen zurück (Zugriffsschlüssel-ID, geheimen Zugriffsschlüssel und ein Sitzungstoken).

2. Speichern Sie die temporären Sicherheitsanmeldeinformationen in der AWS CLI-Konfigurationsdatei unter dem Profil `TempCred`.

```
[profile TempCred]
aws_access_key_id = temp-access-key-ID
aws_secret_access_key = temp-secret-access-key
aws_session_token = session-token
region = us-west-2
```

3. Führen Sie in der Befehlszeile den folgenden AWS CLI-Befehl aus, um auf Objekte zuzugreifen. Verwenden Sie dazu die temporären Anmeldeinformationen. Beispielsweise gibt der Befehl die Head-Objekt API an, um die Objekt-Metadaten für das Objekt `HappyFace.jpg` abzurufen.

```
aws s3api get-object --bucket examplebucket --key HappyFace.jpg SaveFileAs.jpg --
profile TempCred
```

   Die `examplerole` zugeordnete Zugriffsrichtlinie erlaubt die Aktionen, deshalb verarbeitet Amazon S3 die Anforderung. Sie können das mit jeder anderen Aktion für jedes andere Objekt im Bucket ausprobieren.

   Wenn Sie eine andere Aktion ausprobieren, z. B. `get-object-acl`, wird die Berechtigung verweigert, weil die Rolle diese Aktion nicht ausführen darf.

```
aws s3api get-object-acl --bucket examplebucket --key HappyFace.jpg --profile TempCred
```

Wir haben den Benutzer Dave verwendet, um die Rolle zu übernehmen und unter Verwendung temporärer Anmeldeinformationen auf das Objekt zuzugreifen. Es könnte auch eine Anwendung in Konto C sein, die auf Objekte in `examplebucket` zugreift. Die Anwendung kann temporäre Anmeldeinformationen erhalten, und Konto C kann die Berechtigung der Anwendung delegieren, um `examplerole` zu übernehmen.

Schritt 4: Bereinigen

1. Nachdem Sie mit den Tests fertig sind, räumen Sie wie folgt auf.
   - Melden Sie sich mit den Anmeldeinformationen von Konto A an der AWS Management Console (AWS Management Console) an und machen Sie Folgendes:
     - Entfernen Sie in der Amazon S3-Konsole die *`examplebucket`* zugeordnete Bucket-Richtlinie. Löschen Sie in den Bucket Properties die Richtlinie im Abschnitt Permissions.
     - Wenn der Bucket für diese Übung erstellt wurde, löschen Sie in der Amazon S3-Konsole die Objekte und löschen dann den Bucket.
     - Entfernen Sie in der IAM-Konsole die `examplerole`, die Sie in Konto A erstellt haben.
     - Entfernen Sie in der IAM-Konsole den Benutzer AccountAadmin.
2. Melden Sie sich mit den Anmeldeinformationen von Konto B an der AWS Management Console (AWS Management Console) an. Löschen Sie in der IAM-Konsole den Benutzer AccountBadmin.
3. Melden Sie sich mit den Anmeldeinformationen von Konto C an der AWS Management Console (AWS Management Console) an. Löschen Sie in der IAM-Konsole den Benutzer AccountCadmin und den Benutzer Dave.

Zugehörige Ressourcen

- Befolgen Sie die Schritte in Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer im IAM-Benutzerhandbuch.
- Ein Beispiel finden Sie in der Praktischen Anleitung: Delegieren des Zugriffs in allen AWS-Konten mithilfe von IAM-Rollen im IAM-Benutzerhandbuch.
- Arbeiten mit Richtlinien im IAM-Benutzerhandbuch.

Verwendung von Bucket-Richtlinien und Benutzerrichtlinien

Bucket-Richtlinien und Benutzerrichtlinien sind zwei Zugriffsrichtlinienoptionen für die Erteilung von Berechtigungen für Ihre Amazon S3 Ressourcen. Beide verwenden die JSON-basierte Zugriffsrichtliniensprache.

Die Themen in diesem Abschnitt beschreiben die Schlüsselelemente der Richtliniensprache mit Schwerpunkt auf Amazon S3–spezifischen Details und bieten Beispiele für Bucket- und Benutzerrichtlinien.

> **Important**
>
> Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.
> Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die Grundkonzepte und für Sie verfügbaren Optionen zum Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen erläutert werden. Weitere Informationen finden Sie unter Einführung in das Verwalten des Zugriffs auf Amazon S3-Ressourcen (p. 330).

Themen

Richtlinien und Berechtigungen in Amazon S3

Diese Seite bietet eine Übersicht über Bucket- und Benutzerrichtlinien in Amazon S3 und beschreibt die Basiselemente einer Richtlinie. Jedes aufgelistete Element verweist auf weitere Details zu diesem Element und auf Beispiele für die Verwendung dieses Elements.

Die vollständige Liste der Amazon S3-Aktionen, -Ressourcen und -Bedingungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Übersicht über die Richtliniensprache

In ihrer einfachsten Form enthält eine Richtlinie die folgenden Elemente:

- Ressourcen (p. 377) – Buckets, Objekte, Zugriffspunkte und Aufgaben sind die Amazon S3-Ressourcen, für die Sie Berechtigungen zulassen oder ablehnen können. In einer Richtlinie identifizieren Sie die Ressource mithilfe eines Amazon-Ressourcennamens (ARN). Weitere Informationen finden Sie unter Amazon S3-Ressourcen (p. 377).
- Aktionen (p. 380) – Amazon S3 unterstützt für jede Ressource einen Satz von Operationen. Sie identifizieren Ressourcenoperationen, die Sie zulassen (oder ablehnen) können, indem Sie Aktionsschlüsselwörter verwenden.

  Beispielsweise ermöglicht die Berechtigung `s3:ListBucket` dem Benutzer die Verwendung der Amazon S3-Operation GET Bucket (List Objects). Weitere Informationen finden Sie unter Amazon S3-Aktionen (p. 380).
- Auswirkung – Zeigt die Auswirkung, wenn ein Benutzer die spezifische Aktion anfordert; entweder allow (Zugriffserlaubnis) oder deny (Zugriffsverweigerung).

  Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten ("allow"), ist der Zugriff automatisch verweigert. Sie können den Zugriff auf eine Ressource auch explizit ablehnen. So können Sie sicherstellen, dass Benutzer nicht auf die Ressource zugreifen können, auch wenn der Zugriff durch eine andere Richtlinie erteilt wird. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Auswirkung.
- Prinzipal (p. 378) – Das Konto oder der Benutzer, das oder der Zugriff auf die Aktionen und Ressourcen in der Anweisung hat. In einer Bucket-Richtlinie ist der Prinzipal der Benutzer, das Konto, der Service oder eine andere Entität, der/die/das der Empfänger dieser Berechtigung ist. Weitere Informationen finden Sie unter Prinzipale (p. 378).
- Bedingung (p. 382) – Bedingungen für den Zeitpunkt, an dem eine Richtlinie in Kraft ist. Sie können AWS-weite Schlüssel und Amazon S3-spezifische Schlüssel verwenden, um Bedingungen in einer Amazon S3-Zugriffsrichtlinie anzugeben. Weitere Informationen finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382).

Die folgende Bucket-Beispielrichtlinie enthält die zuvor genannten Richtlinienelemente. Die Richtlinie gibt Dave, einem Benutzer im Konto *`Account-ID`*, `s3:GetObject`, `s3:GetBucketLocation` und `s3:ListBucket` Amazon S3-Berechtigungen über den `examplebucket`-Bucket.

```
{
    "Version": "2012-10-17",
    "Id": "ExamplePolicy01",
    "Statement": [
```

```
        {
            "Sid": "ExampleStatement01",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::Account-ID:user/Dave"
            },
            "Action": [
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::examplebucket/*",
                "arn:aws:s3:::examplebucket"
            ]
        }
    ]
}
```

Ausführliche Informationen zur Richtliniensprache finden Sie unter Richtlinien und Berechtigungen und IAM-JSON-Richtlinienreferenz im IAM-Benutzerhandbuch.

Amazon S3-Ressourcen

Das folgende allgemeine Format für Amazon-Ressourcennamen (ARNs) identifiziert Ressourcen in AWS:

```
arn:partition:service:region:namespace:relative-id
```

Weitere Informationen zu ARNs finden Sie unter Amazon-Ressourcennamen (ARNs) in der AWS General Reference.

Weitere Informationen zu Ressourcen finden Sie unter IAM-JSON-Richtlinienelemente: Ressource im IAM-Benutzerhandbuch.

Ein Amazon S3-ARN schließt die AWS-Region und den Namespace aus, enthält jedoch Folgendes:

- Partition - `aws` ist ein allgemeiner Partitionsname. Befinden sich Ihre Ressourcen in der Region China (Peking), ist `aws-cn` der Partitionsname.
- Service - `s3`.
- Relative ID - `bucket-name` oder einen `bucket-name/object-key`. Sie können Platzhalter verwenden.

Das ARN-Format für Amazon S3-Ressourcen wird wie folgt reduziert:

```
arn:aws:s3:::bucket_name/key_name
```

Die vollständige Liste der Amazon S3-Ressourcen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Um den ARN für einen S3-Bucket zu finden, können Sie sich die BerechtigungsseitenBucket Policy (Bucket-Richtlinie) oder CORS configuration (CORS-Konfiguration) in der Amazon S3-Konsole ansehen. Weitere Informationen finden Sie in folgenden Themen im Konsolenbenutzerhandbuch für Amazon Simple Storage Service:

- Wie füge ich eine S3-Bucket-Richtlinie hinzu?
- Wie lasse ich ein domänenübergreifendes Ressourcen-Sharing mit CORS zu?

Amazon S3-ARN-Beispiele

Es folgen Beispiele für Amazon S3-Ressourcen-ARNs.

Bucket-Name und Objektschlüssel angegeben

Der folgende ARN gibt das Objekt `/developers/design_info.doc` im Bucket `examplebucket` an.

```
arn:aws:s3:::examplebucket/developers/design_info.doc
```

Platzhalter

Sie können im ARN der Ressource Platzhalter verwenden. Sie können Platzhalterzeichen (`*` und `?`) in einem beliebigen ARN-Segment (durch Doppelpunkte getrennte Teile) verwenden. Ein Sternchen (`*`) steht für kein Zeichen oder eine beliebige Kombination von mehreren Zeichen und ein Fragezeichen (`?`) entspricht einem beliebigen einzelnen Zeichen. Sie können mehrere `*`- oder `?`-Zeichen in jedem Segment verwenden, allerdings sind Platzhalter nicht segmentübergreifend.

- Der folgende ARN verwendet den Platzhalter `*` im Teil mit der relativen ID des ARN, um alle Objekte im Bucket `examplebucket` anzugeben.

  ```
  arn:aws:s3:::examplebucket/*
  ```

- Der folgende ARN verwendet `*`, um alle Amazon S3-Ressourcen (alle S3-Buckets und Objekte in Ihrem Konto) anzugeben.

  ```
  arn:aws:s3:::*
  ```

- Der folgende ARN verwendet beide Platzhalter, `*` und `?`, im Teil `relative-ID`. Er identifiziert alle Objekte in Buckets wie `example1bucket`, `example2bucket`, `example3bucket` usw.

  ```
  arn:aws:s3:::example?bucket/*
  ```

Richtlinienvariablen

Sie können Richtlinienvariablen in Amazon S3-ARNs verwenden. Bei der Richtlinienauswertung werden diese vordefinierten Variablen durch ihre entsprechenden Werte ersetzt. Angenommen, Sie organisieren Ihren Bucket als eine Sammlung von Ordnern, je ein Ordner für jeden Ihrer Benutzer. Der Ordnername entspricht dabei dem Benutzernamen. Um den Benutzern Berechtigungen für ihre Ordner zu erteilen, können Sie eine Richtlinienvariable im Ressourcen-ARN angeben:

```
arn:aws:s3:::bucket_name/developers/${aws:username}/
```

Zur Laufzeit wird bei der Auswertung der Richtlinie die Variable `${aws:username}` im Ressourcen-ARN durch den Benutzernamen ersetzt, der die Anforderung stellt.

Prinzipale

Das Element `Principal` gibt an, welchem Benutzer, Konto, Dienst oder welcher anderen Entität der Zugriff auf eine Ressource gewährt oder verweigert wird. Die folgenden sind Beispiele legen den `Principal` fest. Weitere Informationen finden Sie unter Prinzipal im IAM-Benutzerhandbuch.

Erteilen von Berechtigungen für ein AWS-Konto

Um Berechtigungen für ein AWS-Konto zu erteilen, identifizieren Sie das Konto im folgenden Format.

```
"AWS":"account-ARN"
```

Im Folgenden sind einige Beispiele aufgeführt.

```
"Principal":{"AWS":"arn:aws:iam::AccountNumber-WithoutHyphens:root"}
```

```
"Principal":{"AWS":["arn:aws:iam::AccountNumber1-
WithoutHyphens:root","arn:aws:iam::AccountNumber2-WithoutHyphens:root"]}
```

Amazon S3 unterstützt auch eine kanonische Benutzer-ID, eine verschleierte Form der AWS-Konto-ID. Sie können diese ID im folgenden Format angeben.

```
"CanonicalUser":"64-digit-alphanumeric-value"
```

Im Folgenden wird ein Beispiel gezeigt.

```
"Principal":{"CanonicalUser":"64-digit-alphanumeric-value"}
```

Informationen darüber, wo Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Suchen Ihrer kanonischen Benutzer-ID für das Konto.

> **Important**
>
> Wenn Sie eine kanonische Benutzer-ID in einer Richtlinie verwenden, ändert Amazon S3 möglicherweise die kanonische ID in die entsprechende AWS-Konto-ID. Dies hat keine Auswirkungen auf die Richtlinie, da beide IDs dasselbe Konto identifizieren.

Erteilen von Berechtigungen für einen IAM-Benutzer

Um einem IAM-Benutzer in Ihrem Konto die Berechtigung zu erteilen, müssen Sie ein `"AWS":"user-ARN"`-Name-Wert-Paar bereitstellen.

```
"Principal":{"AWS":"arn:aws:iam::account-number-without-hyphens:user/username"}
```

Erteilen anonymer Berechtigungen

Um jedem Benutzer Berechtigung zu erteilen, auch als anonymer Zugriff bezeichnet, legen Sie den Platzhalter (`"*"`) auf den Wert `Principal` fest. Wenn Sie beispielsweise ihren Bucket als Website konfigurieren, müssen Sie alle Objekte im Bucket öffentlich zugänglich machen. Die folgenden Elemente sind gleichwertig.

```
"Principal":"*"
```

```
"Principal":{"AWS":"*"}
```

> **Warning**
>
> Seien Sie vorsichtig, wenn Sie anonymen Zugriff auf Ihren S3-Bucket erteilen. Wenn Sie anonymen Zugriff gewähren, kann jeder auf der ganzen Welt auf Ihren Bucket zugreifen. Wir empfehlen dringend, nie einen anonymen Schreibzugriff auf Ihren S3-Bucket zu gewähren.

Anfordern von Zugriff über CloudFront-URLs

Sie können anfordern, dass Ihre Benutzer auf Ihre Amazon S3-Inhalte über Amazon CloudFront-URLs und nicht über Amazon S3-URLs zugreifen. Hierzu erstellen Sie eine CloudFront-Ursprungszugriffsidentität

(Origin Access Identity, OAI). Anschließend ändern Sie die Berechtigungen für Ihren Bucket oder für die Objekte in Ihrem Bucket. Das Format für die Angabe des OAI in einer `Principal`-Anweisung ist wie folgt.

```
"Principal":{"CanonicalUser":"Amazon S3 Canonical User ID assigned to origin access
 identity"}
```

Weitere Informationen finden Sie unter Verwenden einer Ursprungszugriffsidentität zum Einschränken des Zugriffs auf Ihre Amazon S3-Inhalte im Entwicklerhandbuch für Amazon CloudFront.

Amazon S3-Aktionen

Amazon S3 definiert eine Reihe von Berechtigungen, die Sie in einer Richtlinie angeben können. Dies sind Schlüsselwörter. Jedes Schlüsselwort ist einer bestimmten Amazon S3-Operation zugeordnet. Weitere Informationen zu Amazon S3-Operationen finden Sie unter Aktionen im Amazon Simple Storage Service API Reference.

Informationen dazu, wie Sie Berechtigungen in Amazon S3 in einer Richtlinie angeben, finden Sie in den folgenden Beispielrichtlinien. Eine Liste der Amazon S3-Aktionen, -Ressourcen und -Bedingungsschlüssel für die Verwendung in Richtlinien finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394). Eine vollständige Liste der Amazon S3-Aktionen finden Sie unter Actions (Aktionen).

Themen

- Beispiel – Objektoperationen (p. 380)
- Beispiel – Bucket-Operationen (p. 381)
- Beispiel – Bucket-Subressourcen-Operationen (p. 381)
- Beispiel – Kontenoperationen (p. 382)

Beispiel – Objektoperationen

Die folgende Beispiel-Bucket-Richtlinie erteilt die Berechtigungen `s3:PutObject` und `s3:PutObjectAcl` für den Benutzer (Dave). Wenn Sie das Element `Principal` entfernen, können Sie die Richtlinie einem Benutzer zuweisen. Dies sind Objektoperationen. Dementsprechend identifiziert der Teil `relative-id` des `Resource`-ARN Objekte (`examplebucket/*`). Weitere Informationen finden Sie unter Amazon S3-Ressourcen (p. 377).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/Dave"
            },
            "Action":   ["s3:PutObject","s3:PutObjectAcl"],
            "Resource": "arn:aws:s3:::examplebucket/*"
        }
    ]
}
```

Berechtigungen für alle Amazon S3-Aktionen

Sie können ein Platzhalterzeichen verwenden, um die Berechtigung für alle Amazon S3-Aktionen zu erteilen.

```
"Action":   "*"
```

Beispiel – Bucket-Operationen

Die folgende Beispiel-Benutzerrichtlinie erteilt die Berechtigungen `s3:CreateBucket`, `s3:ListAllMyBuckets` und `s3:GetBucketLocation` für einen Benutzer. Für alle diese Berechtigungen legen Sie den Teil `relative-id` des `Resource`-ARN auf „*“ fest. Bei allen anderen Bucket-Aktionen müssen Sie einen Bucket-Namen angegeben. Weitere Informationen finden Sie unter Amazon S3-Ressourcen (p. 377).

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action":[
            "s3:CreateBucket",
            "s3:ListAllMyBuckets",
            "s3:GetBucketLocation"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ]
       }
    ]
}
```

Richtlinie für den Konsolenzugriff

Wenn ein Benutzer die AWS Management Console verwenden möchte, um Buckets und den Inhalt eines dieser Buckets anzuzeigen, muss der Benutzer die Berechtigungen `s3:GetBucketLocation` und `s3:ListAllMyBuckets` besitzen. Ein Beispiel finden Sie unter Richtlinie für den Konsolenzugriff im Blogbeitrag Schreiben von IAM-Richtlinien: So gewähren Sie Zugriff auf einen S3 Bucket.

Beispiel – Bucket-Subressourcen-Operationen

Die folgende Beispiel-Benutzerrichtlinie erteilt die `s3:GetBucketAcl`-Berechtigung für den Bucket `examplebucket` für den Benutzer Dave.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::Account-ID:user/Dave"
      },
      "Action": [
        "s3:GetObjectVersion",
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::examplebucket"
    }
  ]
}
```

LÖSCHEN von Objektberechtigungen

Sie können Objekte löschen, indem Sie explizit die DELETE-Objekt-API aufrufen oder ihren Lebenszyklus konfigurieren (vgl. Verwaltung des Objektlebenszyklus (p. 139)), damit Amazon S3 die Objekte entfernen kann, wenn ihr Lebenszyklus abläuft. Um explizit Benutzer oder Konten vom Löschen von Objekten zu

sperren, müssen Sie ihnen die Berechtigungen `s3:DeleteObject`, `s3:DeleteObjectVersion` und `s3:PutLifecycleConfiguration` explizit entziehen.

Explizite Zugriffsverweigerung

Standardmäßig besitzen Benutzer keine Berechtigungen. Wenn Sie Benutzer erstellen, Benutzer zu Gruppen hinzufügen und diesen Berechtigungen erteilen, erhalten diese möglicherweise bestimmte Berechtigungen, die Sie nicht erteilen möchten. Dann können Sie explizite Sperren verwenden, die alle anderen Berechtigungen außer Kraft setzen, die ein Benutzer möglicherweise hat, und somit die Benutzerberechtigungen für bestimmte Aktionen verweigern.

Beispiel – Kontenoperationen

Die folgende Beispiel-Benutzerrichtlinie erteilt die Berechtigung `s3:GetAccountPublicAccessBlock` für einen Benutzer. Für diese Berechtigungen legen Sie den Wert `Resource` auf `"*"` fest. Weitere Informationen finden Sie unter Amazon S3-Ressourcen (p. 377).

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action":[
            "s3:GetAccountPublicAccessBlock"
         ],
         "Resource":[
            "*"
         ]
       }
    ]
}
```

Amazon S3-Bedingungsschlüssel

Mit der Sprache der Zugriffsrichtlinie können Sie bei der Erteilung von Berechtigungen Bedingungen angeben. Um Bedingungen dafür festzulegen, wann eine Richtlinie gültig ist, können Sie das optionale Element `Condition` oder den Block `Condition` verwenden. Sie können vordefinierte AWS-weite Schlüssel und Amazon S3-spezifische Schlüssel verwenden, um Bedingungen in einer Amazon S3-Zugriffsrichtlinie anzugeben.

Im Element `Condition` formulieren Sie Ausdrücke, in denen Sie boolesche Operatoren verwenden (gleich, kleiner als usw.), um die Bedingung auf Übereinstimmung mit den Werten in der Anforderung zu prüfen. Wenn Sie einem Benutzer beispielsweise die Berechtigung zum Hochladen eines Objekts erteilen, kann der Bucket-Eigentümer anfordern, dass das Objekt öffentlich lesbar ist, indem er die Bedingung `StringEquals` wie hier gezeigt hinzufügt:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ],
      "Condition": {
        "StringEquals": {
```

```
          "s3:x-amz-acl": [
            "public-read"
          ]
        }
      }
    }
  ]
}
```

Im Beispiel gibt der Block `Condition` die Bedingung `StringEquals` an, die auf das angegebene Schlüssel-Wert-Paar angewendet wird, `"s3:x-amz-acl":["public-read"]`. Es gibt einen Satz vordefinierter Schlüssel, die Sie zum Ausdruck einer Bedingung verwenden können. Das Beispiel verwendet den Bedingungsschlüssel `s3:x-amz-acl`. Diese Bedingung erfordert, dass der Benutzer in jeder PUT Object-Anforderung den Header `x-amz-acl` mit dem Wert `public-read` angibt.

Themen

AWS-weite Bedingungsschlüssel

AWS stellt einen Satz gemeinsamer Schlüssel bereit, die von allen AWS-Services unterstützt werden, die Richtlinien unterstützen. Diese Schlüssel werden als AWS--weite Schlüssel bezeichnet. Sie verwenden das Präfix `aws:`. Die vollständige Liste der AWS-weiten Bedingungsschlüssel finden Sie unter Verfügbare AWS-Schlüssel für Bedingungen im IAM-Benutzerhandbuch.

Sie können AWS-weite Bedingungsschlüssel in Amazon S3 verwenden. Die folgende Beispiel-Bucket-Richtlinie ermöglicht authentifizierten Benutzern das Verwenden der Aktion `s3:GetObject`, wenn die Anforderung aus einem bestimmten Bereich von IP-Adressen stammt (192.0.2.0.*), sofern die IP-Adresse nicht 192.0.2.188 ist. Im Bedingungsblock sind die Bedingungen `IpAddress` und `NotIpAddress` enthalten, und bei jeder Bedingung ist ein Schlüssel-Wert-Paar zur Auswertung angefügt. Beide Schlüssel-Wert-Paare in diesem Beispiel verwenden den AWS-weiten Schlüssel `aws:SourceIp`.

> **Note**
>
> Die in der Bedingung angegebenen Schlüsselwerte `IPAddress` und `NotIpAddress` verwenden die CIDR-Notation, wie in RFC 4632 beschrieben. Weitere Informationen finden Sie unter http://www.rfc-editor.org/rfc/rfc4632.txt.

```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": "*",
            "Action":["s3:GetObject"]  ,
            "Resource": "arn:aws:s3:::examplebucket/*",
            "Condition" : {
                "IpAddress" : {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NotIpAddress" : {
                    "aws:SourceIp": "192.0.2.188/32"
                }
            }
        }
```

```
    ]
}
```

Sie können auch andere AWS-weite Bedingungsschlüssel in Amazon S3-Richtlinien verwenden. Beispielsweise können Sie die Bedingungsschlüssel `aws:sourceVpce` und `aws:sourceVpc` in Bucket-Richtlinien für VPC-Endpunkte angeben. Beispiele finden Sie unter Beispiel für Bucket-Richtlinien für VPC-Endpunkte für Amazon S3 (p. 454).

Amazon S3-spezifische Bedingungsschlüssel

Sie können Amazon S3-Bedingungsschlüssel mit bestimmten Amazon S3-Aktionen verwenden. Jeder Bedingungsschlüssel wird dem gleichnamigen Anforderungs-Header zugeordnet, der von der API zugelassen wird und auf dem die Bedingung festgelegt werden kann. Amazon S3-spezifische Bedingungsschlüssel diktieren das Verhalten der gleichnamigen Anforderungs-Header. Die vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Der Bedingungsschlüssel `s3:x-amz-acl`, den Sie zum Erteilen der Bedingungsberechtigung `s3:PutObject` verwenden können, definiert das Verhalten des Anforderungs-Headers `x-amz-acl`, den die PUT-Objekt-API unterstützt. Der Bedingungsschlüssel `s3:VersionId`, den Sie für die bedingte Berechtigung für die Bedingung `s3:GetObjectVersion` verwenden können, definiert das Verhalten des Abfrageparameters `versionId`, den Sie in einer GET-Objekt-Anforderung setzen.

Die folgende Bucket-Richtlinie erteilt die Berechtigung `s3:PutObject` für zwei AWS-Konten, wenn die Anforderung den Header `x-amz-acl` enthält, durch den das Objekt öffentlich lesbar ist. Der `Condition`-Block verwendet die `StringEquals`-Bedingung und ist mit dem Schlüssel-Wert-Paar `"s3:x-amz-acl": ["public-read"]` zur Auswertung versehen. Im Schlüssel-Wert-Paar ist `s3:x-amz-acl` ein Amazon S3–spezifischer Schlüssel, wie durch das Präfix `s3:` angezeigt wird.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
         "Sid":"AddCannedAcl",
            "Effect":"Allow",
         "Principal": {
                "AWS": ["arn:aws:iam::account1-ID:root","arn:aws:iam::account2-ID:root"]
            },
         "Action":["s3:PutObject"],
            "Resource": ["arn:aws:s3:::examplebucket/*"],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl":["public-read"]
                }
            }
        }
    ]
}
```

> **Important**
>
> Nicht alle Bedingungen machen für alle Aktionen auch Sinn. Beispielsweise ist es sinnvoll, die Bedingung `s3:LocationConstraint` für eine Richtlinie einzufügen, die die Amazon S3-Berechtigung `s3:CreateBucket` erteilt. Es ist jedoch nicht sinnvoll, diese Bedingung in eine Richtlinie aufzunehmen, die die Berechtigung `s3:GetObject` erteilt. Amazon S3 kann auf semantische Fehler dieses Typs testen, die Amazon S3–spezifische Bedingungen enthalten. Wenn Sie jedoch eine Richtlinie für einen IAM-Benutzer erstellen und eine semantisch

ungültige Amazon S3-Bedingung einfügen, wird kein Fehler gemeldet, da IAM keine Amazon S3-Bedingungen validieren kann.

Beispiele: Amazon S3-Bedingungsschlüssel für Objektoperationen

Dieser Abschnitt enthält Beispiele, die zeigen, wie Sie Amazon S3-spezifische Bedingungsschlüssel für Objektoperationen verwenden können. Die vollständige Liste der Amazon S3-Aktionen, -Bedingungsschlüssel und -Ressourcen, die Sie in Richtlinien angeben können, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Einige der Beispielrichtlinien zeigen, wie Sie Bedingungsschlüssel mit PUT Object-Operationen verwenden können. PUT Object-Operationen ermöglichen für Zugriffskontrolllisten (ACLs) spezifische Header, mit denen Sie ACL-basierte Berechtigungen erteilen können. Mit diesen Schlüsseln kann der Bucket-Eigentümer eine Bedingung festlegen, die bestimmte Zugriffsberechtigungen erfordert, wenn der Benutzer ein Objekt hochlädt. Sie können ACL-basierte Berechtigungen auch mittels der Operation PutObjectAcl erteilen. Weitere Informationen finden Sie unter PutObjectAcl im Amazon S3Amazon Simple Storage Service API Reference. Weitere Informationen über ACLs finden Sie in Zugriffskontrolllisten (ACL) – Übersicht (p. 480).

Beispiel 1: s3:PutObject-Berechtigung mit der Bedingung erteilen, dass der Bucket-Eigentümer vollständige Kontrolle erhalten muss

Die Operation PUT Object erlaubt Zugriffskontrolllisten (ACL)-spezifische Header, mit denen Sie ACL-spezifische Berechtigungen erteilen können. Mit diesen Schlüsseln kann der Bucket-Eigentümer eine Bedingung festlegen, die bestimmte Zugriffsberechtigungen erfordert, wenn der Benutzer ein Objekt hochlädt.

Angenommen, Konto A besitzt einen Bucket und der Kontoadministrator möchte Dave, einem Benutzer in Konto B, Berechtigungen zum Hochladen von Objekten erteilen. Standardmäßig gehören Objekte, die Dave hochgeladen hat, zu Konto B, und Konto A besitzt keine Berechtigungen für diese Objekte. Da der Bucket-Eigentümer die Rechnungen bezahlt, benötigt er volle Berechtigungen für die Objekte, die Dave hochlädt. Der Administrator von Konto A kann für diesen Zweck Dave die Berechtigung `s3:PutObject` erteilen. Dies erfolgt unter der Bedingung, dass die Anforderung ACL-spezifische Header enthält, die die vollständige Berechtigung explizit gewähren oder eine vordefinierte ACL verwenden. Weitere Informationen finden Sie unter PUT Object.

Anfordern des Headers `x-amz-full-control`

Sie können in der Anforderung den Header `x-amz-full-control` mit vollständiger Kontrollberechtigung für den Bucket-Eigentümer anfordern. Die folgende Bucket-Richtlinie erteilt dem Benutzer Dave die Berechtigung `s3:PutObject` mit einer Bedingung, die den Bedingungsschlüssel `s3:x-amz-grant-full-control` enthält, in die Anforderung den Header `x-amz-full-control` aufzunehmen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountB-ID:user/Dave"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
        }
      }
    }
  ]
```

```
}
```

> **Note**
>
> In diesem Beispiel geht es um die kontoübergreifende Berechtigung. Wenn Dave (der die Berechtigung erhält) jedoch zu dem AWS-Konto gehört, dem der Bucket gehört, ist diese bedingte Berechtigung nicht erforderlich. Grund hierfür ist, dass das übergeordnete Konto, dem Dave angehört, Eigentümer von Objekten ist, die der Benutzer hochlädt.

Hinzufügen einer expliziten Zugriffsverweigerung

Die vorherige Bucket-Richtlinie erteilt dem Benutzer Dave in Konto B eine bedingte Berechtigung. Während diese Richtlinie in Kraft ist, ist es Dave möglich, die gleiche Berechtigung ohne Bedingung über eine andere Richtlinie zu erhalten. Beispiel: Dave kann einer Gruppe angehören, der Sie die Berechtigung `s3:PutObject` bedingungslos erteilen. Um solche Berechtigungslücken zu vermeiden, können Sie eine strengere Zugriffsrichtlinie schreiben, indem Sie eine explizite Zugriffsverweigerung hinzufügen. In diesem Beispiel verweigern Sie explizit die Upload-Berechtigung für Benutzer Dave, wenn er nicht die erforderlichen Header in die Anforderung einbindet, die dem Bucket-Eigentümer volle Berechtigungen gewährt. Eine explizite Verweigerung ersetzt immer eine anderswo erteilte Erlaubnis. Im Folgenden finden Sie das geänderte Zugriffsrichtlinienbeispiel mit hinzugefügter expliziter Ablehnung.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::examplebucket/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
                }
            }
        },
        {
            "Sid": "statement2",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::examplebucket/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
                }
            }
        }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Wenn Sie zwei AWS-Konten haben, können Sie die Richtlinie mit AWS Command Line Interface (AWS CLI) testen. Sie fügen die Richtlinie an und verwenden die Anmeldeinformationen von Dave, um die Berechtigung mit dem AWS CLI-Befehl `put-object` zu testen. Sie stellen dem Benutzer Dave die Anmeldeinformationen bereit, indem Sie den Parameter `--profile` hinzufügen. Sie erteilen dem Bucket-

Eigentümer die volle Kontrolle, indem Sie den Parameter `--grant-full-control` hinzufügen. Weitere Informationen zum Einrichten und Verwenden der AWS CLI finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body c:\HappyFace.jpg --
grant-full-control id="AccountA-CanonicalUserID" --profile AccountBUserProfile
```

Anfordern des Headers `x-amz-acl`

Sie können den Header `x-amz-acl` mit einer vordefinierten ACL anfordern, die dem Bucket-Eigentümer die vollständige Kontrollberechtigung erteilt. Um den Header `x-amz-acl` in der Anforderung erforderlich zu machen, können Sie das Schlüssel-Wert-Paar im Block `Condition` ersetzen und den Bedingungsschlüssel `s3:x-amz-acl` wie im folgenden Beispiel dargestellt angeben.

```
"Condition": {
 "StringNotEquals": {
  "s3:x-amz-acl": "bucket-owner-full-control"
 }
```

Um die Berechtigung mithilfe der AWS CLI zu testen, geben Sie den Parameter `--acl` an. Die AWS CLI fügt dann beim Senden der Anforderung den Header `x-amz-acl` hinzu.

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body c:\HappyFace.jpg --
acl "bucket-owner-full-control" --profile AccountBadmin
```

Beispiel 2: 3:PutObject-Berechtigung unter der Bedingung erteilen, dass Objekte mit serverseitiger Verschlüsselung gespeichert werden

Angenommen, Konto A besitzt einen Bucket. Der Konto-Administrator möchte Jane, einer Benutzerin in Konto A, die Berechtigung zum Hochladen von Objekten mit einer Bedingung erteilen, dass Jane immer serverseitige Verschlüsselung anfordert, damit Amazon S3 die Objekte verschlüsselt speichert. Der Administrator von Konto A kann für diesen Zweck wie gezeigt mit dem Bedingungsschlüssel `s3:x-amz-server-side-encryption` arbeiten. Das Schlüssel-Wert-Paar im Block `Condition` gibt den Schlüssel `s3:x-amz-server-side-encryption` an.

```
"Condition": {
     "StringNotEquals": {
         "s3:x-amz-server-side-encryption": "AES256"
     }
```

Um die Berechtigung mithilfe der AWS CLI zu testen, müssen Sie den erforderlichen Parameter `--server-side-encryption` hinzufügen.

```
aws s3api put-object --bucket example1bucket --key HappyFace.jpg --body c:\HappyFace.jpg --
server-side-encryption "AES256" --profile AccountBadmin
```

Beispiel 3: s3:PutObject-Berechtigung zum Kopieren von Objekten mit einer Einschränkung für die Kopierquelle erteilen

Wenn Sie in der PUT-Objektanforderung ein Quellobjekt angeben, handelt es sich um eine Kopieroperation (siehe PUT Object - Copy). Dementsprechend kann der Bucket-Eigentümer einem Benutzer die Berechtigung erteilen, Objekte mit Einschränkungen in Bezug auf die Quelle zu kopieren, z. B.:

- das Kopieren von Objekten nur aus dem `sourcebucket`-Bucket erlauben.
- das Kopieren von Objekten aus dem Bucket `sourcebucket` und nur der Objekte erlauben, deren Schlüsselnamenpräfix mit `public/` f beginnt (z. B. `sourcebucket/public/*`).

- das Kopieren nur eines bestimmten Objekts aus dem Bucket `sourcebucket` erlauben (z. B. `sourcebucket/example.jpg`).

Die folgende Bucket-Richtlinie erteilt dem Benutzer Dave die Berechtigung `s3:PutObject`. Sie erlaubt ihm, nur Objekte mit der Bedingung zu kopieren, dass die Anforderung den Header `s3:x-amz-copy-source` enthält und der Header-Wert das Präfix des Schlüsselnamens `/examplebucket/public/*` angibt.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Sid": "cross-account permission to user in your own account",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
            },
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::examplebucket/*"
        },
        {
            "Sid": "Deny your user permission to upload object if copy source is not /
bucket/folder",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::examplebucket/*",
            "Condition": {
                "StringNotLike": {
                    "s3:x-amz-copy-source": "examplebucket/public/*"
                }
            }
        }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Berechtigung mit dem AWS CLI-Befehl `copy-object` testen. Sie geben die Quelle an, indem Sie den Parameter `--copy-source` hinzufügen. Das Schlüsselnamenpräfix muss dem Präfix entsprechen, das in der Richtlinie zulässig ist. Sie müssen die Anmeldeinformationen des Benutzers Dave mit dem Parameter `--profile` angeben. Weitere Informationen zum Festlegen der AWS CLI finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

```
aws s3api copy-object --bucket examplebucket --key HappyFace.jpg
--copy-source examplebucket/public/PublicHappyFace1.jpg --profile AccountADave
```

Erteilen der Berechtigung, nur ein bestimmtes Objekt zu kopieren

Die vorherige Richtlinie verwendet die Bedingung `StringNotLike`. Um nur das Kopieren eines bestimmten Objekts zu erlauben, müssen Sie die Bedingung von `StringNotLike` in `StringNotEquals` ändern und dann den genauen Objektschlüssel wie gezeigt angeben.

```
"Condition": {
       "StringNotEquals": {
           "s3:x-amz-copy-source": "examplebucket/public/PublicHappyFace1.jpg"
       }
}
```

Beispiel 4: Gewähren von Zugriff auf eine bestimmte Version eines Objekts

Angenommen, Konto A besitzt einen versionsaktivierten Bucket. Der Bucket beinhaltet mehrere Versionen des Objekts `HappyFace.jpg`. Der Kontoadministrator möchte nun dem Benutzer Dave die Berechtigung erteilen, nur eine bestimmte Version des Objekts zu erhalten. Der Kontoadministrator erreicht dies, indem er Dave die Berechtigung `s3:GetObjectVersion` bedingt erteilt wie unten gezeigt. Das Schlüssel-Wert-Paar im Block `Condition` gibt den Bedingungsschlüssel `s3:VersionId` an. In diesem Fall muss Dave die genaue Objekt-Versions-ID kennen, um das Objekt abzurufen.

Weitere Informationen finden Sie unter GetObject im Amazon Simple Storage Service API Reference.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
            },
            "Action": ["s3:GetObjectVersion"],
            "Resource": "arn:aws:s3:::examplebucketversionenabled/HappyFace.jpg"
        },
        {
            "Sid": "statement2",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
            },
            "Action": ["s3:GetObjectVersion"],
            "Resource": "arn:aws:s3:::examplebucketversionenabled/HappyFace.jpg",
            "Condition": {
                "StringNotEquals": {
                    "s3:VersionId": "AaaHbAQitwiL_h47_44lRO2DDfLlBO5e"
                }
            }
        }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Berechtigungen mit dem AWS CLI;-Befehl `get-object` mit dem Parameter `--version-id` testen, der die spezifische Objektversion identifiziert. Der Befehl ruft das Objekt ab und speichert es in der Datei `OutputFile.jpg`.

```
aws s3api get-object --bucket examplebucketversionenabled --key HappyFace.jpg
 OutputFile.jpg --version-id AaaHbAQitwiL_h47_44lRO2DDfLlBO5e --profile AccountADave
```

Beispiel 5: Objekt-Uploads auf Objekte mit einer bestimmten Speicherklasse beschränken

Angenommen, Konto A besitzt einen Bucket. Der Kontoadministrator möchte Dave, einen Benutzer in Konto A, insofern einschränken, nur Objekte in den Bucket hochladen zu können, die mit der Speicherklasse `STANDARD_IA` gespeichert werden. Um das Hochladen von Objekten auf eine bestimmte Speicherklasse einzuschränken, kann der Administrator von Konto A den Bedingungsschlüssel `s3:x-amz-storage-class` verwenden wie in der folgenden Bucket-Beispielrichtlinie gezeigt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
```

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
      },
      "Action": "s3:PutObject",
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ],
      "Condition": {
        "StringEquals": {
          "s3:x-amz-storage-class": [
            "STANDARD_IA"
          ]
        }
      }
    }
  ]
}
```

Beispiel 6: Erteilen von Berechtigungen basierend auf Objekt-Tags

Beispiele für die Verwendung von Objektmarkierungs-Bedingungsschlüsseln mit Amazon S3-Operationen finden Sie unter Objektmarkierung und Zugriffskontrollrichtlinien (p. 133).

Beispiele — Amazon S3-Bedingungsschlüssel für Bucket-Operationen

Dieser Abschnitt enthält Beispielrichtlinien, die zeigen, wie Sie Amazon S3-spezifische Bedingungsschlüssel für Bucket-Operationen verwenden können.

Beispiel 1: Erteilen einer Benutzerberechtigung zum Erstellen von Buckets nur in einer bestimmten Region

Angenommen, ein AWS-Kontoadministrator möchte seinem Benutzer (Dave) die Berechtigung erteilen, einen Bucket nur in der Region Südamerika (São Paulo) zu erstellen. Der Kontoadministrator kann die folgende Benutzerrichtlinie anfügen, welche die Berechtigung `s3:CreateBucket` mit der angegebenen Bedingung gewährt. Das Schlüssel-Wert-Paar im Block `Condition` gibt den Schlüssel `s3:LocationConstraint` und die Region `sa-east-1` als seinen Wert an.

> **Note**
>
> In diesem Beispiel erteilt der Bucket-Eigentümer einem seiner Benutzer die Berechtigung, daher kann entweder eine Bucket-Richtlinie oder eine Benutzerrichtlinie verwendet werden. Dieses Beispiel zeigt eine Benutzerrichtlinie.

Eine Liste der Amazon S3-Regionen finden Sie unter Regionen und Endpunkte im AWS General Reference.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action":[
            "s3:CreateBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition": {
             "StringLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
```

```
        }
    ]
}
```

Hinzufügen einer expliziten Zugriffsverweigerung

Die vorangegangene Richtlinie hindert den Benutzer an der Erstellung eines Buckets in einer anderen Region als `sa-east-1`. Einige andere Richtlinien gewähren diesem Benutzer möglicherweise jedoch die Berechtigung, Buckets in einer anderen Region zu erstellen. Wenn der Benutzer beispielsweise zu einer Gruppe gehört, ist der Gruppe möglicherweise eine Richtlinie angefügt, die alle Benutzer der Gruppe zum Erstellen von Buckets in einer anderen Region berechtigt. Um sicherzustellen, dass der Benutzer keine Berechtigung zum Erstellen von Buckets in einer anderen Region erhält, können Sie in der oben gezeigten Richtlinie eine explizite Anweisung zur Ablehnung hinzufügen.

Die Anweisung `Deny` verwendet die Bedingung `StringNotLike`. Das bedeutet, dass, eine Anforderung zum Erstellen eines Buckets abgelehnt wird, wenn die Standorteinschränkung nicht `sa-east-1` ist. Die explizite Verweigerung erlaubt dem Benutzer nicht, einen Bucket in einer anderen Region zu erstellen, unabhängig davon, welche andere Berechtigung der Benutzer erhält. Die folgende Richtlinie enthält eine explizite Zugriffsverweigerungsanweisung.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action":[
            "s3:CreateBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition": {
             "StringLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Action":[
            "s3:CreateBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition": {
             "StringNotLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       }
    ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Richtlinie mit dem folgenden `create-bucket` AWS CLI-Befehl testen. In diesem Beispiel wird die Datei `bucketconfig.txt` verwendet, um die Standortbeschränkung anzugeben. Merken Sie sich den Windows-Pfad der Datei. Sie müssen den Bucket-Namen und -Pfad entsprechend aktualisieren. Sie müssen die Anmeldeinformationen des Benutzers bereitstellen, indem Sie den Parameter `--profile`

hinzufügen. Weitere Informationen zum Einrichten und Verwenden der AWS CLI finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

```
aws s3api create-bucket --bucket examplebucket --profile AccountADave --create-bucket-
configuration file://c:/Users/someUser/bucketconfig.txt
```

Die Datei `bucketconfig.txt` gibt die Konfiguration wie folgt an.

```
{"LocationConstraint": "sa-east-1"}
```

Beispiel 2: Abrufen einer Liste von Objekten in einem Bucket mit einem bestimmten Präfix

Sie können den Bedingungsschlüssel `s3:prefix` verwenden, um die Antwort der API GET Bucket (ListObjects) auf Schlüsselnamen mit einem bestimmten Präfix einzuschränken. Wenn Sie der Bucket-Eigentümer sind, können Sie einen Benutzer beschränken, den Inhalt eines bestimmten Präfixes im Bucket aufzulisten. Dieser Bedingungsschlüssel ist nützlich, wenn Objekte im Bucket nach Schlüsselnamenpräfixen organisiert sind. Die Amazon S3-Konsole verwendet Schlüsselnamenpräfixe zum Anzeigen von Ordnerkonzepten. Nur die Konsole unterstützt das Ordnerkonzept. Die Amazon S3-API unterstützt ausschließlich Buckets und Objekte. Weitere Informationen zur Verwendung von Präfixen und Trennzeichen zum Filtern von Zugriffsberechtigungen finden Sie unter Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien (p. 461).

Wenn es beispielsweise zwei Objekte mit den Schlüsselnamen `public/object2.jpg` und `public` gibt, zeigt die Konsole die Objekte unter dem Ordner `public/object1.jpg` an. In der Amazon S3-API sind dies Objekte mit Präfixen, nicht Objekte in Ordnern. Wenn Sie jedoch in der Amazon S3-API die Objektschlüssel mithilfe solcher Präfixe organisieren, können Sie der Berechtigung `s3:ListBucket` die Bedingung `s3:prefix` erteilen, die dem Benutzer das Abrufen einer Liste mit Schlüsselnamen mit diesen spezifischen Präfixen ermöglicht.

In diesem Beispiel sind der Bucket-Eigentümer und das übergeordnete Konto, zu dem der Benutzer gehört, dieselben. Der Bucket-Eigentümer kann also entweder eine Bucket-Richtlinie oder eine Benutzerrichtlinie verwenden. Weitere Informationen zu anderen Bedingungsschlüsseln, die Sie mit der API „GET Bucket (ListObjects)“ verwenden können, finden Sie unter ListObjects.

-Benutzerrichtlinie

Die folgende Benutzerrichtlinie erteilt die Berechtigung für den Bucket `s3:ListBucket` (siehe GET Bucket (List Objects)) mit der Bedingung, dass der Benutzer in der Anforderung das Präfix `prefix` mit dem Wert `projects` anführt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::examplebucket"
         ],
         "Condition" : {
             "StringEquals" : {
                 "s3:prefix": "projects"
             }
          }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
```

```
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::examplebucket"
         ],
         "Condition" : {
             "StringNotEquals" : {
                 "s3:prefix": "projects"
             }
          }
       }
    ]
}
```

Die Bedingung schränkt den Benutzer ein, nur Objektschlüssel mit dem Präfix `projects` auflisten zu können. Die hinzugefügte explizite Verweigerung verhindert, dass die Anforderung des Benutzers auf Auflistung von Schlüsseln mit irgendeinem anderen Präfix verweigert wird, unabhängig davon, welche anderen Berechtigungen der Benutzer gegebenenfalls hat. Zum Beispiel ist es möglich, dass der Benutzer die Berechtigung erhält, Objektschlüssel ohne Einschränkung aufzulisten, entweder durch Aktualisierungen der vorherigen Benutzerrichtlinie oder durch eine Bucket-Richtlinie. Da jedoch die explizite Zugriffsverweigerung stets Vorrang hat, wird die Benutzeranforderung zum Auflisten anderer Schlüssel, die nicht das Präfix `project` enthalten, abgelehnt.

Bucket-Richtlinie

Wenn Sie der oben gezeigten Richtlinie das Element `Principal` hinzufügen, das den Benutzer identifiziert, erhalten Sie eine Bucket-Richtlinie wie gezeigt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Principal": {
            "AWS": "arn:aws:iam::BucketOwner-accountID:user/user-name"
         },
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::examplebucket"
         ],
         "Condition" : {
             "StringEquals" : {
                 "s3:prefix": "examplefolder"
             }
          }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Principal": {
            "AWS": "arn:aws:iam::BucketOwner-AccountID:user/user-name"
         },
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::examplebucket"
         ],
         "Condition" : {
```

```
            "StringNotEquals" : {
                "s3:prefix": "examplefolder"
            }
        }
    }
  ]
}
```

Testen der Richtlinie mit der AWS CLI

Sie können die Richtlinie mit dem folgenden `list-object` AWS CLI-Befehl testen. Im Befehl geben Sie Benutzeranmeldeinformationen mit dem Parameter `--profile` an. Weitere Informationen zum Einrichten und Verwenden der AWS CLI finden Sie unter Einrichten der Tools für die beispielhaften Walkthroughs (p. 349).

```
aws s3api list-objects --bucket examplebucket --prefix examplefolder --profile AccountADave
```

Wenn der Bucket versionsaktiviert ist, müssen Sie die Berechtigung `s3:ListBucketVersions` in der vorherigen Richtlinie erteilen, um die Objekte im Bucket auflisten zu können, und nicht die Berechtigung `s3:ListBucket`. Diese Berechtigung unterstützt auch den `s3:prefix`-Bedingungsschlüssel.

Beispiel 3: Festlegen der maximalen Anzahl von Schlüsseln

Sie können den Bedingungsschlüssel `s3:max-keys` verwenden, um die maximale Anzahl von Schlüsseln festzulegen, die der Anforderer in den Anforderungen GET Bucket (ListObjects) oder ListObjectVersions zurückgeben kann. Standardmäßig gibt die API bis zu 1000 Schlüssel zurück. Die Liste der numerischen Bedingungsoperatoren, die Sie mit `s3:max-keys` und begleitenden Beispielen verwenden können, finden Sie unter Numerische Bedingungsoperatoren im IAM-Benutzerhandbuch.

Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3

Amazon S3 (Servicepräfix: `s3`) stellt die folgenden servicespezifischen Ressourcen, Aktionen und Bedingungskontextschlüssel für die Verwendung in IAM-Berechtigungsrichtlinien bereit.

Referenzen:

- Erfahren Sie, wie Sie diesen Service konfigurieren.
- Zeigen Sie eine Liste der API-Operationen an, die für diesen Service verfügbar sind.
- Erfahren Sie, wie Sie diesen Service und seine Ressourcen mithilfe von IAM-Berechtigungsrichtlinien schützen.

> **Note**
>
> Dieser Abschnitt wird zurzeit aktualisiert. Die vollständigen Tabellen finden Sie später in diesem Monat.

Themen

- Von Amazon S3 definierte Aktionen (p. 394)
- Von Amazon S3 definierte Ressourcentypen (p. 444)
- Bedingungsschlüssel für Amazon S3 (p. 444)

Von Amazon S3 definierte Aktionen

Sie können die folgenden Aktionen im Element `Action` einer IAM-Richtlinienanweisung angeben. Verwenden Sie Richtlinien, um Berechtigungen zum Ausführen einer Operation in AWS zu erteilen. Wenn Sie eine Aktion in einer Richtlinie verwenden, erlauben oder verweigern Sie in der Regel den Zugriff auf die API-Operation oder den CLI-Befehl mit demselben Namen. Dabei kann es mitunter vorkommen, dass

eine einzige Aktion den Zugriff auf mehr als eine Operation steuert. Alternativ erfordern einige Operationen mehrere verschiedene Aktionen.

Die Spalte Ressourcentypen gibt an, ob die Aktion Berechtigungen auf Ressourcenebene unterstützt. Wenn es keinen Wert für diese Spalte gibt, müssen Sie alle Ressourcen ("*") im Element `Resource` Ihrer Richtlinienanweisung angeben. Wenn die Spalte einen Ressourcentyp enthält, können Sie einen ARN dieses Typs in einer Anweisung mit dieser Aktion angeben. Erforderliche Ressourcen sind in der Tabelle mit einem Sternchen (*) gekennzeichnet. Wenn Sie einen Berechtigungs-ARN auf Ressourcenebene in einer Anweisung mit dieser Aktion angeben, muss er von diesem Typ sein. Einige Aktionen unterstützen mehrere Ressourcentypen. Wenn der Ressourcentyp optional ist (nicht als erforderlich angegeben), können Sie sich für einen Typ entscheiden.

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
AbortMultipartUpload	Bricht einen mehrteiligen Upload ab	Schreiben	object* (p. 444)		
				s3:DataAccessPointArn (p. 445) s3:DataAccessPointAccount (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
BypassGovernanceRetention	Gewährt die Umgehung der Einstellungen für die Aufbewahrung von Objekten im Governance-Modus	Verwaltung von Berechtigungen	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:RequestObjectTag/<key> (p. 445) s3:RequestObjectTagKeys (p. 445)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-copy-source (p. 446) s3:x-amz-grant-full-control (p. 446) s3:x-amz-grant-read (p. 446) s3:x-amz-grant-read-acp (p. 446) s3:x-amz-grant-write (p. 446) s3:x-amz-grant-write-acp (p. 446) s3:x-amz-metadata-directive (p. 446) s3:x-amz-server-side-encryption (p. 446)	

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:x-amz-server-side-encryption-aws-kms-key-id (p. 447) s3:x-amz-storage-class (p. 447) s3:x-amz-website-redirect-location (p. 447) s3:object-lock-mode (p. 446) s3:object-lock-retain-until-date (p. 446) s3:object-lock-remaining-retention-days (p. 446) s3:object-lock-legal-hold (p. 446)	
CreateAccessPoint	Erstellt einen neuen Zugriffspunkt.	Schreiben	accesspoint* (p. 444)		

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:locationconstraint (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446)	
CreateBucket	Erstellt einen Bucket	Schreiben	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:authtype (p. 445) s3:locationconstraint (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-grant-full-control (p. 446) s3:x-amz-grant-read (p. 446) s3:x-amz-grant-read-acp (p. 446) s3:x-amz-grant-write (p. 446) s3:x-amz-grant-write-acp (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
CreateJob	Erstellt einen neuen S3 Stapeloperationen-Auftrag.	Schreiben		s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446) s3:RequestJobPriority (p. 445) s3:RequestJobOperation (p. 445)	
DeleteAccessPoint	Löscht den in der URI genannten Zugriffspunkt	Schreiben	accesspoint* (p. 444)		
				s3:DataAccessPointArn (p. 445) s3:DataAccessPointAccount (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
DeleteAccessPointPolicy	Löschen der Richtlinie auf einem bestimmten Zugriffspunkt	Verwaltung von Berechtigungen	accesspoint* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointArn (p. 445) s3:DataAccessPointAccount (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
DeleteBucket	Löscht den in der URI benannten Bucket	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
DeleteBucketPolicy	Richtlinie aus dem angegebenen Bucket löschen	Verwaltung von Berechtigungen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
DeleteBucketWebsite	Löscht die Website-Konfiguration eines Buckets	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
DeleteObject	Entfernt die Null-Version (falls vorhanden) eines Objekts und fügt eine Löschmarkierung ein, die zur aktuellen Version des Objekts wird	Schreiben	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
DeleteObjectTagging	Diese Implementierung der DELETE-Operation verwendet die Subressource "tagging", um den kompletten Tag-Satz aus dem angegebenen Objekt zu entfernen.	Markieren	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
DeleteObjectVersion	Zum Entfernen einer bestimmten Version eines Objekts müssen Sie der Bucket-Eigentümer sein und die Subressource versionId verwenden.	Schreiben	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	
DeleteObjectVersionTagging	DELETE Object tagging (for a Specific Version of the Object)	Markieren	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	
DescribeJob	Ruft die Konfigurationsparameter und den Status eines S3 Stapeloperationen-Auftrags ab.	Lesen	job* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
GetAccelerateConfiguration	Diese Implementierung der GET-Operation verwendet die accelerate-Subressource, um den "Transfer Acceleration"-Status eines Buckets – "Enabled" oder "Suspended" – zurückzugeben.	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetAccessPoint	Abrufen der Metadaten des Zugriffspunkts	Lesen		s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetAccessPointPolicy	Geben Sie die Richtlinie eines bestimmten Zugriffspunkts zurück.	Lesen	accesspoint* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetAccessPointPolicyStatus	Abrufen des Richtlinienstatus für die Richtlinie eines bestimmten Zugriffspunkts	Lesen	accesspoint* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
GetAccountPublicAccessBlock	Ruft die PublicAccessBlock-Konfiguration für ein AWS-Konto ab	Lesen		s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetAnalyticsConfiguration	Diese Implementierung der GET-Operation gibt eine analytics-Konfiguration (identifiziert durch die analytics-Konfigurations-ID) aus dem Bucket zurück.	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketAcl	Zugriffskontrollliste (ACL) eines Buckets zurückgeben	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketCORS	Gibt den cors-Konfigurationsinformationssatz für den Bucket zurück	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketLocation	Gibt die Region eines Buckets zurück	Lesen	bucket* (p. 444)		
GetBucketLogging	Protokollierungsstatus eines Buckets und die Berechtigungen zurückgeben, die Benutzer benötigen, um den Status anzuzeigen und zu ändern	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketNotification	Benachrichtigungskonfiguration eines Buckets zurückgeben	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketObjectLockConfiguration	GET Objektsperre-Konfiguration für einen bestimmten Bucket	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446)	
GetBucketPolicy	Richtlinie aus dem angegebenen Bucket zurückgeben	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketPolicyStatus	Ruft den Richtlinienstatus für einen bestimmten S3-Bucket ab, der angibt, ob der Bucket öffentlich ist	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketPublicAccessBlock	Ruft die PublicAccessBlock-Konfiguration für einen bestimmten S3-Bucket ab	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketRequestPayment	Zahlungsanforderungskonfiguration eines Buckets zurückgeben	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketTagging	Den dem Bucket zugeordneten Tag-Satz zurückgeben	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketVersioning	Versioning-Status eines Buckets zurückgeben	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetBucketWebsite	Gibt die einem Bucket zugeordnete Website-Konfiguration zurück	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetEncryptionConfiguration	Gibt die für den Bucket festgelegten Verschlüsselungskonfigurationsinformationen zurück	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
GetInventoryConfiguration	Diese Implementierung der GET-Operation gibt eine Bestandskonfiguration (identifiziert durch die Bestandskonfigurations-ID) aus dem Bucket zurück.	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetLifecycleConfiguration	Gibt die für den Bucket festgelegten Lebenszyklus-Konfigurationsinformationen zurück	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetMetricsConfiguration	Ruft eine Kennzahlenkonfiguration für die CloudWatch-Anforderungskennzahlen (angegeben durch die Kennzahlenkonfigurations-ID) aus dem Bucket ab. Beachten Sie, dass dies nicht die täglichen Speicherkennzahlen umfasst.	Lesen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObject	Ruft Objekte aus Amazon S3 ab	Lesen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectAcl	Zugriffskontrollliste (ACL) eines Objekts zurückgeben	Lesen	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
GetObjectLegalHold	GET Object Legal Hold für ein bestimmtes Objekt	Lesen	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectRetention	GET Object Legal Hold für ein bestimmtes Objekt	Lesen	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
GetObjectTagging	Diese Implementierung der GET-Operation gibt die Tags zurück, die einem Objekt zugeordnet sind. Sie senden die GET-Anforderung für die dem Objekt zugeordnete Subressource "tagging".	Lesen	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectTorrent	Torrent-Dateien aus einem Bucket zurückgeben	Lesen	object* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectVersion	Verwenden Sie die Subressource versionId, um eine andere Version zurückzugeben.	Lesen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectVersionAcl	Verwenden Sie die Subressource versionId, um die ACL-Informationen zu einer anderen Version zurückzugeben.	Lesen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectVersionForReplication	Berechtigung durch S3-Replikation wahrgenommen	Lesen	object* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectVersionTagging	GET Object tagging (for a Specific Version of the Object)	Lesen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetObjectVersionTorrent	Verwenden Sie die Subressource versionId, um die Torrent-Dateien zu einer anderen Version zurückzugeben.	Lesen	object* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	
GetReplicationConfiguration	Gibt die für den Bucket festgelegten Replikationskonfigurationsinformationen zurück.	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListAccessPoints	Listet Zugriffspunkte auf.	Lesen		s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListAllMyBuckets	Gibt eine Liste aller Buckets zurück, deren Eigentümer der authentifizierte Absender der Anforderung ist	Liste		s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListBucket	Gibt einige oder alle (bis zu 1 000) der Objekte in einem Bucket zurück	Liste	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:delimiter (p. 445) s3:max-keys (p. 445) s3:prefix (p. 446) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListBucketMultipartUploads	Listet alle laufenden mehrteiligen Uploads auf.	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListBucketVersions	Verwenden Sie die Subressource "versions", um Metadaten zu allen Versionen der Objekte in einem Bucket aufzulisten.	Lesen	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:delimiter (p. 445) s3:max-keys (p. 445) s3:prefix (p. 446) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListJobs	Listet alle aktuellen Aufträge sowie Aufträge auf, die kürzlich beendet wurden	Lesen		s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ListMultipartUploadParts	Listet die im Rahmen eines bestimmten mehrteiligen Uploads bereits hochgeladenen Teile auf.	Lesen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ObjectOwnerOverrideToBucketOwner	Berechtigung durch S3-Replikationseinstellungen übernommen	Verwaltung von Berechtigungen	object* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutAccelerateConfiguration	Diese Implementierung der PUT-Operation verwendet die Subressource "accelerate", um den "Transfer Acceleration"-Status eines Buckets zurückzugeben.	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
PutAccessPointPolicy	Hinzufügen oder Ersetzen einer Datenrichtlinie auf einem Zugriffspunkt.	Verwaltung von Berechtigungen	accesspoint* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutAccountPublicAccessBlock	Erstellen oder Ändern der PublicAccessBlock-Konfiguration für ein AWS-Konto	Verwaltung von Berechtigungen		s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutAnalyticsConfiguration	Diese Implementierung der PUT-Operation fügt eine Analysekonfiguration (identifiziert durch die Analyse-Konfigurations-ID) in den Bucket ein. Sie können maximal 1 000 Analysekonfigurationen pro Bucket verwenden.	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
PutBucketAcl	Berechtigungen für einen Bucket mit Zugriffskontrolllisten (Access Control Lists, ACLs) festlegen	Verwaltung von Berechtigungen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-grant-full-control (p. 446) s3:x-amz-grant-read (p. 446) s3:x-amz-grant-read-acp (p. 446) s3:x-amz-grant-write (p. 446) s3:x-amz-grant-write-acp (p. 446)	
PutBucketCORS	Legt die CORS-Konfiguration für den Bucket fest	Schreiben	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketLogging	Protokollierungsparameter für einen Bucket festlegen	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketNotification	Ermöglicht das Empfangen von Benachrichtigungen zu bestimmten Ereignissen im Bucket	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketObjectLockConfiguration	PUT Objektsperre-Konfiguration für einen bestimmten Bucket	Schreiben	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446)	
PutBucketPolicy	Richtlinie einem Bucket hinzufügen oder in einem Bucket austauschen	Verwaltung von Berechtigungen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketPublicAccessBlock	Erstellen oder Ändern der PublicAccessBlock-Konfiguration für einen bestimmten S3-Bucket	Verwaltung von Berechtigungen	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketRequestPayment	Zahlungsanforderungskonfiguration eines Buckets festlegen	Schreiben	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketTagging	Einem Bucket einen Satz Tags hinzufügen	Markieren	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketVersioning	Legt den Versioning-Status eines Buckets fest	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutBucketWebsite	Legt die Konfiguration der Website fest, die in der Subressource "website" angegeben ist	Schreiben	bucket* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentypen (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutEncryptionConfiguration	Legt die Verschlüsselungskonfiguration für den Bucket fest	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutInventoryConfiguration	Diese Implementierung der PUT-Operation fügt eine Bestandskonfiguration (identifiziert durch die Bestands-ID) in den Bucket ein. Sie können maximal 1 000 Bestandskonfigurationen pro Bucket verwenden.	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
PutLifecycleConfiguration	Erstellt eine neue Lebenszykluskonfiguration für den Bucket oder ersetzt eine vorhandene Lebenszykluskonfiguration	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutMetricsConfiguration	Legt eine Kennzahlenkonfiguration für die CloudWatch-Anforderungskennzahlen (angegeben durch die Kennzahlenkonfigurations-ID) im Bucket fest oder aktualisiert sie	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutObject	Fügt ein Objekt in einen Bucket ein	Schreiben	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:RequestObjectTag/<key> (p. 445) s3:RequestObjectTagKeys (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-copy-source (p. 446) s3:x-amz-grant-full-control (p. 446) s3:x-amz-grant-read (p. 446) s3:x-amz-grant-read-acp (p. 446) s3:x-amz-grant-write (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:x-amz-grant-write-acp (p. 446) s3:x-amz-metadata-directive (p. 446) s3:x-amz-server-side-encryption (p. 446) s3:x-amz-server-side-encryption-aws-kms-key-id (p. 447) s3:x-amz-storage-class (p. 447) s3:x-amz-website-redirect-location (p. 447) s3:object-lock-mode (p. 446) s3:object-lock-retain-until-date (p. 446) s3:object-lock-remaining-retention-days (p. 446) s3:object-lock-legal-hold (p. 446)	

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
PutObjectAcl	Zugriffskontrolllisten (ACL)-Berechtigungen für ein bereits in einem Bucket vorhandenes Objekt festlegen	Verwaltung von Berechtigungen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-grant-full-control (p. 446) s3:x-amz-grant-read (p. 446) s3:x-amz-grant-read-acp (p. 446) s3:x-amz-grant-write (p. 446) s3:x-amz-grant-write-acp (p. 446) s3:x-amz-storage-	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				class (p. 447)	
PutObjectLegalHold	PUT Object Legal Hold für ein bestimmtes Objekt	Schreiben	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446) s3:object-lock-legal-hold (p. 446)	
PutObjectRetention	PUT Object Retention für ein bestimmtes Objekt	Schreiben	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446) s3:object-lock-mode (p. 446) s3:object-lock-retain-until-date (p. 446) s3:object-lock-remaining-retention-days (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
PutObjectTagging	Diese Implementierung der PUT-Operation verwendet die Subressource "tagging", um einem Objekt einen Satz Tags hinzuzufügen.	Markieren	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:RequestObjectTag/<key> (p. 445) s3:RequestObjectTagKeys (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
PutObjectVersionAcl	Die ACL eines Objekts wird auf Ebene der Objektversion festgelegt.	Verwaltung von Berechtigungen	object* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-acl (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-grant-full-control (p. 446) s3:x-amz-grant-read (p. 446) s3:x-amz-grant-read-acp (p. 446) s3:x-amz-grant-write (p. 446) s3:x-amz-grant-write-acp (p. 446)	

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:x-amz-storage-class (p. 447)	
PutObjectVersionTagging	PUT Object tagging (for a Specific Version of the Object)	Markieren	object* (p. 444)		
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:ExistingObjectTag/<key> (p. 445) s3:RequestObjectTag/<key> (p. 445) s3:RequestObjectTagKeys (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:versionid (p. 446) s3:x-amz-content-sha256 (p. 446)	

Aktionen	Beschreibung	Zugriffsebe	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
PutReplicationConfiguration	In einem "versioning"-fähigen Bucket erstellt diese Operation eine Replikationskonfiguration (oder ersetzt eine ggf. vorhandene Konfiguration).	Schreiben	bucket* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ReplicateDelete	Berechtigung durch S3-Replikation wahrgenommen	Schreiben	object* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
ReplicateObject	Berechtigung durch S3-Replikation wahrgenommen	Schreiben	object* (p. 444)		

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446) s3:x-amz-server-side-encryption (p. 446) s3:x-amz-server-side-encryption-aws-kms-key-id (p. 447)	
ReplicateTags	Berechtigung durch S3-Replikation wahrgenommen	Markieren	object* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
RestoreObject	Stellt eine temporäre Kopie eines archivierten Objekts wieder her	Schreiben	object* (p. 444)		

Aktionen	Beschreibung	Zugriffseben	Ressourcent (*erforderlich	Bedingungss	Abhängige Aktionen
				s3:DataAccessPointAccount (p. 445) s3:DataAccessPointArn (p. 445) s3:AccessPointNetworkOrigin (p. 445) s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446)	
UpdateJobPriority	Aktualisiert die Priorität eines vorhandenen Auftrags	Schreiben	job* (p. 444)		
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446) s3:RequestJobPriority (p. 445) s3:ExistingJobPriority (p. 445) s3:ExistingJobOperation (p. 445)	
UpdateJobStatus	Aktualisiert den Status für den angegebenen Auftrag	Schreiben	job* (p. 444)		

Aktionen	Beschreibung	Zugriffsebene	Ressourcentyp (*erforderlich)	Bedingungsschlüssel	Abhängige Aktionen
				s3:authtype (p. 445) s3:signatureage (p. 446) s3:signatureversion (p. 446) s3:x-amz-content-sha256 (p. 446) s3:ExistingJobPriority (p. 445) s3:ExistingJobOperation (p. 445) s3:JobSuspendedCause (p. 445)	

Von Amazon S3 definierte Ressourcentypen

Die folgenden Ressourcentypen werden von diesem Service definiert und können im Element `Resource` von IAM-Berechtigungsrichtlinienanweisungen verwendet werden. Jede Aktion in der Tabelle "Actions" (Aktionen) (p. 394) identifiziert die Ressourcentypen, die mit der Aktion angegeben werden können. Ein Ressourcentyp kann auch definieren, welche Bedingungsschlüssel Sie in einer Richtlinie einschließen können. Diese Schlüssel werden in der letzten Spalte der Tabelle angezeigt.

Ressourcentypen	ARN	Bedingungsschlüssel
accesspoint	`arn:${Partition}:s3:${Region}:${Account}:accesspoint/${AccessPointName}`	
bucket	`arn:${Partition}:s3:::${BucketName}`	
object	`arn:${Partition}:s3:::${BucketName}/${ObjectName}`	
job	`arn:${Partition}:s3:${Region}:${Account}:job/${JobId}`	

Bedingungsschlüssel für Amazon S3

Amazon S3 definiert die folgenden Bedingungsschlüssel, die im Element `Condition` einer IAM-Richtlinie verwendet werden können. Diese Schlüssel können Sie verwenden, um die Bedingungen zu verfeinern, unter denen die Richtlinienanweisung angewendet wird.

Eine Liste der globalen Bedingungsschlüssel, die für alle Services verfügbar sind, finden Sie unter Verfügbare globale Bedingungsschlüssel in der IAM-Richtlinienreferenz.

Bedingungsschlüsse	Beschreibung	Typ
s3:AccessPointNetworkOrigin	Der Netztyp, aus dem der an der Anforderung beteiligte Zugriffspunkt Datenverkehr empfangen kann	Zeichenfolge
s3:DataAccessPointAccount	Die AWS Konto-ID des Kontos, das den an der Anforderung beteiligten Zugriffspunkt für Datenoperationen besitzt	Zeichenfolge
s3:DataAccessPointArn	Der ARN der an der Anforderung beteiligten Zugriffspunkts	Zeichenfolge
s3:ExistingJobOperation	Filtert den Zugriff auf die Aktualisierung der Aufgabenpriorität nach Operation.	Zeichenfolge
s3:ExistingJobPriority	Filtert den Zugriff auf das Abbrechen vorhandener Aufgaben nach Prioritätsbereich.	Numerischer Wert
s3:ExistingObjectTag/<key>	Ermöglicht das Prüfen, ob ein Objekt-Tag den spezifischen Tag-Schlüssel und -Wert enthält	Zeichenfolge
s3:JobSuspendedCause	Filtert den Zugriff auf das Abbrechen angehaltener Aufgaben nach einem bestimmten Grund für das Anhalten von Aufgaben(z. B. AWAITING_CONFIRMATION).	Zeichenfolge
s3:LocationConstraint	Ermöglicht das Beschränken des Erstellens von Buckets durch Benutzer auf eine bestimmte Region	Zeichenfolge
s3:RequestJobOperation	Filtert den Zugriff auf die Aktualisierung der Aufgabenpriorität nach Operation.	Zeichenfolge
s3:RequestJobPriority	Filtert den Zugriff auf das Erstellen neuer Aufgaben nach Prioritätsbereich.	Numerischer Wert
s3:RequestObjectTag/<key>	Schränkt die Tag-Schlüssel und -Werte ein, die Sie für Objekte zulassen möchten.	Zeichenfolge
s3:RequestObjectTagKeys	Schränkt die Tag-Schlüssel ein, die Sie für Objekte zulassen möchten.	Zeichenfolge
s3:VersionId	Ermöglicht das Beschränken der Berechtigung für die Aktion s3:PutObjectVersionTagging auf eine bestimmte Objektversion.	Zeichenfolge
s3:authtype	Schränkt eingehende Anforderungen auf eine bestimmte Authentifizierungsmethode ein.	Zeichenfolge
s3:delimiter	Ermöglicht das Festlegen der Anforderung, dass der Benutzer den Trennzeichenparameter in der "GET Bucket Object versions"-Anforderung angeben muss.	Zeichenfolge
s3:locationconstraint	Ermöglicht das Beschränken des Benutzers auf das Erstellen eines Buckets in nur einer bestimmten Region	Zeichenfolge
s3:max-keys	Ermöglicht das Beschränken der Anzahl der von Amazon S3 zurückgegebenen Schlüssel in Antworten auf ListBucket-Anforderungen, indem der Benutzer gezwungen wird, den Parameter max-keys anzugeben	Numerischer Wert

Bedingungsschlüsse	Beschreibung	Typ
s3:object-lock-legal-hold	Aktiviert die Erzwingung der gesetzlichen Sperrfrist des angegebenen Objekts	Zeichenfolge
s3:object-lock-mode	Aktiviert die Erzwingung des Aufbewahrungsmodus des angegebenen Objekts	Zeichenfolge
s3:object-lock-remaining-retention-days	Aktiviert die Erzwingung eines Objekts relativ zu den verbleibenden Aufbewahrungstagen	Zeichenfolge
s3:object-lock-retain-until-date	Aktiviert die Erzwingung einer bestimmten Aufbewahrungsfrist	Zeichenfolge
s3:prefix	Ermöglicht das Beschränken der Antwort auf die API-Funktion ListBucket auf Schlüsselnamen mit einem bestimmten Präfix	Zeichenfolge
s3:signatureage	Gibt die Länge der Zeit in Millisekunden an, die eine Signatur in einer authentifizierten Anforderung gültig ist.	Numerischer Wert
s3:signatureversion	Gibt die Version von AWS Signature an, die für authentifizierte Anforderungen unterstützt wird.	Zeichenfolge
s3:versionid	Filtert den Zugriff nach einer bestimmten Objektversion.	Zeichenfolge
s3:x-amz-acl	Ermöglicht das Voraussetzen bestimmter Zugriffsberechtigungen, wenn ein Objekt hochgeladen wird	Zeichenfolge
s3:x-amz-content-sha256	Lässt für Ihren Bucket keine nicht signierten Inhalte zu.	Zeichenfolge
s3:x-amz-copy-source	Ermöglicht das Beschränken der Kopierquelle auf einen bestimmten Bucket, einen bestimmten Ordner im Bucket oder ein bestimmtes Objekt in einem Bucket	Zeichenfolge
s3:x-amz-grant-full-control	Erfordert den Header „x-amz-grant-full-control“ (vollständige Kontrolle) in einer Anforderung.	Zeichenfolge
s3:x-amz-grant-read	Erfordert den Header „x-amz-grant-read“ (Lesezugriff) in einer Anforderung.	Zeichenfolge
s3:x-amz-grant-read-acp	Erfordert den Header „x-amz-grant-read-acp“ (Leseberechtigungen für die ACL) in einer Anforderung.	Zeichenfolge
s3:x-amz-grant-write	Erfordert den Header „x-amz-grant-write“ (Schreibzugriff) in einer Anforderung.	Zeichenfolge
s3:x-amz-grant-write-acp	Erfordert den Header „x-amz-grant-write-acp“ (Schreibberechtigungen für die ACL) in einer Anforderung.	Zeichenfolge
s3:x-amz-metadata-directive	Ermöglicht das Erzwingen bestimmter Verhaltensweisen (KOPIEREN vs. ERSETZEN), wenn Objekte hochgeladen werden	Zeichenfolge
s3:x-amz-server-side-encryption	Ermöglicht es, vom Benutzer das Angeben dieses Headers zu fordern, um sicherzustellen, dass vom Benutzer hochgeladene Objekte beim Speichern verschlüsselt werden	Zeichenfolge

Bedingungsschlüsse	Beschreibung	Typ
s3:x-amz-server-side-encryption-aws-kms-key-id	Erfordert einen bestimmten AWS KMS-kundenverwalteten CMK für serverseitige Verschlüsselung.	Zeichenfolge
s3:x-amz-storage-class	Filtert den Zugriff nach Speicherklasse.	Zeichenfolge
s3:x-amz-website-redirect-location	Filtert den Zugriff nach einem bestimmten Website-Umleitungsort.	Zeichenfolge

Beispiele für Bucket-Richtlinien

Dieser Abschnitt zeigt einige Beispiele für typische Anwendungsfälle für Bucket-Richtlinien. Die Richtlinien verwenden die Zeichenfolgen *`bucket`* und *`examplebucket`* im Ressourcenwert. Um diese Richtlinien zu testen, ersetzen Sie diese Zeichenfolge durch Ihren Bucket-Namen. Informationen zur Sprache der Zugriffsrichtlinie finden Sie unter Richtlinien und Berechtigungen in Amazon S3 (p. 376).

> **Note**
>
> Bucket-Richtlinien sind auf eine Größe von 20 KB beschränkt.

Sie können den AWS Policy Generator verwenden, um eine Bucket-Richtlinie für Ihren Amazon S3-Bucket zu erstellen. Sie können dann das generierte Dokument verwenden, um Ihre Bucket-Richtlinie festzulegen, indem Sie die Amazon S3-Konsole, verschiedene Drittanbieterwerkzeuge oder Ihre Anwendung verwenden.

> **Important**
>
> Beim Testen von Berechtigungen unter Verwendung der Amazon S3 müssen Sie der Konsole zusätzliche Berechtigungen erteilen, die die Konsole benötigt – `s3:ListAllMyBuckets`, `s3:GetBucketLocation` und `s3:ListBucket`. Ein detailliertes Beispiel für eine Richtlinie, die Berechtigungen für Benutzer erteilt und sie unter Verwendung der Konsole testet, finden Sie unter Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien (p. 461).

Themen

Berechtigungen an mehrere Konten mit zusätzlichen Bedingungen erteilen

Die folgende Beispielrichtlinie gewährt die Berechtigungen `s3:PutObject` und `s3:PutObjectAcl` für mehrere AWS-Konten und erfordert, dass jede Anforderung für diese Operationen die `public-read`-Zugriffskontrollliste (ACL) enthält. Weitere Informationen finden Sie unter Amazon S3-Aktionen (p. 380) und Amazon S3-Bedingungsschlüssel (p. 382).

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AddCannedAcl",
      "Effect":"Allow",
      "Principal": {"AWS":
 ["arn:aws:iam::111122223333:root","arn:aws:iam::444455556666:root"]},
      "Action":["s3:PutObject","s3:PutObjectAcl"],
      "Resource":["arn:aws:s3:::examplebucket/*"],
      "Condition":{"StringEquals":{"s3:x-amz-acl":["public-read"]}}
    }
  ]
}
```

Erteilen von Leseberechtigungen an einen anonymen Benutzer

Die folgende Beispielrichtlinie erteilt allen öffentlichen anonymen Benutzern die `s3:GetObject`-Berechtigung. (Eine Liste der Berechtigungen und der von ihnen zugelassenen Operationen finden Sie unter Amazon S3-Aktionen (p. 380).) Diese Berechtigung gestattet jedem, die Objektdaten zu lesen, was praktisch ist, wenn Sie Ihren Bucket als Website konfigurieren und wollen, dass jeder die Objekte in Ihrem Bucket lesen kann. Bevor Sie eine Bucket-Richtlinie verwenden, um einem anonymen Benutzer Leserechte zu gewähren, müssen Sie die Einstellungen für den öffentlichen Zugriff für Ihren Bucket deaktivieren. Weitere Informationen finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff (p. 608).

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"PublicRead",
      "Effect":"Allow",
      "Principal": "*",
      "Action":["s3:GetObject"],
      "Resource":["arn:aws:s3:::examplebucket/*"]
    }
  ]
}
```

> **Warning**
>
> Seien Sie vorsichtig, wenn Sie anonymen Zugriff auf Ihren Amazon S3-Bucket gewähren oder die Einstellungen für den öffentlichen Zugriff deaktivieren. Wenn Sie anonymen Zugriff gewähren, kann jeder auf der ganzen Welt auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, niemals anonymen Zugriff auf Ihren Amazon S3-Bucket zu gewähren, es sei denn, Sie müssen dies ausdrücklich tun, z. B. beim Hosting von statischen Websites (p. 602).

Beschränken des Zugriffs auf bestimmte IP-Adressen

Im folgenden Beispiel wird jedem Benutzer die Berechtigungen zum Ausführen von Amazon S3-Operationen an Objekten im angegebenen S3-Bucket verweigert, es sei denn, die Anforderung stammt aus dem in der Bedingung angegebenen IP-Adressbereich.

Die Anweisung identifiziert 54.240.143.0/24 als Bereich zulässiger Internet Protocol Version 4 (IPv4)-IP-Adressen.

Der `Condition`-Block verwendet die Bedingungen `NotIpAddress` und den Bedingungsschlüssel `aws:SourceIp`, wobei es sich um einen AWS-übergreifenden Bedingungsschlüssel handelt. Weitere Informationen über diese Bedingungsschlüssel finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382). Die `aws:SourceIp` IPv4-Werte verwenden die CIDR-Standardnotation.

Weitere Informationen finden Sie unter Referenz zu IAM JSON-Richtlinienelementen in der IAM-Benutzerhandbuch.

> **Important**
>
> Ersetzen Sie den IP-Adressbereich in diesem Beispiel durch einen geeigneten Wert für Ihren Anwendungsfall, bevor Sie diese Richtlinie verwenden. Andernfalls verlieren Sie die Möglichkeit, auf Ihren Bucket zuzugreifen.

```
{
  "Version": "2012-10-17",
  "Id": "S3PolicyId1",

  "Statement": [

    {
      "Sid": "IPAllow",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
         "NotIpAddress": {"aws:SourceIp": "54.240.143.0/24"}

      }
    }
  ]
}
```

IPv4- und IPv6-Adressen zulassen

Wenn Sie mit der Verwendung von IPv6-Adressen beginnen, sollten Sie alle Richtlinien Ihrer Organisation zusätzlich zu den bereits vorhandenen IPv4-Adressbereichen mit Ihren IPv6-Adressbereichen aktualisieren, um die Funktion Ihrer Richtlinien während der Umstellung auf IPv6 sicherzustellen.

Das folgende Beispiel für eine Bucket-Richtlinie zeigt, wie Sie IPv4- und IPv6-Adressbereiche kombinieren können, um alle gültigen IP-Adressen in Ihrer Organisation abzudecken. Die Beispielrichtlinie würde Zugriff auf die IP-Adressen `54.240.143.1` und `2001:DB8:1234:5678::1` des Beispiels erteilen und den Zugriff auf die Adressen `54.240.143.129` und `2001:DB8:1234:5678:ABCD::1` verweigern.

Die IPv6-Werte für `aws:SourceIp` müssen im CIDR-Standardformat angegeben werden. Für IPv6 unterstützen wir die Verwendung von `::` zur Darstellung eines Buckets von 0s (z. B. `2032001:DB8:1234:5678::/64`). Weitere Informationen finden Sie unter Bedingungsoperatoren für IP-Adressen im IAM-Benutzerhandbuch.

> **Important**
>
> Ersetzen Sie die IP-Adressbereiche in diesem Beispiel durch geeignete Werte für Ihren Anwendungsfall, bevor Sie diese Richtlinie verwenden. Andernfalls verlieren Sie möglicherweise die Möglichkeit, auf Ihren Bucket zuzugreifen.

```
{
  "Id":"PolicyId2",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AllowIPmix",
      "Effect":"Allow",
      "Principal":"*",
      "Action":"s3:*",
      "Resource":"arn:aws:s3:::examplebucket/*",
```

```
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": [
            "54.240.143.0/24",
            "2001:DB8:1234:5678::/64"
          ]
        },
        "NotIpAddress": {
          "aws:SourceIp": [
             "54.240.143.128/30",
             "2001:DB8:1234:5678:ABCD::/80"
          ]
        }
      }
    }
  ]
}
```

Beschränken des Zugriffs auf einen spezifischen HTTP-Referrer

Angenommen, Sie haben eine Website mit einem Domänennamen (`www.example.com` oder `example.com`) mit Links zu Fotos und Videos, die in Ihrem Amazon S3-Bucket (`examplebucket`) gespeichert sind. Standardmäßig sind alle Amazon S3-Ressourcen privat, sodass nur das AWS-Konto, das die Ressourcen erstellt hat, auf sie zugreifen kann. Um Lesezugriff auf diese Objekte von Ihrer Website aus zu erlauben, können Sie eine Bucket-Richtlinie hinzufügen, die mit dem `s3:GetObject`-Schlüssel eine `aws:Referer`-Berechtigung mit der Bedingung vergibt, dass die Get-Anforderung von bestimmten Webseiten kommen muss. Die folgende Richtlinie spezifiziert die `StringLike`-Bedingung mit dem `aws:Referer`-Bedingungsschlüssel.

```
{
  "Version":"2012-10-17",
  "Id":"http referer policy example",
  "Statement":[
    {
      "Sid":"Allow get requests originating from www.example.com and example.com.",
      "Effect":"Allow",
      "Principal":"*",
      "Action":"s3:GetObject",
      "Resource":"arn:aws:s3:::examplebucket/*",
      "Condition":{
        "StringLike":{"aws:Referer":["http://www.example.com/*","http://example.com/*"]}
      }
    }
  ]
}
```

Stellen Sie sicher, dass die von Ihnen verwendeten Browser den HTTP `referer`-Header in der Anfrage enthalten.

Erteilen der Berechtigung für eine Amazon CloudFront OAI

Über das folgende Beispiel einer Bucket-Richtlinie wird eine CloudFront-Ursprungszugriffsidentitätsberechtigung (Origin Access Identity, OAI) erteilt, um alle Objekte in Ihrem Amazon S3 abzurufen (zu lesen). Sie können eine CloudFront-OAI verwenden, um Benutzern den Zugriff auf Objekte in Ihrem Bereich über CloudFront, aber nicht direkt über Amazon S3 zu ermöglichen. Weitere Informationen finden Sie unter Einschränken des Zugriffs auf Amazon S3-Inhalte durch Verwenden einer Ursprungszugriffsidentität im Entwicklerhandbuch für Amazon CloudFront.

Bei der folgenden Richtlinie wird die ID der OAI als `Principal` der Richtlinie verwendet. Weitere Informationen zum Verwenden von S3-Bucket-Richtlinien zum Erteilen des Zugriffs auf eine CloudFront-

OAI finden Sie unter Verwenden von Amazon S3-Bucket-Richtlinien im Entwicklerhandbuch für Amazon CloudFront.

Zur Verwendung dieses Beispiels gehen Sie wie folgt vor:

- Ersetzen Sie *EH1HDMB1FH2TC* durch die ID der OAI. Die ID der OAI finden Sie auf der Seite „Origin Access Identity“ (Ursprungszugriffsidentität) in der CloudFront-Konsole oder unter ListCloudFrontOriginAccessIdentities in der CloudFront-API.
- Ersetzen Sie *aws-example-bucket* durch den Namen Ihres Amazon S3-Buckets.

```
{
    "Version": "2012-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity EH1HDMB1FH2TC"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::aws-example-bucket/*"
        }
    ]
}
```

Hinzufügen einer Bucket-Richtlinie zur Anforderung einer MFA

Amazon S3 unterstützt den MFA-geschützten API-Zugriff. Mittels dieser Funktion können Sie eine Multifaktor-Authentifizierung (MFA) für den Zugriff auf Ihre Amazon S3-Ressourcen durchsetzen. Die Multifaktor-Authentifizierung bietet ein zusätzliches Sicherheitsniveau, das Sie auf Ihre AWS-Umgebung anwenden können. Es handelt sich um eine Sicherheitsfunktion, die die Angabe eines gültigen MFA-Codes von Benutzern erfordert, mit dem das physische Eigentum eines MFA-Geräts belegt wird. Weitere Informationen finden Sie unter AWS-Multifaktor-Authentifizierung. Sie können die MFA für alle Anfragen zum Zugriff auf Ihre Amazon S3-Ressourcen vorschreiben.

Sie können die MFA-Anforderung mit dem `aws:MultiFactorAuthAge`-Schlüssel in einer Bucket-Richtlinie durchsetzen. AWS Identity and Access Management (IAM) Benutzer können auf Amazon S3-Ressourcen zugreifen, indem sie temporäre Anmeldeinformationen verwenden, die von AWS Security Token Service (AWS STS) ausgegeben werden. Sie geben den MFA-Code zum Zeitpunkt der AWS STS-Anfrage an.

Wenn Amazon S3 eine Anforderung mit Multifaktor-Authentifizierung empfängt, liefert der `aws:MultiFactorAuthAge`-Schlüssel einen numerischen Wert, der angibt, wie lange (in Sekunden) es her ist, dass die temporären Anmeldeinformationen erstellt wurden. Wenn die in der Anforderung bereitgestellten temporären Anmeldeinformationen nicht mit einem MFA-Gerät erstellt wurden, ist dieser Schlüsselwert null (nicht vorhanden). In einer Bucket-Richtlinie können Sie eine Bedingung hinzufügen, um diesen Wert zu überprüfen, wie im folgenden Beispiel für eine Bucket-Richtlinie gezeigt. Die Richtlinie verweigert jede Amazon S3-Operation zum `/taxdocuments`-Ordner im `examplebucket`-Bucket, wenn die Anforderung nicht über die MFA authentifiziert wird. Um mehr über die MFA zu erfahren, lesen Sie Multifaktor-Authentifizierung (MFA) in AWS in IAM-Benutzerhandbuch.

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
```

```
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::examplebucket/taxdocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true }}
      }
    ]
 }
```

Die `Null`-Bedingung im `Condition`-Block wird zu true ausgewertet, wenn der `aws:MultiFactorAuthAge`-Schlüsselwert null ist, d. h. die temporären Anmeldeinformationen in der Anforderung wurden ohne den MFA-Schlüssel erstellt.

Die folgende Bucket-Richtlinie ist eine Erweiterung der vorhergehenden Bucket-Richtlinie. Sie beinhaltet zwei Richtlinienanweisungen. Eine Anweisung gewährt jedem die `s3:GetObject`-Berechtigung für einen Bucket (`examplebucket`). Eine weitere Anweisung schränkt den Zugriff auf den `examplebucket/taxdocuments`-Ordner im Bucket weiter ein, indem sie eine MFA erzwingt.

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::examplebucket/taxdocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true } }
      },
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": "*",
        "Action": ["s3:GetObject"],
        "Resource": "arn:aws:s3:::examplebucket/*"
      }
    ]
 }
```

Optional können Sie eine numerische Bedingung verwenden, um die Dauer zu beschränken, für die der `aws:MultiFactorAuthAge`-Schlüssel gültig ist, unabhängig von der Lebensdauer der temporären Sicherheitsanmeldeinformationen, die für die Authentifizierung der Anforderung verwendet wurden. Beispielsweise überprüft die folgende Bucket-Richtlinie zusätzlich zur geforderten MFA-Authentifizierung auch, vor wie langer Zeit die temporäre Sitzung erstellt wurde. Die Richtlinie verweigert jede Operation, wenn der `aws:MultiFactorAuthAge`-Schlüsselwert angibt, dass die temporäre Sitzung vor mehr als einer Stunde erstellt wurde (3600 Sekunden).

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::examplebucket/taxdocuments/*",
        "Condition": {"Null": {"aws:MultiFactorAuthAge": true }}
      },
      {
        "Sid": "",
        "Effect": "Deny",
```

```
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::examplebucket/taxdocuments/*",
        "Condition": {"NumericGreaterThan": {"aws:MultiFactorAuthAge": 3600 }}
      },
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": "*",
        "Action": ["s3:GetObject"],
        "Resource": "arn:aws:s3:::examplebucket/*"
      }
   ]
}
```

Erteilung von kontenübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt

Sie können einem anderen AWS-Konto erlauben, Objekte in Ihren Bucket hochzuladen. Sie können jedoch entscheiden, dass Sie als Bucket-Eigentümer die volle Kontrolle über die in Ihren Bucket hochgeladenen Objekte haben müssen. Die folgende Richtlinie erzwingt, dass einem bestimmten AWS-Konto (`111111111111`) die Möglichkeit zum Hochladen von Objekten verweigert wird, es sei denn, dieses Konto gewährt dem durch die E-Mail-Adresse (`xyz@amazon.com`) identifizierten Bucket-Eigentümer Vollzugriff. Die `StringNotEquals`-Bedingung in der Richtlinie spezifiziert den `s3:x-amz-grant-full-control`-Bedingungsschlüssel, um die Anforderung auszudrücken (siehe Amazon S3-Bedingungsschlüssel (p. 382)).

```
{
   "Version":"2012-10-17",
   "Statement":[
     {
       "Sid":"111",
       "Effect":"Allow",
       "Principal":{"AWS":"1111111111"},
       "Action":"s3:PutObject",
       "Resource":"arn:aws:s3:::examplebucket/*"
     },
     {
       "Sid":"112",
       "Effect":"Deny",
       "Principal":{"AWS":"1111111111" },
       "Action":"s3:PutObject",
       "Resource":"arn:aws:s3:::examplebucket/*",
       "Condition": {
         "StringNotEquals": {"s3:x-amz-grant-full-control":["emailAddress=xyz@amazon.com"]}
       }
     }
   ]
}
```

Gewähren von Berechtigungen für Amazon S3-Bestand und Amazon S3-Analysen

Der Amazon S3-Bestand erstellt Listen der Objekte in einem Amazon S3-Bucket, und der Amazon S3-Analyseexport erstellt Ausgabedateien der in der Analyse verwendeten Daten. Der Bucket, dessen Objekte die Bestandserfassung auflistet, wird als Quell-Bucket bezeichnet. Die Buckets, in die die Bestandsdatei und die Analyseexportdatei geschrieben werden, werden als Ziel-Buckets bezeichnet. Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket erstellen, wenn Sie den Bestand für einen Amazon S3-Bucket und den Analyseexport einrichten. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503) und Amazon S3-Analyse – Speicherklassenanalyse (p. 282).

Das folgende Beispiel für eine Bucket-Richtlinie erteilt die Amazon S3-Berechtigung, Objekte aus dem Konto für den Quell-Bucket in den Ziel-Bucket zu schreiben (PUTs). Sie verwenden eine solche Bucket-Richtlinie für den Ziel-Bucket, wenn Sie einen Amazon S3-Bestand und einen Amazon S3-Analyseexport einrichten.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"InventoryAndAnalyticsExamplePolicy",
      "Effect":"Allow",
      "Principal": {"Service": "s3.amazonaws.com"},
      "Action":["s3:PutObject"],
      "Resource":["arn:aws:s3:::destination-bucket/*"],
      "Condition": {
          "ArnLike": {
              "aws:SourceArn": "arn:aws:s3:::source-bucket"
           },
          "StringEquals": {
              "aws:SourceAccount": "1234567890",
              "s3:x-amz-acl": "bucket-owner-full-control"
           }
       }
    }
  ]
}
```

Beispiel für Bucket-Richtlinien für VPC-Endpunkte für Amazon S3

Sie können den Zugriff auf Buckets über bestimmte Amazon Virtual Private Cloud (Amazon VPC)-Endpunkte oder bestimmte VPCs über Amazon S3-Bucket-Richtlinien steuern. Dieser Abschnitt enthält exemplarische Bucket-Richtlinien, mit denen der Zugriff auf Amazon S3-Buckets von VPC-Endpunkten aus gesteuert werden kann. Informationen zum Einrichten von VPC-Endpunkten finden Sie unter VPC-Endpunkte im Amazon VPC Benutzerhandbuch.

Amazon VPC ermöglicht es Ihnen, AWS-Ressourcen in einem von Ihnen definierten virtuellen Netzwerk zu starten. Ein VPC-Endpunkt ermöglicht es Ihnen, eine private Verbindung zwischen Ihrer VPC und einem anderen AWS-Service herzustellen, ohne dass Sie über das Internet, über eine VPN-Verbindung, über eine NAT-Instance oder über AWS Direct Connect Zugang benötigen.

Ein VPC-Endpunkt für Amazon S3 ist eine logische Entity innerhalb einer VPC, die ausschließlich Verbindungen mit Amazon S3 zulässt. Der VPC-Endpunkt leitet Anforderungen an Amazon S3 weiter und die Antworten zurück an die VPC. VPC-Endpunkte verändern ausschließlich die Art der Weiterleitung von Anforderungen. Öffentliche Amazon S3-Endpunkte und DNS-Namen funktionieren für VPC-Endpunkte weiterhin. Wichtige Informationen zur Verwendung von Amazon VPC-Endpunkten mit Amazon S3 finden Sie unter VPC-Gateway-Endpunkte und Endpunkte für Amazon S3 im Amazon VPC Benutzerhandbuch.

VPC-Endpunkte für Amazon S3 unterstützen die Kontrolle des Zugriffs auf Ihre Amazon S3-Daten auf zwei Arten:

- Sie können steuern, welche Anfragen, Benutzer oder Gruppen durch einen spezifischen VPC-Endpunkt erlaubt sind. Informationen zu dieser Art der Zugriffssteuerung finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten in der Amazon VPC Benutzerhandbuch.
- Sie können steuern, welche VPCs oder VPC-Endpunkte Zugriff auf Ihre Buckets haben, indem Sie Amazon S3-Bucket-Richtlinien verwenden. Beispiele für diese Art Zugriffssteuerung durch Bucket-Richtlinienfinden Sie in den folgenden Themen zur Zugriffsbeschränkung.

Themen

Important

Wenn Sie die in diesem Abschnitt beschriebenen Amazon S3-Bucket-Richtlinien für VPC-Endpunkte anwenden, können Sie Ihren Zugriff auf den Bucket blockieren, ohne dies zu beabsichtigen. Bucket-Berechtigungen, die den Bucket-Zugriff auf Verbindungen, die von Ihrem VPC-Endpunkt ausgehen, gezielt einschränken sollen, können alle Verbindungen zum Bucket blockieren. Informationen zur Behebung dieses Problems finden Sie unter Meine Bucket-Richtlinie hat die falsche VPC- oder VPC-Endpunkt-ID. Wie kann ich die Richtlinie so ändern, dass ich auf den Bucket zugreifen kann? im AWS Support Knowledge Center.

Beschränken des Zugriffs auf einen bestimmten VPC-Endpunkt

Das Folgende ist ein Beispiel für eine Amazon S3-Bucket-Richtlinie, die den Zugriff auf einen bestimmten Bucket (`awsexamplebucket1`) nur vom VPC-Endpunkt mit der ID `vpce-1a2b3c4d` aus beschränkt. Die Richtlinie lehnt sämtlichen Zugriff auf den Bucket ab, der nicht über den angegebenen Endpunkt erfolgt. Der Endpunkt wird über die `aws:sourceVpce`-Bedingung festgelegt. Die `aws:sourceVpce`-Bedingung erfordert keinen Amazon-Ressourcennamen (ARN) für die VPC-Endpunkt-Ressource, sondern nur die VPC-Endpunkt-ID. Weitere Informationen über die Verwendung von Bedingungen in einer Richtlinie finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382).

Important

- Bevor Sie die folgende Beispielrichtlinie verwenden, ersetzen Sie die VPC-Endpunkt-ID durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls können Sie nicht auf Ihren Bucket zugreifen.
- Diese Richtlinie deaktiviert den Konsolenzugriff auf den angegebenen Bucket, da Konsolenanforderungen nicht vom angegebenen VPC-Endpunkt stammen.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1415115909152",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPCE-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::awsexamplebucket1",
                    "arn:aws:s3:::awsexamplebucket1/*"],
       "Condition": {
         "StringNotEquals": {
           "aws:sourceVpce": "vpce-1a2b3c4d"
         }
       }
     }
   ]
}
```

Beschränkung des Zugriffs auf eine bestimmte VPC

Sie können eine Bucket-Richtlinie mit der `aws:sourceVpc`-Bedingung erstellen, die den Zugriff auf eine bestimmte VPC beschränkt. Dies ist hilfreich, wenn Sie mehrere VPC-Endpunkte in der gleichen VPC konfiguriert haben und den Zugriff auf Ihre Amazon S3-Buckets für alle Ihre Endpunkte verwalten möchten. Nachfolgend finden Sie eine Beispielrichtlinie, die der VPC `vpc-111bbb22` den Zugriff auf `awsexamplebucket1` und dessen Objekte gewährt. Die Richtlinie lehnt sämtlichen Zugriff auf den Bucket

ab, der nicht über die angegebene VPC erfolgt. Für den `vpc-111bbb22`-Bedingungsschlüssel wird kein ARN für die VPC-Ressource benötigt, sondern nur die VPC-ID.

Important

- Bevor Sie die folgende Beispielrichtlinie verwenden, ersetzen Sie die VPC-ID durch einen geeigneten Wert für Ihren Anwendungsfall. Andernfalls können Sie nicht auf Ihren Bucket zugreifen.
- Diese Richtlinie deaktiviert den Konsolenzugriff auf den angegebenen Bucket, da Konsolenanforderungen nicht von der angegebenen VPC stammen.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1415115909153",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::awsexamplebucket1",
                    "arn:aws:s3:::awsexamplebucket1/*"],
       "Condition": {
         "StringNotEquals": {
           "aws:sourceVpc": "vpc-111bbb22"
         }
       }
     }
   ]
}
```

Verwandte Ressourcen

- Beispiele für Bucket-Richtlinien (p. 447)
- VPC-Endpunkte im Amazon VPC Benutzerhandbuch

Beispiele für Benutzerrichtlinien

Dieser Abschnitt zeigt mehrere IAM-Benutzerrichtlinien zum Steuern des Benutzerzugriffs auf Amazon S3. Informationen zur Sprache der Zugriffsrichtlinie finden Sie unter Richtlinien und Berechtigungen in Amazon S3 (p. 376).

Die folgenden Beispielrichtlinien funktionieren, wenn Sie sie programmgesteuert testen. Um sie mit der Amazon S3-Konsole verwenden zu können, müssen Sie aber zusätzliche Berechtigungen gewähren, die für die Konsole erforderlich sind. Informationen zur Verwendung von Richtlinien wie dieser mit der Amazon S3-Konsole finden Sie unter Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien (p. 461).

Themen

- Einem IAM-Benutzer den Zugriff auf einen Ihrer Buckets erlauben (p. 457)
- Jedem IAM-Benutzer Zugriff auf einen Ordner in einem Bucket erlauben (p. 457)
- Einer Gruppe erlauben, einen freigegebenen Ordner in Amazon S3 zu haben (p. 460)
- Allen Ihren Benutzern erlauben, Objekte in einem Teil des Unternehmens-Buckets zu lesen (p. 460)
- Einem Partner erlauben, Dateien in einem bestimmten Bereich des Unternehmens-Buckets abzulegen (p. 461)
- Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien (p. 461)

Einem IAM-Benutzer den Zugriff auf einen Ihrer Buckets erlauben

In diesem Beispiel möchten Sie einem IAM-Benutzer in Ihrem AWS-Konto den Zugriff auf einen Ihrer Buckets (`examplebucket`) gewähren und dem Benutzer erlauben, Objekte hinzuzufügen, zu aktualisieren und zu löschen.

Zusätzlich zum Erteilen der Berechtigungen `s3:PutObject`, `s3:GetObject` und `s3:DeleteObject` für den Benutzer, gewährt die Richtlinie die Berechtigungen `s3:ListAllMyBuckets`, `s3:GetBucketLocation` und `s3:ListBucket`. Dies sind die zusätzlichen Berechtigungen, die von der Konsole benötigt werden. Außerdem sind die Aktionen `s3:PutObjectAcl` und `s3:GetObjectAcl` erforderlich, um Objekte in der Konsole kopieren, ausschneiden und einfügen zu können. Ein detailliertes Beispiel für eine Richtlinie, die Berechtigungen für Benutzer erteilt und sie unter Verwendung der Konsole testet, finden Sie unter Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien (p. 461).

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListAllMyBuckets"
         ],
         "Resource":"arn:aws:s3:::*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetBucketLocation"
         ],
         "Resource":"arn:aws:s3:::examplebucket"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:GetObject",
            "s3:GetObjectAcl",
            "s3:DeleteObject"
         ],
         "Resource":"arn:aws:s3:::examplebucket/*"
      }
   ]
}
```

Jedem IAM-Benutzer Zugriff auf einen Ordner in einem Bucket erlauben

In diesem Beispiel möchten Sie zwei IAM-Benutzern, Alice und Bob, den Zugriff auf Ihren Bucket `examplebucket` gewähren, damit sie Objekte hinzufügen, aktualisieren und löschen können. Allerdings möchten Sie den Zugriff beider Benutzer auf einen einzelnen Ordner im Bucket einschränken. Sie könnten Ordner mit Namen erstellen, die den Benutzernamen entsprechen.

```
examplebucket
   Alice/
   Bob/
```

Um jedem Benutzer den Zugriff nur auf seinen Ordner zu gewähren, können Sie für jeden Benutzer eine Richtlinie schreiben und sie ihnen einzeln zuweisen. Sie können beispielsweise die folgende

Richtlinie der Benutzerin Alice zuweisen, um ihr spezifische Amazon S3-Berechtigungen für den Ordner `examplebucket/Alice` zu gewähren.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::examplebucket/Alice/*"
      }
   ]
}
```

Dann fügen Sie dem Benutzer `Bob` eine ähnliche Richtlinie hinzu, für den Ordner `Bob` im Wert `Resource`.

Statt Richtlinien einzelnen Benutzern zuzuweisen, können Sie auch eine einzelne Richtlinie mit einer Richtlinienvariable schreiben und die Richtlinie einer Gruppe zuweisen. Sie müssen zuerst eine Gruppe erstellen und die Benutzer Alice und Bob dann in die Gruppe aufnehmen. Die folgende Beispielrichtlinie erlaubt eine Reihe von Amazon S3-Berechtigungen für den Ordner `examplebucket/${aws:username}`. Wenn die Richtlinie ausgewertet wird, wird die Richtlinienvariable `${aws:username}` durch den Benutzernamen des Anforderers ersetzt. Wenn Alice beispielsweise eine Anforderung zum Anlegen eines Objekts sendet, ist die Operation nur zulässig, wenn Alice das Objekt in den Ordner `examplebucket/Alice` hochlädt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::examplebucket/${aws:username}/*"
      }
   ]
}
```

> **Note**
>
> Bei der Verwendung von Richtlinienvariablen müssen Sie explizit die Version `2012-10-17` in der Richtlinie angeben. Die Standardversion der Zugriffsrichtliniensprache, 2008-10-17, unterstützt keine Richtlinienvariablen.

Wenn Sie die vorherige Richtlinie auf der Amazon S3;-Konsole testen möchten, erfordert die Konsole die Berechtigung für zusätzliche Amazon S3-Berechtigungen, wie in der folgenden Richtlinie gezeigt. Weitere Informationen darüber, wie die Konsole diese Berechtigungen verwendet, finden Sie unter Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien (p. 461).

```
{
```

```
 "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGroupToSeeBucketListInTheConsole",
      "Action": [ "s3:ListAllMyBuckets", "s3:GetBucketLocation" ],
      "Effect": "Allow",
      "Resource": [ "arn:aws:s3:::*"  ]
    },
    {
      "Sid": "AllowRootLevelListingOfTheBucket",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::examplebucket"],
      "Condition":{
            "StringEquals":{
                    "s3:prefix":[""], "s3:delimiter":["/"]
                           }
                 }
    },
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::examplebucket"],
      "Condition":{  "StringLike":{"s3:prefix":["${aws:username}/*"] }
       }
    },
      {
     "Sid": "AllowUserSpecificActionsOnlyInTheSpecificUserPrefix",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::examplebucket/${aws:username}/*"
      }
  ]
}
```

Note

In der 2012-10-17-Version der Richtlinie beginnen Richtlinienvariablen mit `$`. Diese Syntaxänderung kann möglicherweise einen Konflikt verursachen, wenn Ihr Objektschlüssel einen `$` enthält. Um beispielsweise einen Objektschlüssel `my$file` in eine Richtlinie aufzunehmen, geben Sie das Zeichen `$` mit `${$}`, `my${$}file` an.

Obwohl IAM-Benutzernamen freundliche, von Menschen lesbare Bezeichner sind, müssen sie global nicht eindeutig sein. Wenn beispielsweise der Benutzer Bob die Organisation verlässt, und ein anderer Bob beitritt, könnte der neue Bob auf die alten Informationen von Bob zugreifen. Anstatt Benutzernamen zu verwenden, könnten Sie Ordner erstellen, die auf Benutzer-IDs basieren. Jede Benutzer-ID muss eindeutig sein. In diesem Fall müssen Sie die vorherige Richtlinie ändern, und die Richtlinienvariable `${aws:userid}` verwenden. Weitere Informationen zu den Benutzerkennungen finden Sie unter IAM-Kennungen im IAM-Benutzerhandbuch.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
```

```
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::my_corporate_bucket/home/${aws:userid}/*"
      }
   ]
}
```

Nicht-IAM-Benutzern (Nutzern mobiler Apps) den Zugriff auf Ordner in einem Bucket erlauben

Angenommen, Sie möchten eine mobile App entwickeln, ein Spiel, in dem Benutzerdaten in einem S3-Bucket gespeichert werden. Sie möchten für jeden App-Benutzer einen Ordner in Ihrem Bucket erstellen. Sie möchten auch den Zugriff jedes Benutzers auf seinen eigenen Ordner beschränken. Sie können jedoch keine Ordner erstellen, bevor jemand Ihre App herunterlädt und das Spiel beginnt, weil Sie keine Benutzer-ID haben.

In diesem Fall können Sie von den Nutzern verlangen, dass Sie sich in Ihrer App über einen öffentlichen Identitätsanbieter anmelden, wie z. B. mit der Anmeldung bei Amazon, Facebook oder Google. Nachdem sich Benutzer über einen dieser Anbieter bei Ihrer App angemeldet haben, verfügen sie über eine Benutzer-ID, mit der Sie zur Laufzeit benutzerspezifische Ordner erstellen können.

Sie können dann den Web-Identitätsverbund in AWS Security Token Service verwenden, um Informationen vom Identitätsanbieter in Ihre App zu integrieren und um temporäre Sicherheitsanmeldeinformationen für jeden Benutzer zu erhalten. Sie können dann IAM-Richtlinien erstellen, die es der App ermöglichen, auf Ihren Bucket zuzugreifen und solche Vorgänge wie das Erstellen von benutzerspezifischen Ordnern und das Hochladen von Daten durchzuführen. Weitere Informationen zum Web-Identitätsverbund finden Sie unter Über den Web-Identitätsverbund im IAM-Benutzerhandbuch.

Einer Gruppe erlauben, einen freigegebenen Ordner in Amazon S3 zu haben

Durch Anfügen der folgenden Richtlinie an die Gruppe erhalten alle Benutzer der Gruppe Zugriff auf den folgenden Ordner in Amazon S3: `my_corporate_bucket/share/marketing`. Gruppenmitglieder dürfen nur auf die spezifischen Amazon S3-Berechtigungen zugreifen, die in der Richtlinie gegeben sind, und nur für Objekte im angegebenen Ordner.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::my_corporate_bucket/share/marketing/*"
      }
   ]
}
```

Allen Ihren Benutzern erlauben, Objekte in einem Teil des Unternehmens-Buckets zu lesen

In diesem Beispiel erstellen Sie eine Gruppe namens `AllUsers`, die alle IAM-Benutzer enthält, die dem AWS-Konto gehören. Anschließend fügen Sie eine Richtlinie hinzu, die der Gruppe Zugriff auf `GetObject`

und `GetObjectVersion` gewährt, jedoch nur für Objekte im Ordner `my_corporate_bucket/readonly`.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Resource":"arn:aws:s3:::my_corporate_bucket/readonly/*"
      }
   ]
}
```

Einem Partner erlauben, Dateien in einem bestimmten Bereich des Unternehmens-Buckets abzulegen

In diesem Beispiel erstellen Sie eine Gruppe namens `WidgetCo`, die eine Partnerfirma darstellt. Sie erstellen einen IAM-Benutzer für die bestimmte Person oder Anwendung bei der Partnerfirma, die Zugriff benötigt, und dann setzen Sie den Benutzer in die Gruppe ein.

Sie fügen dann eine Richtlinie hinzu, die der Gruppe den Zugriff `PutObject` auf den folgenden Ordner im Unternehmens-Bucket gibt: `my_corporate_bucket/uploads/widgetco`.

Sie möchten verhindern, dass die Gruppe `WidgetCo` etwas anderes mit dem Bucket macht, und deshalb fügen Sie eine Anweisung hinzu, die jedem die Amazon S3-Berechtigungen außer `PutObject` für einer Amazon S3-Ressource im AWS-Konto explizit verweigert. Dieser Schritt ist nur notwendig, wenn an anderer Stelle in Ihrem AWS-Konto eine umfassende Richtlinie verwendet wird, die Benutzern einen breiten Zugriff auf Amazon S3-Ressourcen ermöglicht.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"s3:PutObject",
         "Resource":"arn:aws:s3:::my_corporate_bucket/uploads/widgetco/*"
      },
      {
         "Effect":"Deny",
         "NotAction":"s3:PutObject",
         "Resource":"arn:aws:s3:::my_corporate_bucket/uploads/widgetco/*"
      },
      {
         "Effect":"Deny",
         "Action":"s3:*",
         "NotResource":"arn:aws:s3:::my_corporate_bucket/uploads/widgetco/*"
      }
   ]
}
```

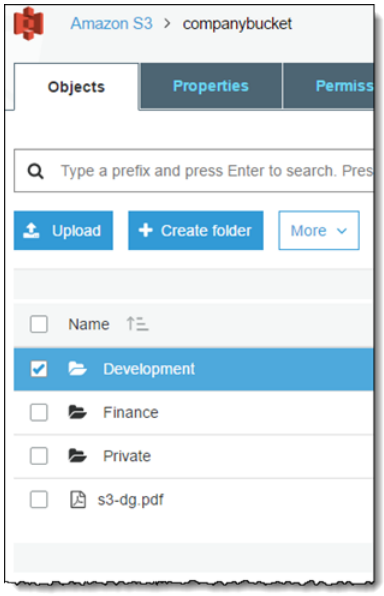
Walkthrough: Kontrollieren des Zugriffs auf einen Bucket mit Benutzerrichtlinien

Diese Anleitung erklärt, wie Benutzerberechtigungen in Amazon S3 funktionieren. In diesem Beispiel erstellen Sie einen Bucket mit Ordnern. Sie können dann in Ihrem AWS-Konto AWS Identity and Access Management (IAM)-Benutzer erstellen und diesen Benutzern inkrementelle Berechtigungen für Ihren Amazon S3-Bucket und die darin enthaltenen Ordner erteilen.

Themen

Grundlagen zu Buckets und Ordnern

Das Amazon S3-Datenmodell ist eine Flatfile-Struktur: Sie erstellen einen Bucket und der Bucket speichert Objekte. Es gibt keine Hierarchie für Unter-Buckets oder Unterordner. Sie können aber eine Ordnerhierarchie nachbilden. Tools, wie die Amazon S3-Konsole, können eine Ansicht dieser logischen Ordner und Unterordner in Ihrem Bucket präsentieren, wie in der folgenden Abbildung dargestellt.

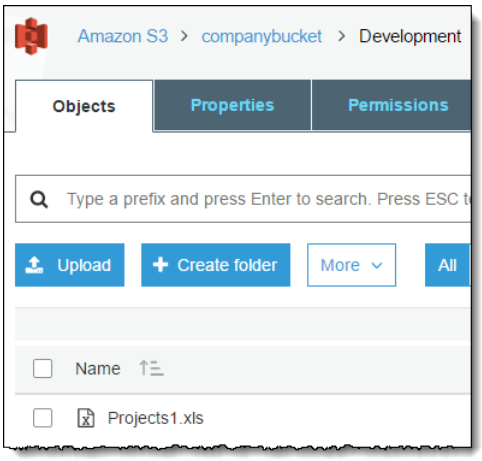


Die Konsole zeigt, dass sich in einem Bucket mit dem Namen `companybucket` die drei Ordner `Private`, `Development` und `Finance` und das Objekt `s3-dg.pdf` befinden. Die Konsole verwendet die Objektnamen (Schlüssel), um eine logische Hierarchie mit Ordnern und Unterordnern zu erzeugen. Betrachten Sie die folgenden Beispiele:

- Wenn Sie den Ordner `Development` anlegen, erstellt die Konsole ein Objekt mit dem Schlüssel `Development/`. Beachten Sie den als Trennzeichen dienenden abschließenden Schrägstrich (`/`).
- Wenn Sie ein Objekt mit dem Namen `Projects1.xls` in den Ordner `Development` hochladen, lädt die Konsole das Objekt hoch und weist ihm den Schlüssel `Development/Projects1.xls` zu.

  Im Schlüssel ist `Development` das Präfix und `/` ist das Trennzeichen. Das Amazon S3-API unterstützt in ihren Operationen Präfixe und Separatoren. Beispielsweise können Sie eine Liste aller Objekte in

einem Bucket mit einem bestimmten Präfix und Separator erhalten. Wenn Sie in der Konsole den Ordner `Development` öffnen, listet die Konsole die Objekte im betreffenden Ordner auf. Im folgenden Beispiel enthält der Ordner `Development` ein Objekt.



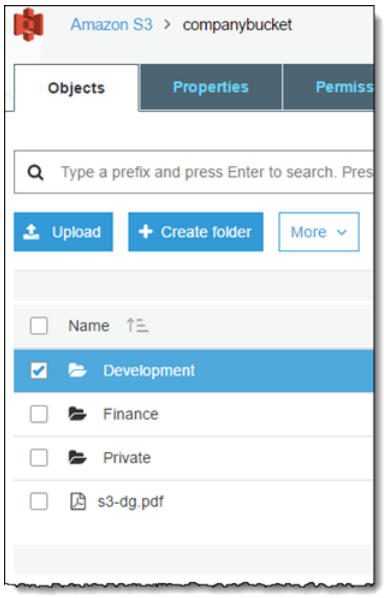
Wenn die Konsole den Ordner `Development` im Bucket `companybucket` auflistet, sendet sie eine Anforderung an Amazon S3, in der das Präfix `Development` und das Trennzeichen `/` in der Anforderung angegeben werden. Die Antwort der Konsole sieht genauso aus, wie eine Ordnerliste im Dateisystem Ihres Computers. Das vorherige Beispiel zeigt, dass der Bucket `companybucket` ein Objekt mit dem Schlüssel `Development/Projects1.xls` enthält.

Die Konsole verwendet Objektschlüssel, um eine logische Hierarchie herzuleiten. Amazon S3 besitzt keine physische Hierarchie, sondern nur Buckets, die Objekte in einer Flatfile-Struktur enthalten. Wenn Sie Objekte mit der Amazon S3-API erstellen, können Sie Objektschlüssel verwenden, die eine logische Hierarchie implizieren. Wenn Sie eine logische Hierarchie von Objekten erstellen, können Sie den Zugriff auf einzelne Ordner verwalten, wie in dieser exemplarischen Anleitung veranschaulicht.

Bevor Sie beginnen, müssen Sie mit dem Konzept des Bucket-Inhalts auf Stammebene vertraut sein. Angenommen, Ihr Bucket `companybucket` enthält die folgenden Objekte:

- `Private/privDoc1.txt`
- `Private/privDoc2.zip`
- `Development/project1.xls`
- `Development/project2.xls`
- `Finance/Tax2011/document1.pdf`
- `Finance/Tax2011/document2.pdf`
- `s3-dg.pdf`

Diese Objektschlüssel erzeugen eine logische Hierarchie mit `Private`, `Development` und `Finance` als Stammebenen-Ordner und `s3-dg.pdf` als Stammebenen-Objekt. Wenn Sie den Bucket-Namen in der Amazon S3-Konsole auswählen, erscheinen die Elemente in der Stammebene, wie in der folgenden Abbildung dargestellt. Die Konsole zeigt die Präfixe der obersten Ebene (`Private/`, `Development/` und `Finance/`) als Stammebenen-Ordner. Der Objektschlüssel `s3-dg.pdf` hat kein Präfix und erscheint daher als Stammebenen-Element.



Walkthrough-Übersicht

In dieser Anleitung erstellen Sie einen Bucket mit drei Ordnern (`Private`, `Development` und `Finance`).

Sie haben zwei User, Alice und Bob. Sie möchten, dass Alice nur auf den Ordner `Development` und Bob nur auf den Ordner `Finance` zugreift. Sie möchten, das der Inhalt des Ordners `Private` privat bleibt. In der Anleitung verwalten Sie den Zugriff durch Erstellen von IAM-Benutzern (im Beispiel werden die Benutzernamen Alice und Bob verwendet) und gewähren ihnen die erforderlichen Berechtigungen.

IAM unterstützt auch die Schaffung von Benutzergruppen und von Berechtigungen auf Gruppenebene, die für alle User in der Gruppe gelten. Dies hilft Ihnen bei der Verwaltung der Berechtigungen. In dieser Übung benötigen sowohl Alice als auch Bob einige gemeinsame Berechtigungen. Sie erstellen also eine Gruppe mit dem Namen `Consultants` und fügen dann Alice und Bob der Gruppe hinzu. Sie erteilen zunächst Berechtigungen, indem Sie der Gruppe eine Gruppenrichtlinie zuweisen. Fügen Sie dann benutzerspezifische Berechtigungen durch Zuweisen von Richtlinien zu bestimmten Benutzern hinzu.

> **Note**
>
> Die Anleitung verwendet `companybucket` als Bucket-Namen, Alice und Bob als IAM-Benutzer und `Consultants` als Gruppennamen. Da Bucket-Namen in Amazon S3 global eindeutig sein müsssen, müssen Sie den Bucket-Namen durch einen von Ihnen erstellten Namen ersetzen.

Vorbereitung auf den Walkthrough

In diesem Beispiel verwenden Sie Ihre AWS-Anmeldeinformationen, um IAM-Benutzer zu erstellen. Zu Beginn haben diese Benutzer keine Berechtigungen. Sie gewähren diesen Benutzern nach und nach Berechtigungen, damit sie spezifische Amazon S3-Aktionen ausführen können. Um diese Berechtigungen zu testen, melden Sie sich mit den Anmeldeinformationen eines jeden Benutzers bei der Konsole an. Wenn Sie als AWS-Kontoinhaber schrittweise die Berechtigungen ausbauen und als IAM-Benutzer testen, müssen Sie sich mit den unterschiedlichen Anmeldeinformationen an- und abmelden. Sie können diesen Test auch mit einem Browser durchführen, das Verfahren schreitet aber schneller fort, wenn Sie zwei verschiedene Browser verwenden. Verwenden Sie einen Browser, um eine Verbindung mit der AWS Management Console mit den Anmeldeinformationen Ihres AWS-Kontos herzustellen, und einen anderen, um eine Verbindung mit den IAM-Benutzer-Anmeldeinformationen herzustellen.

Um sich bei der AWS Management Console mit Ihren AWS-Anmeldeinformationen anzumelden, gehen Sie zu https://console.aws.amazon.com/. Ein IAM-Benutzer kann sich nicht über denselben Link anmelden.

Ein IAM-Benutzer muss eine IAM-aktivierte Anmeldeseite verwendet werden. Als Kontoinhaber können Sie diesen Link Ihren Benutzern bereitstellen.

Weitere Informationen zu IAM finden Sie unter Die Benutzeranmeldeseite der AWS Management Console im IAM-Benutzerhandbuch.

So stellen Sie einen Anmelde-Link für IAM-Benutzer bereit

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich IAM Dashboard aus.
3. Notieren Sie sich die URL unter IAM users sign in link: (Anmelde-Link für IAM-Benutzer:). Sie geben diesen Link an die IAM-Benutzer weiter, damit sie sich mit ihrem IAM-Benutzernamen und -Passwort in der Konsole anmelden können.

Schritt 1: Erstellen eines Buckets

In diesem Schritt melden Sie sich bei der Amazon S3-Konsole mit ihren AWS-Konto-Anmeldeinformationen an, erstellen einen Bucket, fügen dem Bucket Ordner (`Development`, `Finance` und `Private`) hinzu und laden ein oder zwei Beispieldokumente in jeden Ordner hoch.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Erstellen Sie einen Bucket.

   Schrittweise Anleitungen finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
3. Laden Sie ein Dokument in den Bucket hoch.

   Diese Übung geht davon aus, dass sich das Dokument `s3-dg.pdf` auf der Stammebene dieses Buckets befindet. Wenn Sie andere Dokumente hochladen, ändern Sie ihren Dateinamen in `s3-dg.pdf`.
4. Fügen Sie drei Ordner mit dem Namen `Private`, `Finance` and `Development` zum Bucket hinzu.

   Schrittweise Anleitungen zum Erstellen eines Ordners finden Sie unter Erstellen eines Ordners im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
5. Laden Sie ein oder zwei Dokumente in jeden Ordner hoch.

   Für diese Übung wird angenommen, dass Sie einige Dokumente in jeden Ordner hochgeladen haben, sodass der Bucket Objekte mit den folgenden Schlüsseln enthält:

   - `Private/privDoc1.txt`
   - `Private/privDoc2.zip`
   - `Development/project1.xls`
   - `Development/project2.xls`
   - `Finance/Tax2011/document1.pdf`
   - `Finance/Tax2011/document2.pdf`
   - `s3-dg.pdf`

   Schrittweise Anleitungen finden Sie unter Wie lade ich Dateien und Ordner in einen S3-Bucket hoch? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Schritt 2: Erstellen von IAM-Benutzern und einer Gruppe

Sie verwenden jetzt die IAM-Konsole, um die beiden IAM-Benutzer Alice und Bob zu Ihrem AWS-Konto hinzuzufügen. Sie erstellen auch eine Administratorgruppe mit dem Namen `Consultants` und fügen dann beide Benutzer der Gruppe hinzu.

> Warning
>
> Wenn Sie die Benutzer und die Gruppe erstellen, fügen Sie keine Richtlinien an, die diesen Benutzern Berechtigungen erteilen. Anfänglich haben diese Benutzer keine Berechtigungen. In den folgenden Abschnitten gewähren Sie nach und nach Berechtigungen. Sie müssen zunächst sicherstellen, dass Sie diesen IAM-Benutzern Passwörter zugewiesen haben. Sie verwenden diese Benutzer-Anmeldeinformationen zum Testen der Amazon S3-Aktionen und zum Überprüfen, ob die Berechtigungen wie erwartet funktionieren.

Schrittweise Anleitungen zum Erstellen eines neuen IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS-Konto im IAM-Benutzerhandbuch. Wenn Sie die Benutzer für diese Anleitung erstellen, wählen Sie AWS Management Console access (AWS-Managementkonsolenzugriff) aus und deaktivieren Sie Programmatic access (Programmgesteuerter Zugriff).

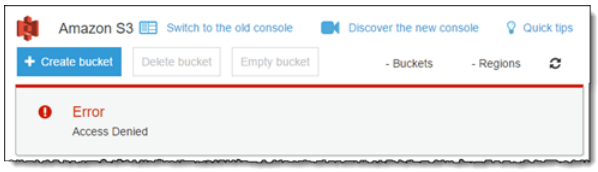
Eine schrittweise Anleitung zum Erstellen einer Administratorgruppe finden Sie unter Erstellen Ihres ersten Administratorbenutzers und Ihrer ersten Administratorgruppe in IAM im IAM-Benutzerhandbuch.

Schritt 3: Überprüfen, dass die IAM-Benutzer über keine Berechtigungen verfügen

Wenn Sie zwei Browser verwenden, können Sie jetzt den zweiten Browser verwenden, um sich mit einer der IAM-Benutzer-Anmeldeinformationen bei der Konsole anzumelden.

1. Melden Sie sich über den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmelde-Link für IAM-Benutzer bereit (p. 465)) bei der AWS Management Console an und verwenden Sie dazu eine der IAM-Benutzeranmeldeinformationen.
2. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.

   Die folgende Konsolenmeldung informiert Sie darüber, dass der Zugang verweigert wurde.



Nun können Sie damit beginnen, den Benutzern schrittweise Berechtigungen zu erteilen. Sie weisen zunächst eine Gruppenrichtlinie zu, die beiden Benutzern die erforderlichen Berechtigungen gewährt.

Schritt 4: Erteilen von Berechtigungen auf Gruppenebene

Sie möchten den Benutzern Folgendes ermöglichen:

- Auflisten aller Buckets, die dem übergeordneten Konto gehören. Um das zu tun, müssen Bob und Alice die Berechtigung für die Aktion `s3:ListAllMyBuckets` besitzen.
- Auflisten aller Elemente, Ordner und Objekte auf Stammebene im Bucket `companybucket`. Um das zu tun, müssen Bob und Alice die Berechtigung für die Aktion `s3:ListBucket` im Bucket `companybucket` besitzen.

Zuerst erstellen Sie eine Richtlinie, die diese Berechtigungen gewährt, und dann weisen Sie sie der Gruppe `Consultants` zu.

Schritt 4.1: Erteilen der Berechtigung zum Auflisten aller Buckets

In diesem Schritt erstellen Sie eine verwaltete Richtlinie, die den Benutzern die Mindestberechtigungen für die Auflistung aller Buckets des übergeordneten Kontos erteilt. Dann weisen Sie die Richtlinie der Gruppe `Consultants` zu. Wenn Sie einem Benutzer oder einer Benutzergruppe die verwaltete Richtlinie zuordnen, erhält der Benutzer oder die Gruppe die Berechtigung, alle Buckets des übergeordneten AWS-Kontos aufzulisten.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   Note

   Da Sie Benutzerberechtigungen erteilen, müssen Sie sich mit Ihren Anmeldeinformationen für das AWS-Konto anmelden und nicht als IAM-Benutzer.

2. Erstellen Sie die verwaltete Richtlinie.

   a. Wählen Sie links im Navigationsbereich Policies (Richtlinien) und dann Create Policy (Richtlinie erstellen) aus.

   b. Wählen Sie die Registerkarte JSON.

   c. Kopieren Sie die folgende Zugriffsrichtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGroupToSeeBucketListInTheConsole",
      "Action": ["s3:ListAllMyBuckets"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::*"]
    }
  ]
}
```

   Eine Richtlinie ist ein JSON-Dokument. Im Dokument ist ein `Statement` ein Array von Objekten, die jeweils eine Berechtigung unter Verwendung einer Sammlung von Namenswertpaaren beschreiben. Die vorangegangene Richtlinie beschreibt eine bestimmte Berechtigung. Die `Action` definiert den Zugriffstyp. In der Richtlinie ist `s3:ListAllMyBuckets` eine vordefinierte Amazon S3-Aktion. Diese Aktion umfasst die Operation Amazon S3 GET Service, die eine Liste aller Buckets des authentifizierten Absenders zurückgibt. Der Wert des `Effect`-Elements bestimmt, ob die spezifische Berechtigung gewährt oder verweigert wird.

   d. Wählen Sie Review policy (Richtlinie überprüfen) aus. Geben Sie auf der nächsten Seite in das Feld Name `AllowGroupToSeeBucketListInTheConsole` ein und wählen Sie dann Create policy (Richtlinie erstellen).

      Note

      Der Eintrag Summary (Übersicht) enthält eine Nachricht, die angibt, dass die Richtlinie keinerlei Berechtigungen gewährt. Für diese Anleitung können Sie diese Nachricht getrost ignorieren.

3. Weisen Sie die von `AllowGroupToSeeBucketListInTheConsole` verwaltete Richtlinie, die Sie erstellt haben, der Gruppe `Consultants` zu.

   Schrittweise Informationen zum Anfügen einer verwalteten Richtlinie finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien im IAM-Benutzerhandbuch.

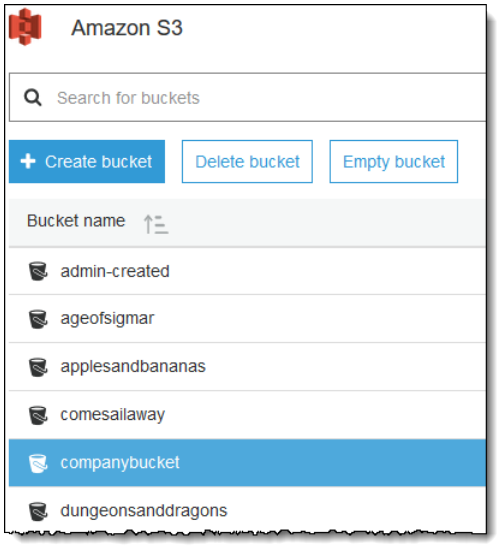
Sie fügen die Richtliniendokumente in der IAM-Konsole den IAM-Benutzern und Gruppen hinzu. Da Sie möchten, dass beide Benutzer die Buckets auflisten können, weisen Sie die Richtlinie der Gruppe zu.

4. Die Berechtigung testen.

   a. Melden Sie sich über den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmelde-Link für IAM-Benutzer bereit (p. 465)) bei der Konsole an und verwenden Sie eine der IAM-Benutzeranmeldeinformationen.

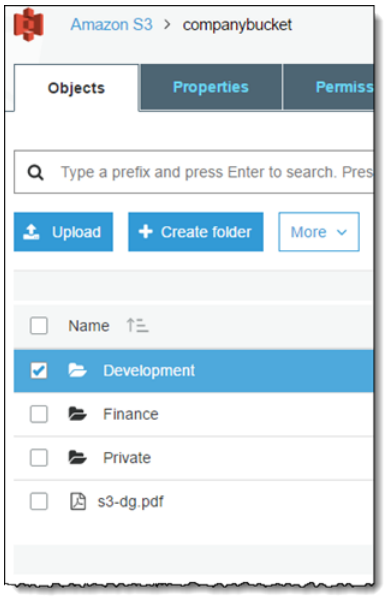
   b. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.

      Die Konsole sollte nun alle Buckets auflisten, nicht aber die Objekte in den jeweiligen Buckets.



Schritt 4.2: Benutzern gestatten, dass sie den Bucket-Inhalt auf Stammebene auflisten

Als Nächstes gestatten Sie allen Benutzern in der Gruppe `Consultants`, Elemente im Bucket `companybucket` aufzulisten. Wenn ein Benutzer den Unternehmensbucket in der Amazon S3-Konsole auswählt, werden die Elemente in der Stammebene des Buckets angezeigt.



Note

Dieses Beispiel verwendet `companybucket` zur Veranschaulichung. Sie müssen den Namen des Bucket verwenden, den Sie erstellt haben.

Um zu verstehen, welche Anforderung die Konsole an Amazon S3 sendet, wenn Sie einen Bucket-Namen auswählen, welche Antwort Amazon S3 zurückgibt und wie die Konsole die Antwort interpretiert, ist es notwendig, etwas tiefer in die Materie einzudringen.

Wenn Sie auf einen Bucket-Namen klicken, sendet die Konsole die Anforderung GET Bucket (List Objects) an Amazon S3. Diese Anforderung enthält die folgenden Parameter:

- Der Parameter `prefix` mit einer leeren Zeichenfolge als Wert.
- Der Parameter `delimiter` mit `/` als Wert.

Es folgt ein Beispiel einer Anforderung.

```
GET ?prefix=&delimiter=/ HTTP/1.1
Host: companybucket.s3.amazonaws.com
Date: Wed, 01 Aug  2012 12:00:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:xQE0diMbLRepdf3YB+FIEXAMPLE=
```

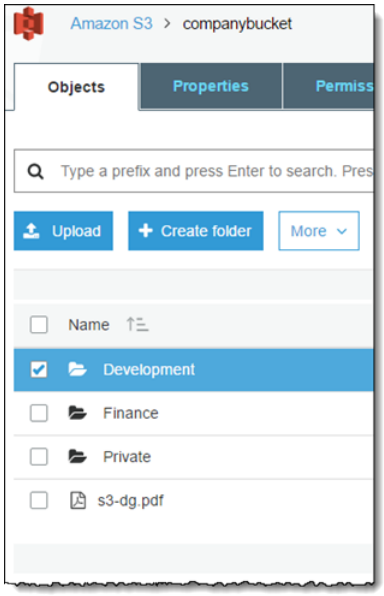
Amazon S3 gibt eine Antwort zurück, die das folgende `<ListBucketResult/>`-Element enthält:

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>companybucket</Name>
  <Prefix></Prefix>
  <Delimiter>/</Delimiter>
   ...
  <Contents>
    <Key>s3-dg.pdf</Key>
    ...
  </Contents>
  <CommonPrefixes>
    <Prefix>Development/</Prefix>
  </CommonPrefixes>
```

```
  <CommonPrefixes>
    <Prefix>Finance/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>Private/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Der Schlüsselobjekt `s3-dg.pdf` enthält nicht das Schrägstrich-Trennzeichen (`/`), und Amazon S3 gibt den Schlüssel im `<Contents>`-Element zurück. Alle anderen Schlüssel im Beispiel-Bucket enthalten jedoch das Trennzeichen `/`. Amazon S3 gruppiert diese Schlüssel und gibt ein `<CommonPrefixes>`-Element für jeden der verschiedenen Präfixwerte `Development/`, `Finance/` und `Private/` zurück, also eine Unterzeichenfolge vom Anfang dieser Schlüssel bis zum ersten Auftreten des angegebenen Trennzeichens `/`.

Die Konsole interpretiert dieses Ergebnis und zeigt die Elemente auf der Stammebene als drei Ordner und einen Objektschlüssel an.



Wenn nun Bob oder Alice den Ordner Development öffnen, sendet die Konsole die Anforderung GET Bucket (List Objects) an Amazon S3, wobei die Parameter `prefix` und `delimiter` auf die folgenden Werte eingestellt sind:

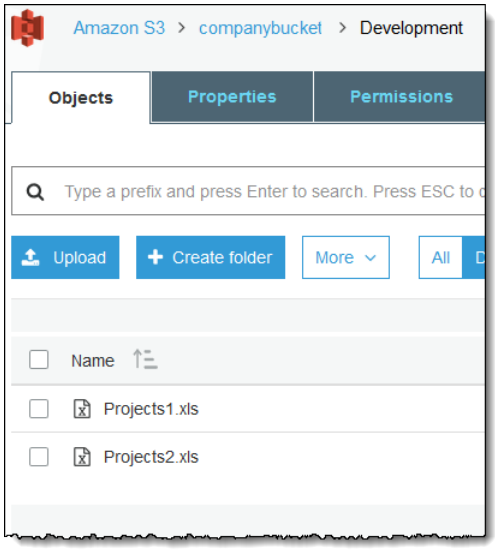
- Der Parameter `prefix` mit dem Wert `Development/`.
- Der Parameter `delimiter` mit dem Wert "`/`".

In der Antwort gibt Amazon S3 die Objektschlüssel mit dem angegebenen Präfix zurück.

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>companybucket</Name>
  <Prefix>Development</Prefix>
  <Delimiter>/</Delimiter>
   ...
  <Contents>
    <Key>Project1.xls</Key>
    ...
  </Contents>
  <Contents>
```

```
    <Key>Project2.xls</Key>
    ...
  </Contents>
</ListBucketResult>
```

Die Konsole zeigt die Objektschlüssel an.



Kehren Sie nun wieder zu dem Vorgang zurück, Benutzern die Berechtigung zum Auflisten von Elementen auf der Stammebene zu erteilen. Damit der Bucket-Inhalt aufgelistet werden kann, müssen die Benutzer die Berechtigung zum Aufruf der `s3:ListBucket`-Aktion besitzen, wie in der folgenden Richtlinienanweisung dargestellt. Um sicherzustellen, dass sie nur den Inhalt auf der Stammebene sehen, können Sie als Bedingung hinzufügen, dass Benutzer in der Anforderung ein leeres `prefix` angeben müssen – d. h., dass es ihnen nicht gestattet sein soll, auf einen der Stammebenen-Ordner doppelzuklicken. Schließlich fügen Sie noch eine Bedingung hinzu, die den Zugriff im Ordnerstil dadurch vorschreibt, dass Benutzeranforderungen den Parameter `delimiter` mit dem Wert "`/`" enthalten müssen.

```
{
  "Sid": "AllowRootLevelListingOfCompanyBucket",
  "Action": ["s3:ListBucket"],
  "Effect": "Allow",
  "Resource": ["arn:aws:s3:::companybucket"],
  "Condition":{
         "StringEquals":{
             "s3:prefix":[""], "s3:delimiter":["/"]
                       }
          }
}
```

Wenn Sie einen Bucket in der Amazon S3-Konsole auswählen, sendet die Konsole zuerst die Anforderung GET Bucket location, um die AWS-Region zu finden, in der der Bucket bereitgestellt wird. Dann verwendet die Konsole den regionsspezifischen Endpunkt für den Bucket, um die Anforderung GET Bucket (List Objects) zu senden. Wenn Benutzer die Konsole verwenden sollen, müssen Sie ihnen folglich die Berechtigung für die Aktion `s3:GetBucketLocation` gewähren, wie in der folgenden Richtlinienanweisung veranschaulicht.

```
{
   "Sid": "RequiredByS3Console",
   "Action": ["s3:GetBucketLocation"],
   "Effect": "Allow",
```

```
   "Resource": ["arn:aws:s3:::*"]
}
```

Den Benutzern gestatten, den Bucket-Inhalt auf Stammebene aufzulisten

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   Verwenden Sie für die Anmeldung in der Konsole ihre Anmeldeinformationen für das AWS-Konto, nicht jene eines IAM-Benutzers.

2. Ersetzen Sie die vorhandene `AllowGroupToSeeBucketListInTheConsole`-verwaltete Richtlinie, die der Gruppe `s3:ListBucket` zugeordnet ist, durch die folgende Richtlinie, durch die die Aktion `Consultants` ebenfalls gestattet wird. Denken Sie daran, den Namen *`companybucket`* in der Richtlinie `Resource` durch den Namen Ihres Buckets zu ersetzen.

   Schritt-für Schritt-Anleitungen finden Sie unter Bearbeiten von IAM-Richtlinien im IAM-Benutzerhandbuch. Wenn Sie den schrittweisen Anweisungen folgen, beachten Sie bitte die Anweisungen zum Anwenden Ihrer Änderungen auf alle Prinzipal-Entitäten, denen die Richtlinie zugeordnet ist.

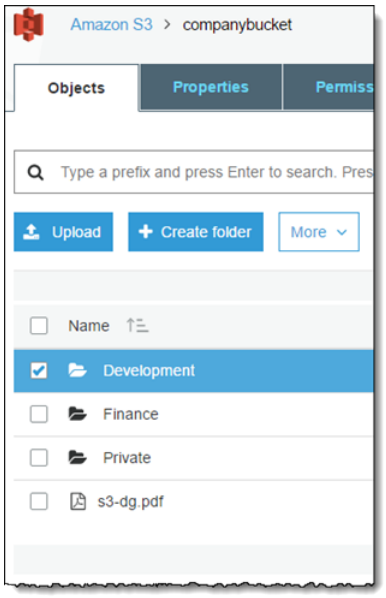
```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Sid":
 "AllowGroupToSeeBucketListAndAlsoAllowGetBucketLocationRequiredForListBucket",
       "Action": [ "s3:ListAllMyBuckets", "s3:GetBucketLocation" ],
       "Effect": "Allow",
       "Resource": [ "arn:aws:s3:::*"  ]
     },
     {
       "Sid": "AllowRootLevelListingOfCompanyBucket",
       "Action": ["s3:ListBucket"],
       "Effect": "Allow",
       "Resource": ["arn:aws:s3:::companybucket"],
       "Condition":{
             "StringEquals":{
                    "s3:prefix":[""], "s3:delimiter":["/"]
                           }
                  }
     }
  ]
}
```

3. Testen Sie die aktualisierten Berechtigungen.

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmelde-Link für IAM-Benutzer bereit (p. 465)) für die Anmeldung in der AWS Management Console.

      Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.

   b. Wählen Sie den von Ihnen erstellten Bucket aus. Die Konsole zeigt die Bucket-Elemente auf Stammebene an. Wenn Sie Ordner im Bucket auswählen, können Sie den Ordnerinhalt nicht sehen, da Sie diese Berechtigungen noch nicht gewährt haben.



Dieser Test ist erfolgreich, wenn Benutzer die Amazon S3-Konsole verwenden. Wenn Sie einen Bucket in der Konsole auswählen, sendet die Konsolenimplementierung eine Anforderung mit dem Parameter `prefix` mit einer leeren Zeichenfolge als Wert und dem Parameter `delimiter` mit "`/`" als Wert.

Schritt 4.3: Übersicht über die Gruppenrichtlinie

Die Wirkung der von Ihnen hinzugefügten Gruppenrichtlinie ist, dass die IAM-Benutzer Alice und Bob über die folgenden Mindestberechtigungen verfügen:

- Auflisten aller Buckets, die dem übergeordneten Konto gehören.
- Ansicht der Elemente auf Stammebene im Bucket `companybucket`.

Die Benutzer können jedoch noch nicht sehr viele Aktionen ausführen. Als Nächstes erteilen Sie die folgenden benutzerspezifischen Berechtigungen:

- Erlauben Sie Bob, Objekte im Ordner `Development` aufzurufen und abzulegen.
- Erlauben Sie Bob, Objekte im Ordner `Finance` aufzurufen und abzulegen.

Für benutzerspezifische Berechtigungen fügen Sie eine Richtlinie zum spezifischen Benutzer hinzu, nicht für die Gruppe. Im folgenden Abschnitt erteilen Sie Alice die Berechtigung, mit dem Ordner `Development` zu arbeiten. Sie können die Schritte wiederholen, um Bob eine ähnliche Berechtigung für das Arbeiten im Ordner `Finance` zu erteilen.

Schritt 5: Erteilen spezifischer Berechtigungen für IAM-Benutzer Alice

Nun gewähren Sie Alice zusätzliche Berechtigungen, damit sie den Inhalt des Ordners `Development` aufrufen und Objekte in diesem Ordner ablegen kann.

Schritt 5.1: Erteilen der Berechtigung zum Auflisten des Inhalts des Development-Ordners für IAM-Benutzer Alice

Damit Alice den Inhalt des Ordners `Development` auflisten kann, müssen Sie der Benutzerin Alice eine Richtlinie zuweisen, die die Berechtigung für die Aktion `s3:ListBucket` für den Bucket `companybucket`

erteilt, unter der Voraussetzung, dass die Anforderung das Präfix `Development/` enthält. Da diese Richtlinie nur auf die Benutzerin Alice angewendet werden soll, verwenden Sie eine Inline-Richtlinie. Weitere Informationen zu eingebundenen Richtlinien finden Sie unter Verwaltete Richtlinien und eingebundene Richtlinien im IAM-Benutzerhandbuch.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   Verwenden Sie für die Anmeldung in der Konsole ihre Anmeldeinformationen für das AWS-Konto, nicht jene eines IAM-Benutzers.

2. Erstellen Sie eine Inline-Richtlinie, um der Benutzerin Alice die Berechtigung zu erteilen, den Inhalt des Ordners `Development` aufzulisten.

   a. Wählen Sie im Navigationsbereich auf der linken Seite Users (Benutzer) aus.

   b. Klicken Sie auf den Benutzernamen Alice.

   c. Wählen Sie auf der Benutzerdetailseite die Registerkarte Permissions (Berechtigungen) und dann Add inline policy (Inline-Richtlinie hinzufügen) aus.

   d. Wählen Sie die Registerkarte JSON.

   e. Kopieren Sie die folgende Richtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
      "Sid": "AllowListBucketIfSpecificPrefixIsIncludedInRequest",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::companybucket"],
      "Condition":{  "StringLike":{"s3:prefix":["Development/*"] }
       }
    }
  ]
}
```

   f. Wählen Sie Review policy (Richtlinie überprüfen) aus. Geben Sie auf der nächsten Seite in das Feld Name einen Namen ein und wählen Sie dann Create policy (Richtlinie erstellen).

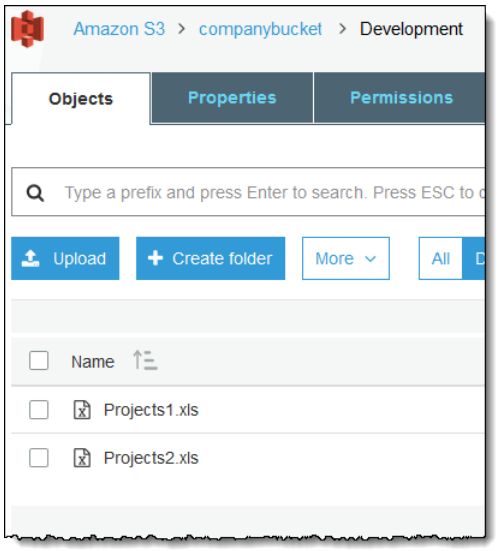
3. Testen Sie die geänderten Berechtigungen für Alice:

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmelde-Link für IAM-Benutzer bereit (p. 465)) für die Anmeldung bei der AWS Management Console.

   b. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.

   c. Prüfen Sie in der Amazon S3-Konsole, ob Alice die Liste der Objekte im Ordner `Development/` des Buckets sehen kann.

      Wenn die Benutzerin den Ordner `/Development` auswählt, um die Liste der darin enthaltenen Objekte anzuzeigen, sendet die Amazon S3-Konsole die Anforderung `ListObjects` zusammen mit dem Präfix `/Development` an Amazon S3. Da die Benutzerin dazu berechtigt ist, die Objektliste mit dem Präfix `Development` und dem Trennzeichen `/` anzuzeigen, gibt Amazon S3 die Objektliste mit dem Schlüsselpräfix `Development/` zurück, und die Konsole zeigt die Liste an.



Schritt 5.2: Erteilen von Berechtigungen, auf die Objekte im Development-Ordner zuzugreifen und Objekte darin abzulegen, für IAM-Benutzer Alice

Damit Alice Objekte im Ordner `Development` ablegen und aufrufen kann, benötigt sie die Berechtigung für die Aktionen `s3:GetObject` und `s3:PutObject`. Die folgenden Richtlinienanweisungen räumen diese Berechtigungen ein, vorausgesetzt die Anforderung enthält den Parameter `prefix` mit dem Wert `Development/`.

```
{
    "Sid":"AllowUserToReadWriteObjectData",
    "Action":["s3:GetObject", "s3:PutObject"],
    "Effect":"Allow",
    "Resource":["arn:aws:s3:::companybucket/Development/*"]
 }
```

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   Verwenden Sie für die Anmeldung in der Konsole ihre Anmeldeinformationen für das AWS-Konto, nicht jene eines IAM-Benutzers.
2. Bearbeiten Sie die Inline-Richtlinie, die Sie im vorherigen Schritt erstellt haben.

   a. Wählen Sie im Navigationsbereich auf der linken Seite Users (Benutzer) aus.
   b. Klicken Sie auf den Benutzernamen "Alice".
   c. Wählen Sie auf der Benutzerdetailseite die Registerkarte Permissions (Berechtigungen) aus und erweitern Sie den Bereich Inline Policies (Inline-Richtlinien).
   d. Wählen Sie neben dem Namen der im vorherigen Schritt erstellten Richtlinie Edit Policy (Richtlinie bearbeiten) aus.
   e. Kopieren Sie die folgende Richtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein, wobei die vorhandene Richtlinie ersetzt wird.

```
{
     "Version": "2012-10-17",
     "Statement":[
      {
         "Sid":"AllowListBucketIfSpecificPrefixIsIncludedInRequest",
```

```
            "Action":["s3:ListBucket"],
            "Effect":"Allow",
            "Resource":["arn:aws:s3:::companybucket"],
            "Condition":{
               "StringLike":{"s3:prefix":["Development/*"]
               }
            }
         },
         {
           "Sid":"AllowUserToReadWriteObjectDataInDevelopmentFolder",
           "Action":["s3:GetObject", "s3:PutObject"],
           "Effect":"Allow",
           "Resource":["arn:aws:s3:::companybucket/Development/*"]
         }
      ]
}
```

3. Testen Sie die aktualisierte Richtlinie:

   a. Verwenden Sie den Anmeldelink für IAM-Benutzer (siehe So stellen Sie einen Anmelde-Link für IAM-Benutzer bereit (p. 465)) für die Anmeldung in der AWS Management Console.

   b. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.

   c. Prüfen Sie in der Amazon S3-Konsole, ob Alice nun im Ordner `Development` ein Objekt hinzufügen oder herunterladen kann.

Schritt 5.3: Ausdrückliches Verweigern von Berechtigungen für den Zugriff auf alle anderen Ordner im Bucket für IAM-Benutzer Alice

Die Benutzerin kann jetzt den Inhalt auf Stammebene im Bucket `companybucket` auflisten. Sie kann auch Objekte im Ordner `Development` aufrufen und ablegen. Wenn Sie die Zugriffsberechtigungen weiter verbessern möchten, können Sie Alice den Zugriff auf andere Ordner im Bucket explizit verweigern. Wenn es irgendeine andere Richtlinie (Bucket-Richtlinie oder ACL) gibt, die Alice den Zugriff auf andere Ordner im Bucket gewährt, überschreibt diese explizite Zugriffsverweigerung diese Berechtigungen.

Sie können die folgende Anweisung zur Benutzerrichtlinie für Alice hinzufügen, die von allen Anforderungen von Alice an Amazon S3 erfordert, dass der Parameter `prefix` enthalten ist, dessen Wert entweder `Development/*` oder eine leere Zeichenfolge ist.

```
{
   "Sid": "ExplicitlyDenyAnyRequestsForAllOtherFoldersExceptDevelopment",
   "Action": ["s3:ListBucket"],
   "Effect": "Deny",
   "Resource": ["arn:aws:s3:::companybucket"],
   "Condition":{  "StringNotLike": {"s3:prefix":["Development/*",""] },
                  "Null"         : {"s3:prefix":false }
    }
}
```

Beachten Sie, dass im Block `Condition` zwei bedingte Ausdrücke enthalten sind. Das Ergebnis dieser bedingten Ausdrücke wird durch das logische `AND` verknüpft. Wenn beide Bedingungen wahr sind, ist das Ergebnis der bedingten Ausdrücke wahr. Da der Wert für `Effect` in dieser Richtlinie `Deny` lautet, wenn `Condition` als "true" bewertet wird, können Benutzer die angegebene `Action` nicht durchführen.

- Der bedingte Ausdruck `Null` stellt sicher, dass die Anforderung von Alice den Parameter `prefix` enthält.

  Der Parameter `prefix` erfordert einen ordnerartigen Zugriff. Wenn Sie eine Anforderung ohne den Parameter `prefix` senden, gibt Amazon S3 alle Objektschlüssel zurück.

Wenn die Anforderung den Parameter `prefix` mit einem Nullwert enthält, wird der Ausdruck als wahr ausgewertet, womit die gesamte Bedingung `Condition` wahr ist. Sie müssen eine leere Zeichenfolge für den Parameter `prefix` gestatten. Erinnern Sie sich an die vorangegangene Diskussion. Die leere Zeichenfolge lässt zu, dass Alice Bucket-Elemente auf Stammebene abrufen kann, wie es die Konsole in der vorhergehenden Diskussion macht. Weitere Informationen finden Sie unter Schritt 4.2: Benutzern gestatten, dass sie den Bucket-Inhalt auf Stammebene auflisten (p. 468).

- Der bedingte Ausdruck `StringNotLike` stellt sicher, dass die Anforderung scheitert, wenn der angegebene Wert des Parameters `prefix` nicht `Development/*` ist.

Folgen Sie den Schritten im vorherigen Abschnitt und aktualisieren Sie die Inline-Richtlinie noch einmal, die Sie für die Benutzerin Alice erstellt haben.

Kopieren Sie die folgende Richtlinie und fügen Sie sie in das Textfeld für die Richtlinie ein, wobei die vorhandene Richtlinie ersetzt wird.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllowListBucketIfSpecificPrefixIsIncludedInRequest",
         "Action":["s3:ListBucket"],
         "Effect":"Allow",
         "Resource":["arn:aws:s3:::companybucket"],
         "Condition":{
            "StringLike":{"s3:prefix":["Development/*"]
            }
         }
      },
      {
        "Sid":"AllowUserToReadWriteObjectDataInDevelopmentFolder",
        "Action":["s3:GetObject", "s3:PutObject"],
        "Effect":"Allow",
        "Resource":["arn:aws:s3:::companybucket/Development/*"]
      },
      {
         "Sid": "ExplicitlyDenyAnyRequestsForAllOtherFoldersExceptDevelopment",
         "Action": ["s3:ListBucket"],
         "Effect": "Deny",
         "Resource": ["arn:aws:s3:::companybucket"],
         "Condition":{  "StringNotLike": {"s3:prefix":["Development/*",""] },
                        "Null"         : {"s3:prefix":false }
          }
      }
   ]
}
```

Schritt 6: Erteilen spezifischer Berechtigungen für IAM-Benutzer Bob

Sie können die Schritte wiederholen, um Bob eine ähnliche Berechtigung für den Ordner `Finance` zu erteilen. Führen Sie die Schritte aus, mit denen Sie zuvor Alice Berechtigungen erteilt haben, aber ersetzen Sie den Ordner `Development` durch den Ordner `Finance`. Schrittweise Anleitungen hierzu finden Sie unter Schritt 5: Erteilen spezifischer Berechtigungen für IAM-Benutzer Alice (p. 473).

Schritt 7: Absichern des Ordners „Private“

In diesem Beispiel haben Sie nur zwei Benutzer. Sie haben auf Gruppenebene alle erforderlichen Mindestberechtigungen erteilt und auf Benutzerebene die Berechtigungen nur dann gewährt, wenn die einzelnen Benutzer die Berechtigungen wirklich benötigen. Dieser Ansatz minimiert den Aufwand beim Verwalten der Berechtigungen. Wenn die Anzahl der Benutzer steigt, kann das Verwalten der

Berechtigungen mühsam werden. Sie möchten z. B. nicht, dass irgendwelche der Benutzer in diesem Beispiel auf den Inhalt des Ordners `Private` zugreifen können. Wie stellen Sie sicher, dass Sie nicht versehentlich einem Benutzer Berechtigung darauf erteilen? Sie fügen eine Richtlinie hinzu, die explizit den Zugriff auf den Ordner verweigert. Eine explizite Zugriffsverweigerung überschreibt alle anderen Berechtigungen.

Um sicherzustellen, dass der Ordner `Private` auch privat bleibt, können Sie die folgenden beiden Ablehnungsanweisungen zur Gruppenrichtlinie hinzufügen:

- Fügen Sie die folgende Anweisung hinzu, um jede Aktion auf die Ressourcen im `Private`-Ordner (`companybucket/Private/*`) zu verweigern.

```
{
  "Sid": "ExplictDenyAccessToPrivateFolderToEveryoneInTheGroup",
  "Action": ["s3:*"],
  "Effect": "Deny",
  "Resource":["arn:aws:s3:::companybucket/Private/*"]
}
```

- Sie verweigern auch die Berechtigung für die Aktion Objekte auflisten, wenn die Anforderung das Präfix `Private/` angibt. Wenn Bob oder Alice in der Konsole den Order `Private` öffnen, bewirkt diese Richtlinie, dass Amazon S3 eine Fehlermeldung zurückgibt.

```
{
  "Sid": "DenyListBucketOnPrivateFolder",
  "Action": ["s3:ListBucket"],
  "Effect": "Deny",
  "Resource": ["arn:aws:s3:::*"],
  "Condition":{
      "StringLike":{"s3:prefix":["Private/"]}
   }
}
```

Ersetzen Sie die Gruppenrichtlinie `Consultants` durch eine aktualisierte Richtlinie, die die vorherigen Ablehnungsanweisungen enthält. Nachdem die aktualisierte Richtlinie angewendet wurde, kann keiner der Benutzer in der Gruppe mehr auf den Order `Private` in Ihrem Bucket zugreifen.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

   Verwenden Sie für die Anmeldung in der Konsole ihre Anmeldeinformationen für das AWS-Konto, nicht jene eines IAM-Benutzers.
2. Ersetzen Sie die vorhandene von `AllowGroupToSeeBucketListInTheConsole` verwaltete Richtlinie, die der Gruppe `Consultants` zugeordnet ist, durch die folgende Richtlinie. Denken Sie daran, den Namen `companybucket` in der Richtlinie durch den Namen Ihres Buckets zu ersetzen.

   Weitere Anweisungen finden Sie unter Bearbeiten von kundenverwalteten Richtlinien im IAM-Benutzerhandbuch. Wenn Sie die Anweisungen nachvollziehen, beachten Sie bitte die Anweisungen zum Ändern aller Hauptentitäten, denen die Richtlinie zugeordnet ist.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":
 "AllowGroupToSeeBucketListAndAlsoAllowGetBucketLocationRequiredForListBucket",
      "Action": ["s3:ListAllMyBuckets", "s3:GetBucketLocation"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::*"]
```

```
    },
    {
      "Sid": "AllowRootLevelListingOfCompanyBucket",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::companybucket"],
      "Condition":{
          "StringEquals":{"s3:prefix":[""]}
       }
    },
    {
      "Sid": "RequireFolderStyleList",
      "Action": ["s3:ListBucket"],
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::*"],
      "Condition":{
          "StringNotEquals":{"s3:delimiter":"/"}
       }
     },
    {
      "Sid": "ExplictDenyAccessToPrivateFolderToEveryoneInTheGroup",
      "Action": ["s3:*"],
      "Effect": "Deny",
      "Resource":["arn:aws:s3:::companybucket/Private/*"]
    },
    {
      "Sid": "DenyListBucketOnPrivateFolder",
      "Action": ["s3:ListBucket"],
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::*"],
      "Condition":{
          "StringLike":{"s3:prefix":["Private/"]}
       }
    }
  ]
}
```

Schritt 8: Bereinigen

Öffnen Sie zur Bereinigung die IAM-Konsole und entfernen Sie die Benutzer Alice und Bob. Schritt-für-Schritt-Anleitungen finden Sie unter Löschen eines IAM-Benutzers im IAM-Benutzerhandbuch.

Um sicherzustellen, dass Sie für die Speicherung nicht künftig belastet werden, sollten Sie auch die Objekte und den Bucket löschen, die Sie für diese Übung erstellt haben.

Zugehörige Ressourcen

- Verwalten von IAM-Richtlinien im IAM-Benutzerhandbuch.

Zugriffsverwaltung mit ACLs

Themen

Zugriffskontrolllisten (ACLs) sind eine der auf Ressourcen basierenden Zugriffsrichtlinien-Optionen (siehe Übersicht über die Verwaltung von Zugriffsberechtigungen (p. 330)), mit denen Sie den Zugriff auf Ihre Buckets und Objekte verwalten können. sie können ACLs verwenden, um grundlegende Lese-/Schreibberechtigungen für andere AWS-Konten zu erteilen. Es gibt Limits für die Verwaltung von

Berechtigungen mit ACLs. Sie können beispielsweise nur für andere AWS-Konten Berechtigungen erteilen. Sie können keine Berechtigungen für Benutzer in Ihrem Konto erteilen. Sie können keine bedingten Berechtigung erteilen, und Sie können nicht explizit Berechtigungen verweigern. ACLs sind für spezifische Szenarien geeignet. Wenn ein Bucket-Eigentümer beispielsweise anderen AWS-Konten gestattet, Objekte hochzuladen, können diese Objekte nur unter Verwendung von Objekt-ACLs durch das AWS-Konto verwaltet werden, dem das Objekt gehört.

Die folgenden einführenden Themen erklären die Grundkonzepte und verfügbaren Optionen zum Verwalten des Zugriffs auf Ihre Amazon S3-Ressourcen, und in denen Richtlinien für die Verwendung der verschiedenen Zugriffsrichtlinienoptionen bereitgestellt sind.

- Einführung in das Verwalten des Zugriffs auf Amazon S3-Ressourcen (p. 330)
- Orientierungshilfen für die Verwendung der unterstützten Zugriffsrichtlinienoptionen (p. 344)

Zugriffskontrolllisten (ACL) – Übersicht

Themen

- Wer ist ein Empfänger? (p. 481)
- Welche Berechtigungen kann ich erteilen? (p. 483)
- Beispiel-ACL (p. 485)
- Vordefinierte ACL (p. 486)
- Eine ACL spezifizieren (p. 486)

Amazon S3-Zugriffskontrolllisten (ACLs) ermöglichen Ihnen die Verwaltung des Bucket- und Objektzugriffs. Jedem Bucket und jedem Objekt ist eine ACL als Subressource zugeordnet. Sie definiert, welche AWS-Konten oder Gruppen einen Zugriff erhalten, und um welchen Zugriffstyp es sich handelt. Wenn eine Anfrage für eine Ressource eingeht, überprüft Amazon S3 die entsprechende ACL,um sicherzustellen, dass der Auftraggeber die erforderlichen Zugriffsberechtigungen besitzt.

Wenn Sie einen Bucket oder ein Objekt erstellen, erstellt Amazon S3 eine Standard-ACL, die dem Ressourcen-Eigentümer die volle Kontrolle über die Ressource erteilt. Dies ist in der folgenden Beispiel-Bucket-ACL gezeigt (die Standard-Objekt-ACL hat denselben Aufbau):

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>*** Owner-Canonical-User-ID ***</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>*** Owner-Canonical-User-ID ***</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Die Beispiel-ACL enthält ein `Owner`-Element, das den Eigentümer über die kanonische Benutzer-ID des AWS-Kontos identifiziert. Anweisungen zum Auffinden Ihrer kanonischen Benutzer-ID finden Sie unter

Eine kanonischen Benutzer-ID für ein AWS-Konto ermitteln (p. 482). Das `Grant`-Element identifiziert den Empfänger (ein AWS-Konto oder eine vordefinierte Gruppe) und die erteilte Berechtigung. Diese Standard-ACL hat ein `Grant`-Element für den Eigentümer. Sie erteilen Berechtigungen, indem Sie `Grant`-Elemente hinzufügen, wobei jedes Recht den Empfänger und die Berechtigung identifiziert.

Note

Eine ACL kann bis zu 100 Rechte haben.

Wer ist ein Empfänger?

Ein Empfänger kann ein AWS-Konto sein, oder eine der vordefinierten Amazon S3-Gruppen. Sie erteilen einem AWS-Konto eine Berechtigung über die kanonische Benutzer-ID oder die E-Mail-Adresse Wenn Sie jedoch eine E-Mail-Adresse in Ihre Rechteerteilungsanfrage eintragen, findet Amazon S3 die kanonische Benutzer-ID für dieses Konto und fügt es der ACL hinzu. Die resultierenden ACLs enthalten immer die kanonische Benutzer-ID für das AWS-Konto, nicht die E-Mail-Adresse des AWS-Kontos.

Wenn Sie Zugriffsrechte erteilen, geben Sie jeden Empfänger als Typ=Wert-Paar an, wobei der Typ einer der folgenden ist:

- `id` – Wenn der angegebene Wert die kanonische Benutzer-ID eines AWS-Kontos ist
- `uri` – Wenn Sie einer vordefinierten Gruppe Berechtigungen erteilen
- `emailAddress` – Wenn der angegebene Wert die E-Mail-Adresse eines AWS-Kontos ist

Important

Die Verwendung von E-Mail-Adressen zur Angabe eines Berechtigungsempfängers wird ausschließlich in den folgenden AWS-Regionen unterstützt:

- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Europa (Irland)
- Südamerika (São Paulo)

Eine Liste aller unterstützten Amazon S3-Regionen und Endpunkte finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS-Referenz.

Example Beispiel: E-Mail-Adresse

Der folgende `x-amz-grant-read`-Header gewährt beispielsweise den durch E-Mail-Adressen angegebenen AWS-Konten Berechtigungen zum Lesen von Objektdaten und ihren Metadaten:

```
x-amz-grant-read: emailAddress="xyz@amazon.com", emailAddress="abc@amazon.com"
```

Warning

Wenn Sie anderen AWS-Konten Zugriff auf Ihre Ressourcen erteilen, müssen Sie beachten, dass die AWS-Konten ihre Berechtigungen an Benutzer unterhalb der Konten delegieren können. Man

spricht auch von einem kontenübergreifenden Zugriff. Informationen zum kontenübergreifenden Zugriff finden Sie unter Erstellen einer Rolle, um Berechtigungen an einen IAM-Benutzer zu delegieren im IAM-Benutzerhandbuch.

Eine kanonischen Benutzer-ID für ein AWS-Konto ermitteln

Die kanonische Benutzer-ID ist Ihrem AWS-Konto zugeordnet.
Es handelt sich dabei um eine lange Zeichenfolge, wie z. B. `79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be`. Informationen darüber, wo Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Suchen Ihrer kanonischen Benutzer-ID für das Konto.

Sie können die kanonische Benutzer-IDs eines AWS-Kontos auch ermitteln, indem Sie die ACL eines Buckets oder eines Objekts lesen, für den bzw. das das AWS-Konto Zugriffsberechtigungen besitzt. Wenn ein einzelnes AWS-Konto Berechtigungen durch eine Rechteerteilungsanfrage erhält, wird der ACL ein Rechteerteilungseintrag mit der kanonischen Benutzer-ID des AWS-Kontos hinzugefügt.

> Note
>
> Falls Sie Ihren Bucket öffentlich machen (nicht empfohlen), können beliebige, nicht authentifizierte Benutzer Objekte in den Bucket hochladen. Diese anonymen Benutzer haben kein AWS-Konto. Wenn ein anonymer Benutzer ein Objekt in Ihren Bucket hochlädt, fügt Amazon S3 eine spezielle kanonische Benutzer-ID (`65a011a29cdf8ec533ec3d1ccaae921c`) als Objekt-Eigentümer in der ACL hinzu. Weitere Informationen finden Sie unter Eigentümerschaft an Amazon S3 Buckets und Objekten (p. 331).

Vordefinierte Gruppen in Amazon S3

Amazon S3 besitzt mehrere vordefinierte Gruppen. Wenn Sie einem Konto Zugriff auf eine Gruppe erteilen, geben Sie eine unserer URIs statt einer kanonischen Benutzer-ID an. Wir stellen die folgenden vordefinierten Gruppen bereit:

- Gruppe „Authentifizierte Benutzer“ – Repräsentiert durch `http://acs.amazonaws.com/groups/global/AuthenticatedUsers`.

  Diese Gruppe stellt alle AWS-Konten dar. Die Zugriffsberechtigung für diese Gruppe gestattet jedem AWS-Konto, auf die Ressource zuzugreifen. Alle Anfragen müssen jedoch signiert (authentifiziert) sein.

  > Warning
  >
  > Wenn Sie einen Zugriff auf die Gruppe Authentifizierte Benutzer erteilen, hat jeder AWS-authentifizierte Benutzer weltweit Zugriff auf Ihre Ressource.

- Gruppe „Alle Benutzer“ – Repräsentiert durch `http://acs.amazonaws.com/groups/global/AllUsers`.

  Die Zugriffsberechtigung für diese Gruppe gestattet jedem, auf die Ressource zuzugreifen. Die Anfragen können signiert (authentifiziert) oder nicht signiert (anonym) sein. Nicht signierte Anfragen lassen den Authentifizierungs-Header in der Anfrage weg.

  > Warning
  >
  > Wir empfehlen dringend, dass Sie nie der Gruppe Alle Benutzer `WRITE`-, `WRITE_ACP`- oder `FULL_CONTROL`-Berechtigungen erteilen. Beispielsweise erlauben `WRITE`-Berechtigungen jedem, Objekte in Ihrem Bucket zu speichern, was dann Ihnen in Rechnung gestellt wird. Außerdem können andere Objekte löschen, die Sie vielleicht behalten wollen. Weitere Informationen zu diesen Berechtigungen finden Sie im folgenden Abschnitt Welche Berechtigungen kann ich erteilen? (p. 483).

- Gruppe „Protokollbereitstellung“ – Repräsentiert durch `http://acs.amazonaws.com/groups/s3/LogDelivery`.

Die WRITE-Berechtigung für einen Bucket gestattet dieser Gruppe, Serverzugriff-Protokolle (siehe Amazon S3-Serverzugriffsprotokollierung (p. 777)) in den Bucket zu schreiben.

> **Note**
>
> Bei Verwendung von ACLs kann ein Empfänger ein AWS-Konto sein, oder eine der vordefinierten Amazon S3-Gruppen. Der Empfänger kann jedoch kein IAM-Benutzer sein. Weitere Informationen zu AWS-Benutzern und Berechtigungen in IAM finden Sie unter Verwendung des AWS Identity and Access Managements.

Welche Berechtigungen kann ich erteilen?

Die folgende Tabelle listet die Berechtigungsmenge auf, die Amazon S3 in einer ACL unterstützt. Die Menge der ACL-Berechtigungen ist für eine Objekt-ACL und eine Bucket-ACL gleich. Abhängig vom Kontext (Bucket-ACL oder Objekt-ACL) erteilen jedoch diese ACL-Berechtigungen die Berechtigungen für spezifische Buckets oder Objektoperationen. Die Tabelle listet die Berechtigungen auf und beschreibt, was sie im Kontext der Objekte und Buckets bedeuten.

Berechtigung	Bei Rechteerteilung für einen Bucket	Bei Rechteerteilung für ein Objekt
`READ`	Gestattet dem Empfänger, die Objekte im Bucket aufzulisten	Gestattet dem Empfänger, die Objektedaten und seine Metadaten zu lesen
`WRITE`	Gestattet dem Empfänger, beliebige Objekte im Bucket zu erstellen, zu überschreiben und zu löschen	Nicht zutreffend
`READ_ACP`	Gestattet dem Empfänger, die Bucket-ACL zu lesen	Gestattet dem Empfänger, die Objekt-ACL zu lesen
`WRITE_ACP`	Gestattet dem Empfänger, die ACL für den relevanten Bucket zu schreiben	Gestattet dem Empfänger, die ACL für das relevante Objekt zu schreiben
`FULL_CONTROL`	Gestattet dem Empfänger, die READ-, WRITE-, READ_ACP- und WRITE_ACP-Berechtigungen für den Bucket	Gestattet dem Empfänger, die READ-, WRITE-, READ_ACP- und WRITE_ACPACL-Berechtigungen für das Objekt

> **Warning**
>
> Seien Sie beim Gewähren von Zugriffsberechtigungen auf Ihre S3-Buckets und -Objekte vorsichtig. Beispielsweise kann das Berechtigungsempfänger nach dem Gewähren des `WRITE`-Zugriffs auf einen Bucket beliebige Objekte im Bucket erstellen, überschreiben und löschen. Wir empfehlen dringend, dass Sie vor dem Erteilen von Berechtigungen diesen gesamten Zugriffskontrolllisten (ACL) – Übersicht (p. 480) Abschnitt lesen.

Zuordnung der ACL-Berechtigungen und Zugriffsrichtlinienberechtigungen

Wie in der obigen Tabelle gezeigt, erteilt eine ACL nur eine endliche Menge an Berechtigungen im Vergleich zu der Anzahl an Berechtigungen, die Sie in einer Zugriffsrichtlinie festlegen können (siehe Amazon S3-Aktionen (p. 380)). Jede dieser Berechtigungen erlaubt eine oder mehrere Amazon S3-Operationen.

Die folgende Tabelle zeigt, wie die verschiedenen ACL-Berechtigungen auf die entsprechenden Zugriffsrichtlinienberechtigungen abgebildet werden. Wie Sie sehen, erteilt die Zugriffsrichtlinie mehr Berechtigungen als eine ACL. Sie verwenden ACLs in erster Linie, um grundlegende Lese-/

Schreibberechtigungen zu erteilen, ähnlich den Berechtigungen in einem Dateisystem. Weitere Informationen dazu, wann Sie eine ACL verwenden sollten, finden Sie unter Orientierungshilfen für die Verwendung der unterstützten Zugriffsrichtlinienoptionen (p. 344).

ACL-Berechtigung	Entsprechende Zugriffsrichtlinienberechtigungen, wenn einem Bucket die ACL-Berechtigung erteilt wurde	Entsprechende Zugriffsrichtlinienberechtigungen, wenn einem Objekt die ACL-Berechtigung erteilt wurde
`READ`	`s3:ListBucket`, `s3:ListBucketVersions`, und `s3:ListBucketMultipartUploads`	`s3:GetObject`, `s3:GetObjectVersion`, und `s3:GetObjectTorrent`
`WRITE`	`s3:PutObject` und `s3:DeleteObject`. Wenn der Empfänger der Bucket-Eigentümer ist, gestattet die Erteilung der `WRITE`-Berechtigung in einer Bucket-ACL außerdem, dass die `s3:DeleteObjectVersion`-Aktion für jede Version in diesem Bucket ausgeführt wird.	Nicht zutreffend
`READ_ACP`	`s3:GetBucketAcl`	`s3:GetObjectAcl` und `s3:GetObjectVersionAcl`
`WRITE_ACP`	`s3:PutBucketAcl`	`s3:PutObjectAcl` und `s3:PutObjectVersionAcl`
`FULL_CONTROL`	Dies ist gleichbedeutend mit der Erteilung der `READ`-, `WRITE`-, `READ_ACP`- und `WRITE_ACP`-ACL-Berechtigungen. Dementsprechend wird diese ACL-Berechtigung auf eine Kombination entsprechender Zugriffsrichtlinienberechtigungen abgebildet.	Dies ist gleichbedeutend mit der Erteilung der `READ`-, `READ_ACP`- und `WRITE_ACP`-ACL-Berechtigungen. Dementsprechend wird diese ACL-Berechtigung auf eine Kombination entsprechender Zugriffsrichtlinienberechtigungen abgebildet.

Bedingungsschlüssel

Wenn Sie Zugriffsrichtlinienberechtigungen erteilen, können Sie Bedingungsschlüssel verwenden, um den Wert für die ACL für ein Objekt mithilfe einer Bucket-Richtlinie einzuschränken. Die folgenden Kontextschlüssel entsprechen ACLs. Sie können diese Kontextschlüssel verwenden, um die Verwendung einer bestimmten ACL in einer Anforderung durchzusetzen:

- `s3:x-amz-grant-read` - Erfordert Lesezugriff.
- `s3:x-amz-grant-write` - Erfordert Schreibzugriff.
- `s3:x-amz-grant-read-acp` - Erfordert Lesezugriff auf die Bucket-ACL.
- `s3:x-amz-grant-write-acp` - Erfordert Schreibzugriff auf die Bucket-ACL.
- `s3:x-amz-grant-full-control` - Erfordert vollständige Kontrolle.
- `s3:x-amz-acl` - Erfordert eine Vordefinierte ACL (p. 486).

Beispielrichtlinien, die ACL-spezifische Header enthalten, finden Sie unter Beispiel 1: s3:PutObject-Berechtigung mit der Bedingung erteilen, dass der Bucket-Eigentümer vollständige Kontrolle erhalten

muss (p. 385). Die vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Beispiel-ACL

Die folgende Beispiel-ACL für einen Bucket identifiziert den Ressourcen-Eigentümer und eine Menge von Rechten. Das Format ist die XML-Darstellung einer ACL im Amazon S3 REST API. Der Bucket-Eigentümer hat die `FULL_CONTROL` über die Ressource. Darüber hinaus zeigt die ACL, wie Berechtigungen für eine Ressource an zwei AWS-Konten erteilt werden, identifiziert durch eine kanonische Benutzer-ID, und an zwei der vordefinierten Amazon S3-Gruppen, wie im vorigen Abschnitt beschrieben.

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>Owner-canonical-user-ID</ID>
    <DisplayName>display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>Owner-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>user1-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>WRITE</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>user2-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
      </Grantee>
      <Permission>WRITE</Permission>
    </Grant>

  </AccessControlList>
</AccessControlPolicy>
```

Vordefinierte ACL

Amazon S3 unterstützt einen Satz vordefinierter Rechte, als vordefinierte ACLs bezeichnet. Jede vordefinierte ACL hat eine vordefinierte Menge aus Empfängern und Berechtigungen. Die folgende Tabelle listet die Menge der vordefinierten ACLs und der zugehörigen vordefinierten Rechte auf.

Vordefinierte ACL	Gilt für	Der ACL hinzugefügte Berechtigungen
`private`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL`. Niemand anderer hat Zugriffsrechte (Standard).
`public-read`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL`. Die `AllUsers`-Gruppe (siehe Wer ist ein Empfänger? (p. 481)) erhält `READ`-Zugriff.
`public-read-write`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL`. Die `AllUsers`-Gruppe erhält `READ`- und `WRITE`-Zugriff. Eine solche Erteilung von Rechten für einen Bucket wird im Allgemeinen nicht empfohlen.
`aws-exec-read`	Bucket und Objekt	Der Benutzer erhält `FULL_CONTROL`. Amazon EC2 erhält `READ`-Zugriff für den `GET`-Vorgang eines Amazon Machine Image (AMI)-Bundles von Amazon S3.
`authenticated-read`	Bucket und Objekt	Der Eigentümer erhält `FULL_CONTROL`. Die `AuthenticatedUsers`-Gruppe erhält `READ`-Zugriff.
`bucket-owner-read`	Objekt	Der Objekt-Eigentümer erhält `FULL_CONTROL`. Der Bucket-Eigentümer erhält `READ`-Zugriff. Wenn Sie diese vordefinierte ACL beim Erstellen eines Buckets angebe, ignoriert Amazon S3 sie.
`bucket-owner-full-control`	Objekt	Sowohl der Objekt-Eigentümer, als auch der Bucket-Eigentümer erhalten `FULL_CONTROL` für das Objekt. Wenn Sie diese vordefinierte ACL beim Erstellen eines Buckets angebe, ignoriert Amazon S3 sie.
`log-delivery-write`	Bucket	Die `LogDelivery`-Gruppe erhält `WRITE`- und `READ_ACP`-Berechtigungen für den Bucket. Weitere Informationen über Protokolle finden Sie unter (Amazon S3-Serverzugriffsprotokollierung (p. 777)).

> **Note**
>
> Sie können auch in Ihrer Anfrage nur eine dieser vordefinierten ACLs angeben.

Sie beben mit dem Anfrage-Header `x-amz-acl` eine vordefinierte ACL in Ihrer Anfrage an. Wenn Amazon S3 eine Anfrage mit einer vordefinierten ACL erhält, fügt es die vordefinierten Rechte der ACL der Ressource hinzu.

Eine ACL spezifizieren

Weitere Informationen zum Angeben einer ACL finden Sie in den folgenden Themen:

- Verwalten von ACLs in der AWS Management Console (p. 487)
- Verwalten von ACLs mit AWS SDK for Java (p. 487)
- Verwalten von ACLs mit AWS SDK für .NET (p. 489)
- Verwalten von ACLs mit der REST API (p. 492)

ACLs verwalten

Themen

Es gibt mehrere Möglichkeiten, Ihrer Ressourcen-ACL Rechte hinzuzufügen. Sie können die AWS Management Console verwenden, die eine Benutzeroberfläche für die Verwaltung von Berechtigungen bereitstellt, ohne dass Sie Code schreiben müssen. Sie können direkt die REST API oder einen der AWS SDKs verwenden. Diese Bibliotheken vereinfachen Ihre Programmieraufgabe noch weiter.

Verwalten von ACLs in der AWS Management Console

AWS Management Console stellt eine Benutzeroberfläche für Sie bereit, um auf der ACL basierende Zugriffsberechtigungen für Ihre Buckets und Objekts erteilen zu können. Für Informationen zum Einrichten ACL-basierter Zugriffsberechtigungen in der Konsole vgl. Wie lege ich ACL-Bucket-Berechtigungen fest? und Wie lege ich Berechtigungen für ein Objekt fest? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwalten von ACLs mit AWS SDK for Java

Dieser Abschnitt enthält Beispiele, wie Zugriffskontrolllisten-(ACL-)Berechtigungen für Buckets und Objekte konfiguriert werden. Das erste Beispiel erstellt einen Bucket mit einer vorgefertigten ACL (siehe Vordefinierte ACL (p. 486)), erstellt eine Liste benutzerdefinierter Berechtigungen und ersetzt die vorgefertigte ACL durch eine ACL mit den benutzerdefinierten Berechtigungen. Das zweite Beispiel veranschaulicht, wie Sie eine ACL mit der Methode `AccessControlList.grantPermission()` abändern.

Festlegen von ACL-Berechtigungen

Example

Dieses Beispiel erstellt einen Bucket. In der Anfrage gibt das Beispiel eine vorgefertigte ACL an, die die Protokollbereitstellungsgruppe zum Schreiben von Protokollen in den Bucket berechtigt.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.ArrayList;

public class CreateBucketWithACL {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String userEmailForReadPermission = "*** user@example.com ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .build();
```

```
            // Create a bucket with a canned ACL. This ACL will be replaced by the
 setBucketAcl()
            // calls below. It is included here for demonstration purposes.
            CreateBucketRequest createBucketRequest = new CreateBucketRequest(bucketName,
 clientRegion.getName())
                    .withCannedAcl(CannedAccessControlList.LogDeliveryWrite);
            s3Client.createBucket(createBucketRequest);

            // Create a collection of grants to add to the bucket.
            ArrayList<Grant> grantCollection = new ArrayList<Grant>();

            // Grant the account owner full control.
            Grant grant1 = new Grant(new
 CanonicalGrantee(s3Client.getS3AccountOwner().getId()), Permission.FullControl);
            grantCollection.add(grant1);

            // Grant the LogDelivery group permission to write to the bucket.
            Grant grant2 = new Grant(GroupGrantee.LogDelivery, Permission.Write);
            grantCollection.add(grant2);

            // Save grants by replacing all current ACL grants with the two we just
 created.
            AccessControlList bucketAcl = new AccessControlList();
            bucketAcl.grantAllPermissions(grantCollection.toArray(new Grant[0]));
            s3Client.setBucketAcl(bucketName, bucketAcl);

            // Retrieve the bucket's ACL, add another grant, and then save the new ACL.
            AccessControlList newBucketAcl = s3Client.getBucketAcl(bucketName);
            Grant grant3 = new Grant(new EmailAddressGrantee(userEmailForReadPermission),
 Permission.Read);
            newBucketAcl.grantAllPermissions(grant3);
            s3Client.setBucketAcl(bucketName, newBucketAcl);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Konfigurieren von ACL-Berechtigungen für ein vorhandenes Objekt

Example

Dieses Beispiel aktualisiert die ACL für ein Objekt. Das Beispiel führt die folgenden Aufgaben durch:

- Ruft die ACL eines Objekts ab
- Löscht die ACL durch Entfernen aller vorhandenen Berechtigungen
- Fügt zwei Berechtigungen hinzu: Vollzugriff für den Eigentümer und WRITE_ACP (siehe Welche Berechtigungen kann ich erteilen? (p. 483)) für einen anhand einer E-Mail-Adresse identifizierten Benutzer
- Speichert die ACL im Objekt

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AccessControlList;
import com.amazonaws.services.s3.model.CanonicalGrantee;
import com.amazonaws.services.s3.model.EmailAddressGrantee;
import com.amazonaws.services.s3.model.Permission;

import java.io.IOException;

public class ModifyACLExistingObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String emailGrantee = "*** user@example.com ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Get the existing object ACL that we want to modify.
            AccessControlList acl = s3Client.getObjectAcl(bucketName, keyName);

            // Clear the existing list of grants.
            acl.getGrantsAsList().clear();

            // Grant a sample set of permissions, using the existing ACL owner for Full
 Control permissions.
            acl.grantPermission(new CanonicalGrantee(acl.getOwner().getId()),
 Permission.FullControl);
            acl.grantPermission(new EmailAddressGrantee(emailGrantee),
 Permission.WriteAcp);

            // Save the modified ACL back to the object.
            s3Client.setObjectAcl(bucketName, keyName, acl);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwalten von ACLs mit AWS SDK für .NET

Dieser Abschnitt enthält Beispiele für das Konfigurieren von ACL-Berechtigungen für Amazon S3-Buckets und -Objekte.

Beispiel 1: Erstellen eines Bucket und Verwenden einer vorgefertigten ACL zum Festlegen von Berechtigungen

Dieses C#-Beispiel erstellt einen Bucket. In der Anfrage gibt der Code auch eine vorgefertigte ACL an, die die Protokollbereitstellungsgruppe zum Schreiben der Protokolle in den Bucket berechtigt.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ManagingBucketACLTest
    {
        private const string newBucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            CreateBucketUseCannedACLAsync().Wait();
        }

        private static async Task CreateBucketUseCannedACLAsync()
        {
            try
            {
                // Add bucket (specify canned ACL).
                PutBucketRequest putBucketRequest = new PutBucketRequest()
                {
                    BucketName = newBucketName,
                    BucketRegion = S3Region.EUW1, // S3Region.US,
                                                  // Add canned ACL.
                    CannedACL = S3CannedACL.LogDeliveryWrite
                };
                PutBucketResponse putBucketResponse = await
 client.PutBucketAsync(putBucketRequest);

                // Retrieve bucket ACL.
                GetACLResponse getACLResponse = await client.GetACLAsync(new GetACLRequest
                {
                    BucketName = newBucketName
                });
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("S3 error occurred. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Beispiel 2: Konfigurieren von ACL-Berechtigungen für ein vorhandenes Objekt

Dieses C#-Beispiel aktualisiert die ACL für ein vorhandenes Objekt. Das Beispiel führt die folgenden Aufgaben durch:

- Ruft die ACL eines Objekts ab.

- Löscht die ACL durch Entfernen aller vorhandenen Berechtigungen.
- Fügt zwei Berechtigungen hinzu: Vollzugriff für den Eigentümer und WRITE_ACP für einen anhand einer E-Mail-Adresse identifizierten Benutzer.
- Speichert die ACL durch Senden einer `PutAcl`-Anfrage.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ManagingObjectACLTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key name ***";
        private const string emailAddress = "*** email address ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            TestObjectACLTestAsync().Wait();
        }
        private static async Task TestObjectACLTestAsync()
        {
            try
            {
                    // Retrieve the ACL for the object.
                    GetACLResponse aclResponse = await client.GetACLAsync(new GetACLRequest
                    {
                        BucketName = bucketName,
                        Key = keyName
                    });

                    S3AccessControlList acl = aclResponse.AccessControlList;

                    // Retrieve the owner (we use this to re-add permissions after we clear
 the ACL).
                    Owner owner = acl.Owner;

                    // Clear existing grants.
                    acl.Grants.Clear();

                    // Add a grant to reset the owner's full permission (the previous clear
 statement removed all permissions).
                    S3Grant fullControlGrant = new S3Grant
                    {
                        Grantee = new S3Grantee { CanonicalUser = owner.Id },
                        Permission = S3Permission.FULL_CONTROL

                    };

                    // Describe the grant for the permission using an email address.
                    S3Grant grantUsingEmail = new S3Grant
                    {
                        Grantee = new S3Grantee { EmailAddress = emailAddress },
```

```
                    Permission = S3Permission.WRITE_ACP
                };
                acl.Grants.AddRange(new List<S3Grant> { fullControlGrant,
 grantUsingEmail });

                // Set a new ACL.
                PutACLResponse response = await client.PutACLAsync(new PutACLRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    AccessControlList = acl
                });
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Verwalten von ACLs mit der REST API

Amazon S3-APIs ermöglichen die Einrichtung eines ACL beim Erstellen eines Buckets oder eines Objekts. Amazon S3 bietet auch eine API für die Einrichtung einer ACL für einen vorhandenen Bucket oder ein Objekt. Diese APIs stellen die folgenden Methoden bereit, um eine ACL einzurichten:

- Einrichten von ACL mit Anfrageheadern – Wenn Sie eine Anfrage senden, um eine Ressource zu erstellen (Bucket oder Objekt), richten Sie eine ACL über die Anfrage-Header ein. Mit Hilfe dieser Header können Sie entweder eine vordefinierte ACL angeben, oder explizit Rechte erteilen (mit expliziter Identifizierung von Empfänger und Berechtigungen).
- Einrichtung von ACL mithilfe des Anfragetextes – Wenn Sie eine Anfrage senden, um ACL für eine vorhandene Ressource einzurichten, können Sie die ACL im Header oder im Text der Anfrage einrichten.

Weitere Informationen über die Unterstützung der REST API für das Verwalten von ACLs finden Sie in den folgenden Abschnitten im Amazon Simple Storage Service API Reference:

- GET Bucket ACL
- PUT Bucket ACL
- GET Object acl
- PUT Object ACL
- PUT Object
- PUT Bucket
- PUT Object – Kopieren
- Initiieren eines mehrteiligen Uploads

Spezifische Anforderungs-Header für Zugriffskontrolllisten (Access Control Lists, ACLs)

Sie können Header verwenden, um Berechtigungen auf der Basis von Zugriffskontrolllisten (Access Control Lists, ACLs) zu erteilen. Standardmäßig sind alle Objekte privat. Nur der Besitzer hat die vollständige Zugriffssteuerung. Wenn Sie ein neues Objekt hinzufügen, können Sie einzelnen AWS-Konten oder vordefinierten Gruppen von Amazon S3 definierte Berechtigungen erteilen. Diese Berechtigungen werden

anschließend der Zugriffskontrollliste (Access Control List, ACL) für das Objekt hinzugefügt. Weitere Informationen finden Sie unter Verwendung von ACLs.

Mit dieser Operation können Sie Zugriffsberechtigungen mit einer der folgenden beiden Methoden erteilen:

- Vordefinierte ACL (**`x-amz-acl`**) – Amazon S3 unterstützt einen Satz vordefinierter ACLs, englisch „Canned ACLs“. Jede vordefinierte ACL hat eine vordefinierte Menge aus Empfängern und Berechtigungen. Weitere Informationen finden Sie unter Vordefinierte ACL (p. 486).
- Zugriffsberechtigungen – Um bestimmten AWS-Konten oder -Gruppen explizit Zugriffsberechtigungen zu erteilen, verwenden Sie die folgenden Header. Jeder Header ist bestimmten Berechtigungen zugeordnet, die Amazon S3 in einer ACL unterstützt. Weitere Informationen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht (p. 480). Im Header geben Sie eine Liste der Empfänger an, die die jeweilige Berechtigung erhalten.
  - x-amz-grant-read
  - x-amz-grant-write
  - x-amz-grant-read-acp
  - x-amz-grant-write-acp
  - x-amz-grant-full-control

Verwenden von Amazon S3 Block Public Access

Die Funktion Amazon S3 Block Public Access bietet Einstellungen für Zugriffspunkte, Buckets und Konten, mit denen Sie den öffentlichen Zugriff auf Amazon S3-Ressourcen verwalten können. Standardmäßig erlauben neue Buckets, Zugriffspunkte und Objekte keinen öffentlichen Zugriff. Benutzer können jedoch Bucket-Richtlinien, Zugriffspunkt-Richtlinien oder Objektberechtigungen ändern, um öffentlichen Zugriff zu ermöglichen. S3 Block Public Access-Einstellungen überschreiben diese Richtlinien und Berechtigungen, damit Sie den öffentlichen Zugriff auf diese Ressourcen einschränken können.

Mit S3 Block Public Access können Administratoren und Bucket-Eigentümer problemlos zentrale Kontrollen zur Begrenzung des öffentlichen Zugriffs auf ihre Amazon S3-Ressourcen einrichten. Diese Kontrollen werden unabhängig davon durchgesetzt, wie die Ressourcen erstellt wurden.

Wenn Amazon S3 eine Anforderung zum Zugriff auf einen Bucket oder ein Objekt erhält, wird ermittelt, ob für den Bucket oder das Konto des Bucket-Eigentümers eine Block Public Access-Einstellung vorliegt. Wenn die Anforderung über einen Zugriffspunkt einging, prüft Amazon S3 auch auf Block Public Access-Einstellungen für den Zugriffspunkt. Wenn eine Block Public Access-Einstellung vorhanden ist, die den angeforderten Zugriff verbietet, lehnt Amazon S3 die Anforderung ab.

Amazon S3 Block Public Access bietet vier Einstellungen. Diese Einstellungen sind voneinander unabhängig und können in beliebiger Kombination verwendet werden. Jede Einstellung kann auf einen Zugriffspunkt, einen Bucket oder ein gesamtes AWS-Konto angewendet werden. Wenn sich die Block Public Accesss-Einstellungen für den Zugriffspunkt, den Bucket oder das Konto unterscheiden, wendet Amazon S3 die restriktivste Kombination der Zugriffspunkt-, Bucket- und Kontoeinstellungen an.

Wenn Amazon S3 ermittelt, ob eine Operation von einer Block Public Access-Einstellung untersagt wird, werden alle Anforderungen abgelehnt, die gegen eine Zugriffspunkt-, Bucket- oder Konto-Einstellung verstoßen.

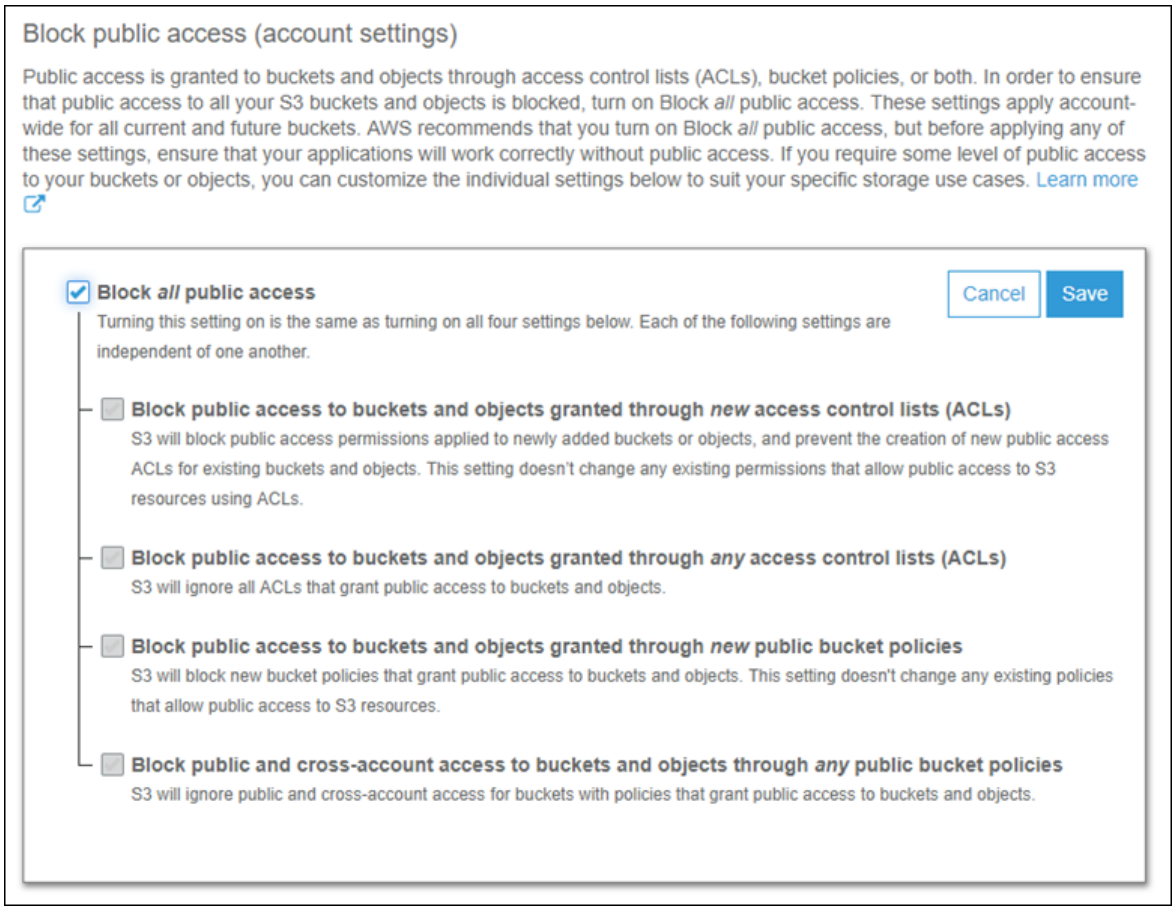
Note

- Sie können Block Public Access-Einstellungen nur für Zugriffspunkte, Buckets und AWS-Konten aktivieren. Amazon S3 unterstützt keine Block Public Access-Einstellungen für einzelne Objekte.
- Wenn Sie die Block Public Access-Einstellungen auf ein Konto anwenden, gelten die Einstellungen global für alle AWS-Regionen. Die Einstellungen werden möglicherweise nicht in allen Regionen umgehend oder gleichzeitig wirksam, werden aber auf jeden Fall von allen Regionen übernommen.

Themen

Aktivieren von Block Public Access auf der Amazon S3-Konsole.

Amazon S3 Block Public Access bietet vier Einstellungen. Sie können diese Einstellungen in beliebiger Kombination auf einzelne Zugriffspunkte, Buckets oder auf ganze AWS-Konten anwenden. Die folgende Abbildung zeigt die Aktivierung von Block Public Access auf der Amazon S3-Konsole für Ihr Konto. Weitere Informationen finden Sie unter Einrichtung von Berechtigungen: Block Public Access im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.



Block Public Access-Einstellungen

S3 Block Public Access bietet vier Einstellungen. Sie können diese Einstellungen in beliebiger Kombination auf einzelne Zugriffspunkte, Buckets oder auf ganze AWS-Konten anwenden. Wenn Sie eine Einstellung auf ein Konto anwenden, gilt sie für alle Buckets und Zugriffspunkte, die dem Konto gehören. Wenn Sie eine Einstellung auf einen Bucket anwenden, gilt diese auch für alle Zugriffspunkte, die diesem Bucket zugeordnet sind.

Die folgende Tabelle enthält die verfügbaren Einstellungen.

Name	Beschreibung
`BlockPublicAcls`	Das Festlegen dieser Option auf `TRUE` hat folgende Verhaltensweise zur Folge: • "PUT Bucket acl"- und "PUT Object acl"-Aufrufe schlagen fehl, wenn die angegebene Zugriffskontrollliste (ACL) öffentlich ist. • "PUT Object"-Aufrufe schlagen fehl, wenn die Anforderung eine öffentliche ACL enthält. • Wenn diese Einstellung auf ein Konto angewendet wird, schlagen "PUT Bucket"-Aufrufe fehl, wenn die Anforderung eine öffentliche ACL enthält. Wen diese Einstellung auf `TRUE` festgelegt ist, schlagen die angegebenen Operationen fehl (gleich, ob sie über die REST-API, die AWS CLI oder AWS-SDKs erfolgen). Vorhandene Richtlinien und ACLs für Buckets und Objekte werden jedoch nicht geändert. Mit dieser Einstellung können Sie den öffentlichen Zugriff einschränken und gleichzeitig die vorhandenen Richtlinien und ACLs für Ihre Buckets und Objekte prüfen, verbessern oder anderweitig ändern. Note Zugriffspunkten sind keine ACLs zugeordnet. Wenn Sie diese Einstellung auf einen Zugriffspunkt anwenden, fungiert sie als Passthrough zum zugrunde liegenden Bucket. Wenn diese Einstellung für einen Zugriffspunkt aktiviert ist, verhalten sich Anforderungen, die über den Zugriffspunkt vorgenommen werden, so als ob der zugrunde liegende Bucket diese Einstellung aktiviert hat, unabhängig davon, ob diese Einstellung für den Bucket tatsächlich aktiviert ist.
`IgnorePublicAcls`	Wenn Sie diese Option auf `TRUE` festlegen, ignoriert Amazon S3 alle öffentlichen ACLs auf einem Bucket und alle darin enthaltenen Objekte. Mit dieser Einstellung können Sie den von ACLs gewährten öffentlichen Zugriff sicher blockieren und gleichzeitig "PUT Object"-Aufrufe zulassen, die eine öffentliche ACL enthalten (im Gegensatz zu `BlockPublicAcls`, das "PUT Object"-Aufrufe zurückweist, die eine öffentliche ACL enthalten). Die Aktivierung dieser Einstellungen wirkt sich nicht auf die Persistenz von vorhandenen ACLs aus und verhindert nicht, dass neue öffentliche ACLs eingerichtet werden. Note Zugriffspunkten sind keine ACLs zugeordnet. Wenn Sie diese Einstellung auf einen Zugriffspunkt anwenden, fungiert sie als Passthrough zum zugrunde liegenden Bucket. Wenn diese Einstellung für einen Zugriffspunkt aktiviert ist, verhalten sich Anforderungen, die über den Zugriffspunkt vorgenommen werden, so als ob der zugrunde liegende Bucket diese Einstellung aktiviert hat, unabhängig davon, ob diese Einstellung für den Bucket tatsächlich aktiviert ist.
`BlockPublicPolicy`	Wenn Sie diese Option auf `TRUE` für einen Bucket einstellen, lehnt Amazon S3 Aufrufe der PUT-Bucket-Richtlinie ab, wenn die angegebene Bucket-Richtlinie den öffentlichen Zugriff zulässt, sowie Aufrufe der PUT-Zugriffspunkt-Richtlinie für alle Zugriffspunkte des Buckets, wenn die angegebene Richtlinie den öffentlichen Zugriff zulässt. Wenn Sie diese Option für einen Zugriffspunkt auf `TRUE` einstellen, lehnt Amazon S3 Aufrufe der PUT-Zugriffspunkt-Richtlinie und der PUT-Bucket-Richtlinie ab, die über den Zugriffspunkt vorgenommen werden,

Name	Beschreibung
	wenn die angegebene Richtlinie (für den Zugriffspunkt oder den zugrunde liegenden Bucket) öffentlich ist. Mit dieser Einstellung können Sie es Benutzern gestatten, Zugriffspunkt- und Bucket-Richtlinien zu verwalten, ohne ihnen die öffentliche Freigabe des Buckets oder der darin enthaltenen Objekte zu erlauben. Die Aktivierung dieser Einstellung hat keine Auswirkungen auf vorhandene Zugriffspunkt- oder Bucket-Richtlinien. **Important** Um diese Einstellung effektiv zu nutzen, empfehlen wir Ihnen, sie auf Konto-Ebene anzuwenden. Eine Bucket-Richtlinie kann Benutzern das Ändern der Block Public Access-Einstellungen eines Buckets gestatten. Daher könnten Benutzer mit der Berechtigung zum Ändern einer Bucket-Richtlinie eine Richtlinie einfügen, die es ihnen erlaubt, die Block Public Access-Einstellungen für den Bucket zu deaktivieren. Wenn diese Einstellung dann für das ganze Konto aktiviert wird statt für einen bestimmten Bucket, blockiert Amazon S3 öffentliche Richtlinien selbst dann, wenn ein Benutzer die Bucket-Richtlinie so ändert, dass diese Einstellung deaktiviert wird.
`RestrictPublicBuckets`	Durch das Festlegen dieser Option auf `TRUE` wird der Zugriff auf einen Zugriffspunkt oder Bucket mit einer öffentlichen Richtlinie auf ausschließlich AWS-Services und autorisierte Benutzer innerhalb des Kontos des Bucket-Eigentümers beschränkt. Diese Einstellung blockiert alle kontenübergreifenden Zugriffe auf den Zugriffspunkt oder Bucket (außer durch AWS-Services) und erlaubt es gleichzeitig Benutzern innerhalb des Kontos, den Zugriffspunkt oder Bucket zu verwalten. Die Aktivierung dieser Einstellung wirkt sich nicht auf vorhandene Zugriffspunkt- oder Bucket-Richtlinien aus – mit folgender Ausnahme: Amazon S3 blockiert den öffentlichen und kontenübergreifenden Zugriff, der von einer öffentlichen Zugriffspunkt- oder Bucket-Richtlinie abgeleitet wird, darunter auch die nicht-öffentliche Delegation auf bestimmte Konten.

Important

- "GET Bucket acl"- und "GET Object acl"-Aufrufe geben immer die effektiven Berechtigungen für den angegebenen Bucket oder das angegebene Objekt zurück. Nehmen wir als Beispiel an, dass ein Bucket über eine ACL verfügt, die den öffentlichen Zugriff zulässt, dass für den Bucket aber auch die Einstellung `IgnorePublicAcls` aktiviert ist. In diesem Fall gibt „GET Bucket acl“ statt der ACL, die tatsächlich mit dem Bucket verknüpft ist, eine ACL zurück, die die von Amazon S3 durchgesetzten Zugriffsberechtigungen widerspiegelt.
- Die Block Public Access-Einstellungen ändern keine vorhandenen Richtlinien oder ACLs. Daher sorgt das Entfernen einer Block Public Access-Einstellung dafür, dass ein Bucket oder ein Objekt mit einer öffentlichen Richtlinie oder ACLs wieder öffentlich zugänglich ist.

Die Bedeutung von „öffentlich“

Buckets

ACLs

Amazon S3 erachtet eine Bucket- oder Objekt-ACL als öffentlich, wenn sie Mitgliedern der vordefinierten Gruppen `AllUsers` oder `AuthenticatedUsers` irgendwelche Berechtigungen erteilt. Weitere Informationen zu vordefinierten Gruppen finden Sie unter Vordefinierte Gruppen in Amazon S3 (p. 482).

Richtlinien

Bei der Evaluierung einer Bucket-Richtlinie beginnt Amazon S3 mit der Annahme, dass die Richtlinie öffentlich ist. Dann evaluiert es die Richtlinie, um festzustellen, ob sie als nicht-öffentlich eingestuft werden kann. Um als nicht-öffentlich zu gelten, darf eine Bucket-Richtlinie nur Zugriff für feste Werte (Werte, die keine Platzhalter aufweisen) gewähren, wie einen oder mehrere der folgenden Werte:

- einen Satz von Classless Inter-Domain Routings (CIDRs), unter Verwendung von `aws:SourceIp`. Weitere Informationen zu CIDR finden Sie unter RFC 4632 auf der RFC-Editor-Website.
- Ein AWS-Prinzipal, Benutzer, Rolle, oder Service-Prinzipal (z. B. `aws:PrincipalOrgID`)
- `aws:SourceArn`
- `aws:SourceVpc`
- `aws:SourceVpce`
- `aws:SourceOwner`
- `aws:SourceAccount`
- `s3:x-amz-server-side-encryption-aws-kms-key-id`
- `aws:userid`, außerhalb des Musters "`AROLEID:*`"
- `s3:DataAccessPointArn`

  Note

  Bei Verwendung in einer Bucket-Richtlinie kann dieser Wert einen Platzhalter für den Namen des Zugriffspunkts enthalten, ohne die Richtlinie öffentlich zu machen, solange die Konto-ID fixiert ist. Das Zulassen des Zugriffs auf `arn:aws:s3:us-west-2:123456789012:accesspoint/*` würde beispielsweise den Zugriff auf jeden Zugriffspunkt ermöglichen, der dem Konto `123456789012` in Region `us-west-2` zugeordnet ist, ohne die Bucket-Richtlinie öffentlich zu machen. Beachten Sie, dass sich dieses Verhalten von dem von Zugriffspunkt-Richtlinien unterscheidet. Weitere Informationen finden Sie unter Zugriffspunkte (p. 498).
- `s3:DataAccessPointAccount`

Nach diesen Regeln gelten die folgenden Beispielrichtlinien als öffentlich.

```
{
  "Principal": { "Federated": "graph.facebook.com" },
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow"
 }
```

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow"
 }
```

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow",
  "Condition": { "StringLike": {"aws:SourceVpc": "vpc-*"}}
```

```
 }
```

Diese Richtlinien können zu nicht-öffentlichen Richtlinien gemacht werden, indem einer der vorgenannten Bedingungsschlüssel unter Verwendung eines festen Wertes eingefügt wird. So können Sie beispielsweise die letzte oben angeführte Richtlinie zu einer nicht-öffentlichen Richtlinie machen, indem Sie `aws:SourceVpc` wie folgt auf einen festen Wert festlegen.

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow",
  "Condition": {"StringEquals": {"aws:SourceVpc": "vpc-91237329"}}
 }
```

Weitere Informationen zu Bucket-Richtlinien finden Sie unter Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375).

Beispiel

Dieses Beispiel zeigt, wie Amazon S3 eine Bucket-Richtlinie bewertet, die Berechtigungen sowohl für den öffentlichen wie auch für den nicht-öffentlichen Zugriff enthält.

Nehmen wir an, dass ein Bucket eine Richtlinie aufweist, die den Zugriff auf einen Satz an festen Prinzipalen gewährt. Nach den zuvor beschriebenen Regeln gilt diese Richtlinie nicht als öffentlich. Wenn Sie daher die Einstellung `RestrictPublicBuckets` aktivieren, bleibt die Richtlinie wie geschrieben in Kraft, da `RestrictPublicBuckets` nur für Buckets mit öffentlichen Richtlinien gilt. Wenn Sie jedoch der Richtlinie eine öffentliche Anweisung hinzufügen, wird `RestrictPublicBuckets` für den Bucket wirksam. Nur AWS Service-Principals und autorisierten Benutzern des Kontos des Bucket-Eigentümers wird der Zugriff auf den Bucket gewährt.

Nehmen wir beispielsweise an, dass ein Bucket, der „Konto-1“ gehört, eine Richtlinie aufweist, die Folgendes enthält:

1. eine Anweisung, die AWS CloudTrail (das ein AWS-Service-Prinzipal ist) Zugriff gewährt
2. eine Anweisung, die Konto "Konto-2" Zugriff gewährt
3. eine Anweisung, die der Öffentlichkeit Zugriff gewährt, z. B. durch das Festlegen von `"Principal": "*"` ohne einschränkende `Condition`

Diese Richtlinie gilt wegen der dritten Anweisung als öffentlich. Wenn diese Richtlinie vorhanden und `RestrictPublicBuckets` aktiviert ist, gewährt Amazon S3 den Zugriff nur durch CloudTrail. Beachten Sie: Obwohl die zweite Anweisung nicht öffentlich ist, deaktiviert Amazon S3 den Zugriff durch „Konto-2“. Das liegt daran, dass die dritte Anweisung die gesamte Richtlinie zu einer öffentlichen Richtlinie macht, so dass `RestrictPublicBuckets` gilt. Somit deaktiviert Amazon S3 den kontenübergreifenden Zugriff, obwohl die Richtlinie den Zugriff an ein bestimmtes Konto („Konto-2“) delegiert. Wenn Sie aber die dritte Anweisung aus der Richtlinie entfernen, gilt die Richtlinie nicht als öffentlich, und `RestrictPublicBuckets` ist nicht mehr gültig. Somit erhält „Konto-2“ wieder Zugriff auf den Bucket, selbst wenn Sie `RestrictPublicBuckets` aktiviert lassen.

Zugriffspunkte

Amazon S3 wertet Block Public Access-Einstellungen für Zugriffspunkte geringfügig anders als für Buckets aus. Die Regeln, die Amazon S3 anwendet, um zu bestimmen, wann eine Zugriffspunkt-Richtlinie öffentlich ist, sind für Zugriffspunkte im Allgemeinen dieselben wie für Buckets, außer in den folgenden Situationen:

- Ein Zugriffspunkt mit einem VPC-Netzwerkursprung wird unabhängig vom Inhalt seiner Zugriffspunkt-Richtlinie immer als nicht öffentlich betrachtet.

- Eine Zugriffspunktrichtlinie, die Zugriff auf eine Gruppe von Zugriffspunkten unter Verwendung von `s3:DataAccessPointArn` gewährt, gilt als öffentlich. Beachten Sie, dass sich dieses Verhalten von dem von Bucket-Richtlinien unterscheidet. Beispielsweise wird eine Bucket-Richtlinie, die Zugriff auf `s3:DataAccessPointArn`-Werte gewährt, die `arn:aws:s3:us-west-2:123456789012:accesspoint/*` entsprechen, nicht als öffentlich betrachtet. Dieselbe Anweisung in einer Zugriffspunkt-Richtlinie würde jedoch den Zugriffspunkt öffentlich machen.

Verwenden von Access Analyzer for S3 zur Überprüfung öffentlicher Buckets

Sie können Buckets mit Bucket-ACLs, Bucket-Richtlinien oder Zugriffspunkt-Richtlinien, die öffentlichen Zugriff gewähren, mithilfe von Access Analyzer for S3 überprüfen. Access Analyzer for S3 macht Sie auf Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen AWS-Konten, einschließlich AWS-Konten außerhalb Ihrer Organisation, Zugriff gewährt wird. Für jeden öffentlichen oder freigegebenen Bucket erhalten Sie Ergebnisse, die die Quelle und die Ebene des öffentlichen oder freigegebenen Zugriffs melden.

Bewaffnet mit dem Wissen, das in den Ergebnissen präsentiert wird, können Sie sofortige und präzise Korrekturmaßnahmen ergreifen. In Access Analyzer for S3 können Sie den gesamten öffentlichen Zugriff auf einen Bucket mit einem einzigen Klick blockieren. Sie können auch einen Drilldown in die Berechtigungseinstellungen auf Bucket-Ebene ausführen, um detaillierte Zugriffsebenen zu konfigurieren. Für bestimmte und geprüfte Anwendungsfälle, die öffentlichen oder freigegebenen Zugriff erfordern, können Sie Ihre Absicht bestätigen und aufzeichnen, dass der Bucket öffentlich oder freigegeben bleibt, indem Sie die Ergebnisse für den Bucket archivieren.

In seltenen Fällen kann es sein, dass Access Analyzer for S3 keine Ergebnisse für einen Bucket meldet, der von einer Amazon S3 Block Public Access-Auswertung als öffentlich gemeldet wird. Dies ist der Fall, da Amazon S3 Block Public Access Richtlinien für aktuelle Aktionen und potenzielle Aktionen überprüft, die in Zukunft hinzugefügt werden könnten, blockiert werden, die dazu führen könnten, dass ein Bucket öffentlich wird. Auf der anderen Seite analysiert Access Analyzer for S3 nur die aktuellen Aktionen, die für den Amazon S3-Service in der Bewertung des Zugriffsstatus angegeben werden.

Weitere Informationen zu Access Analyzer for S3 finden Sie unter Verwenden von Access Analyzer for S3 im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Berechtigungen

Um die Funktionen von Amazon S3 Block Public Access zu nutzen, benötigen Sie die folgenden Berechtigungen:

Operation	Erforderliche Berechtigungen
GET bucket policy status	`s3:GetBucketPolicyStatus`
GET bucket Block Public Access settings	`s3:GetBucketPublicAccessBlock`
PUT bucket Block Public Access settings	`s3:PutBucketPublicAccessBlock`
DELETE bucket Block Public Access settings	`s3:PutBucketPublicAccessBlock`
GET account Block Public Access settings	`s3:GetAccountPublicAccessBlock`
PUT account Block Public Access settings	`s3:PutAccountPublicAccessBlock`
DELETE account Block Public Access settings	`s3:PutAccountPublicAccessBlock`

Operation	Erforderliche Berechtigungen
PUT Zugriffspunkt-Block Public Access-Einstellungen	`s3:PutAccessPointPublicAccessBlock`

Note

Für die DELETE-Operationen sind dieselben Berechtigungen erforderlich wie für die PUT-Operationen. Es gibt keine separaten Berechtigungen für die DELETE-Operationen.

Beispiele

Verwenden von Block Public Access mit der AWS CLI

Sie können Amazon S3 Block Public Access über die AWS CLI verwenden. Welchen Befehl Sie verwenden, hängt davon ab, ob Sie einen Block Public Access-Aufruf auf einem Zugriffspunkt, einem Bucket oder einem Konto durchführen möchten. Weitere Informationen zum Einrichten und Verwenden der AWS CLI finden Sie unter Was ist die AWS Command Line Interface?

- Zugriffspunkt

  Um Block Public Access-Operationen für einen Zugriffspunkt durchzuführen, verwenden Sie den AWS CLI-Service `s3control`. Beachten Sie, dass es derzeit nicht möglich ist, die Block Public Accesss-Einstellungen eines Zugriffspunkts nach dem Erstellen des Zugriffspunkts zu ändern. Die einzige Möglichkeit, Block Public Access-Einstellungen für einen Zugriffspunkt anzugeben, besteht darin, diese beim Erstellen des Zugriffspunkts einzuschließen.

  Bucket

  Um Block Public Access-Operationen auf einem Bucket durchzuführen, verwenden Sie den AWS CLI-Service `s3api`. Folgende Operationen auf Bucket-Ebene verwenden diesen Service:

  - PUT PublicAccessBlock (für einen Bucket)
  - GET PublicAccessBlock (für einen Bucket)
  - DELETE PublicAccessBlock (für einen Bucket)
  - GET BucketPolicyStatus

  Konto

  Um Block Public Access-Operationen auf einem Konto durchzuführen, verwenden Sie den AWS CLI-Service `s3control`. Folgende Operationen auf Konto-Ebene verwenden diesen Service:

  - PUT PublicAccessBlock (für ein Konto)
  - GET PublicAccessBlock (für ein Konto)
  - DELETE PublicAccessBlock (für ein Konto)

Verwenden von Block Public Access mit dem AWS SDK for Java

Die folgenden Beispiele zeigen, wie Sie Amazon S3 Block Public Access mit dem AWS SDK for Java verwenden. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwendung von AWS SDK for Java (p. 809).

Beispiel 1

Dieses Beispiel zeigt, wie Sie mit dem AWS SDK for Java eine Block Public Access-Konfiguration auf einem S3-Bucket einrichten.

```
AmazonS3 client = AmazonS3ClientBuilder.standard()
   .withCredentials(<credentials>)
```

```
    .build();

client.setPublicAccessBlock(new SetPublicAccessBlockRequest()
  .withBucketName(<bucket-name>)
  .withPublicAccessBlockConfiguration(new PublicAccessBlockConfiguration()
    .withBlockPublicAcls(<value>)
    .withIgnorePublicAcls(<value>)
    .withBlockPublicPolicy(<value>)
    .withRestrictPublicBuckets(<value>)));
```

Important

Dieses Beispiel trifft nur für Operationen auf Bucket-Ebene zu, die die `AmazonS3`-Client-Klasse verwenden. Sehen Sie sich für Operationen auf Konto-Ebene das folgende Beispiel an.

Beispiel 2

Dieses Beispiel zeigt, wie Sie eine Block Public Access-Konfiguration auf einem Amazon S3-Konto mit dem AWS SDK for Java einrichten.

```
AWSS3ControlClientBuilder controlClientBuilder = AWSS3ControlClientBuilder.standard();
controlClientBuilder.setRegion(<region>);
controlClientBuilder.setCredentials(<credentials>);

AWSS3Control client = controlClientBuilder.build();
client.putPublicAccessBlock(new PutPublicAccessBlockRequest()
  .withAccountId(<account-id>)
  .withPublicAccessBlockConfiguration(new PublicAccessBlockConfiguration()
    .withIgnorePublicAcls(<value>)
    .withBlockPublicAcls(<value>)
    .withBlockPublicPolicy(<value>)
    .withRestrictPublicBuckets(<value>)));
```

Important

Dieses Beispiel trifft nur für Operationen auf Konto-Ebene zu, die die `AWSS3Control`-Client-Klasse verwenden. Sehen Sie sich für Operationen auf Bucket-Ebene das vorhergehende Beispiel an.

Verwenden von Block Public Access mit anderen AWS-SDKs

Weitere Informationen zur Verwendung von anderen AWS-SDKs finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).

Verwenden von Block Public Access mit den REST-APIs

Weitere Informationen zum Verwenden von Amazon S3 Block Public Access über REST-APIs finden Sie in den folgenden Themen in der Amazon Simple Storage Service API Reference.

- Operationen auf Konto-Ebene
  - PUT PublicAccessBlock
  - GET PublicAccessBlock
  - DELETE PublicAccessBlock

- Operationen auf Bucket-Ebene
  - PUT PublicAccessBlock

- GET PublicAccessBlock
- DELETE PublicAccessBlock
- GET BucketPolicyStatus

Protokollierung und Überwachung in Amazon S3

Die Überwachung ist wichtig, um Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon S3 und Ihrer AWS-Lösungen aufrechtzuerhalten. Sammeln Sie Überwachungsdaten aller Bestandteile Ihrer AWS-Lösung, damit Sie Ausfälle an mehreren Punkten leichter debuggen können. AWS bietet mehrere Tools für die Überwachung Ihrer Amazon S3-Ressourcen und die Reaktion auf mögliche Vorfälle:

Amazon CloudWatch-Alarme

Mit Amazon CloudWatch-Alarmen überwachen Sie eine Metrik über einen von Ihnen definierten Zeitraum. Wenn die Metrik einen bestimmten Schwellenwert überschreitet, wird eine Benachrichtigung an ein Amazon SNS-Thema oder an eine AWS – Auto Scaling-Richtlinie gesendet. CloudWatch-Alarme rufen keine Aktionen auf, weil sie einen besonderen Status aufweisen. Der Status muss sich stattdessen geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).

AWS CloudTrail-Protokolle

CloudTrail liefert Aufzeichnungen der Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in Amazon S3. Mit den von CloudTrail gesammelten Informationen können Sie die an Amazon S3 gestellte Anfrage, die IP-Adresse, von der die Anfrage gestellt wurde, den Initiator der Anfrage, den Zeitpunkt der Anfrage und weitere Angaben bestimmen. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail (p. 750).

Amazon S3-Zugriffsprotokolle

Server-Zugriffsprotokolle enthalten detaillierte Aufzeichnungen über die Anfragen, die an einen Bucket gestellt wurden. Server-Zugriffsprotokolle sind für viele Anwendungen nützlich. Beispielsweise können Zugriffsprotokoll-Informationen bei Sicherheits- und Zugriffsprüfungen nützlich sein. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).

AWS Trusted Advisor

Trusted Advisor stützt sich auf bewährte Methoden, die sich während der gesamten Betriebsgeschichte der Betreuung vieler Hunderttausend AWS-Kunden ergeben haben. Trusted Advisor überprüft Ihre AWS-Umgebung und gibt dann Empfehlungen, sobald sich Möglichkeiten ergeben, Kosten zu senken, die Systemleistung zu verbessern oder Sicherheitslücken zu schließen. Alle AWS-Kunden haben Zugriff auf fünf Trusted Advisor-Prüfungen. Kunden mit dem „Business“- oder „Enterprise“-Support-Plan können alle Trusted Advisor-Prüfungen anzeigen.

Trusted Advisor bietet die folgenden Amazon S3-bezogenen Prüfungen:

- Protokollierungskonfiguration von Amazon S3-Buckets
- Sicherheitsprüfungen für Amazon S3-Buckets mit offenen Zugriffsberechtigungen
- Fehlertoleranzprüfungen für Amazon S3-Buckets, für die kein Versioning aktiviert oder deren Versioning ausgesetzt ist

Weitere Informationen finden Sie unter AWS Trusted Advisor im AWS Support-Benutzerhandbuch.

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls die Protokollierung und Überwachung:

- Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets
- Implementieren Sie die Überwachung mithilfe von AWS-Überwachungstools

- Aktivieren von AWS Config
- Aktivieren der Amazon S3-Server-Zugriffsprotokollierung
- Verwendung von AWS CloudTrail
- Überwachen von AWS-Sicherheitshinweisen

Compliance-Validierung für Amazon S3

Die Sicherheit und Compliance von Amazon S3 wird von externen Prüfern im Rahmen mehrerer AWS-Compliance-Programme bewertet, einschließlich der folgenden:

- System and Organization Controls (SOC)
- Payment Card Industry Data Security Standard (PCI DSS)
- Federal Risk and Authorization Management Program (FedRAMP)
- Health Insurance Portability and Accountability Act (HIPAA)

AWS bietet eine häufig aktualisierte Liste der AWS-Services, die von bestimmten Compliance-Programmen verwendet werden, unter AWS-Services im Bereich von Compliance-Programmen.

Audit-Berichte von Drittanbietern stehen Ihnen zum Download über AWS Artifact zur Verfügung. Weitere Informationen finden Sie unter Herunterladen von Berichten in AWS Artifact.

Weitere Informationen zu AWS-Compliance-Programmen finden Sie unter AWS-Compliance-Programme.

Ihre Compliance-Verantwortung bei der Verwendung von Amazon S3 wird durch die Sensibilität der Daten, die Compliance-Ziele Ihrer Organisation sowie geltende Gesetze und Vorschriften bestimmt. Wenn Ihre Nutzung von Amazon S3 die Einhaltung von Standards wie HIPAA, PCI oder FedRAMP erfordert, stellt AWS Ressourcen zur Unterstützung bereit:

- Kurzanleitungen für Sicherheit und Compliance In diesen Bereitstellungsleitfäden finden Sie wichtige Überlegungen zur Architektur sowie die einzelnen Schritte zur Bereitstellung von sicherheits- und Compliance-orientierten Basisumgebungen in AWS.
- Im Whitepaper Architekturerstellung für HIPAA-Sicherheit und -Compliance wird erläutert, wie Unternehmen AWS nutzen, damit sie die HIPAA-Anforderungen erfüllen können.
- Die AWS-Compliance-Ressourcen enthalten mehrere Arbeitsmappen und Leitfäden, die möglicherweise für Ihre Branche und Ihren Standort relevant sind.
- Mit AWS Config können Sie bewerten, zu welchem Grad die Konfiguration Ihrer Ressourcen den internen Vorgehensweisen, Branchenrichtlinien und Vorschriften entspricht.
- Der AWS-Security Hub liefert einen umfassenden Überblick über den Sicherheitsstatus in AWS. So können Sie die Compliance mit den Sicherheitsstandards und den bewährten Methoden der Branche abgleichen.
- S3 Objektsperre unterstützt Sie dabei, technische Anforderungen von Regulierungsbehörden für Finanzdienstleistungen (wie SEC, FINRA und CFTC) einzuhalten, die den Datenspeicher „Write Once Read Many“ (WORM) für bestimmte Arten von Büchern und Datensatzinformationen erfordern.
- Amazon S3-Bestand (p. 503) können Sie bei der Prüfung und Meldung des Replikations- und Verschlüsselungsstatus Ihrer Objekte für Unternehmens-, Compliance- und regulatorische Anforderungen verwenden.

Amazon S3-Bestand

Amazon S3 Inventory ist eines der Tools, die Amazon S3 bereitstellt, um Ihnen bei der Verwaltung Ihres Speichers zu helfen. Sie können es für die Prüfung und Meldung des Replikations- und Verschlüsselungsstatus Ihrer Objekte für Unternehmens-, Compliance- und regulatorische Anforderungen

verwenden. Außerdem können Sie Business Workflows und Big Data-Aufgaben mithilfe von Amazon S3 Inventory vereinfachen und beschleunigen. Es bietet Ihnen eine geplante Alternative zur Amazon S3-synchronen `List`-API-Operation.

Der Amazon S3-Bestand stellt Comma Separated Values (CSV), Apache Optimized Row Columnar (ORC) oder Apache Parquet (Parquet)-Ausgabedateien bereit, die Ihre Objekte und die zugehörigen Metadaten auf täglicher oder wöchentlicher Basis für einen S3-Bucket oder ein gemeinsam verwendetes Präfix (das heißt, Objekte, deren Namen mit einer allgemeinen Zeichenfolge beginnen) auflisten. Wenn wöchentlich, wird ein Bericht alle sieben Tage nach dem ersten Bericht erstellt. Weitere Informationen zu den Amazon S3 Inventory-Preisen finden Sie unter Amazon S3-Preise.

Sie können mehrere Bestandslisten für einen Bucket konfigurieren. Sie können konfigurieren, welche Objekt-Metadaten in den Bestand aufgenommen werden sollen, ob alle Objektversionen aufgelistet werden oder nur die aktuellen Versionen, wo die Dateiausgabe der Bestandsliste gespeichert werden soll und ob das Bestandsverzeichnis täglich oder wöchentlich generiert werden soll. Sie können auch angeben, dass die Bestandslistendatei verschlüsselt wird.

Sie können den Amazon S3-Bestand auch mittels Standard-SQL abfragen, indem Sie Amazon Athena, Amazon Redshift Spectrum und weitere Tools wie Presto, Apache Hive und Apache Spark verwenden. Sie können Athena problemlos verwenden, um Abfragen zu Ihren Bestandsdateien auszuführen. Sie können Athena für Amazon S3-Inventory-Abfragen in allen Regionen verwenden, in denen Athena verfügbar ist.

Themen

Wie richte ich Amazon S3 Inventory ein?

In diesem Abschnitt wird beschrieben, wie ein Bestand, einschließlich der Details über den Bestand an Quell- und Ziel-Buckets, eingerichtet wird.

Quell- und Ziel-Buckets für Amazon S3 Inventory

Der Bucket, dessen Objekte die Bestandserfassung auflistet, wird als Quell-Bucket bezeichnet. Der Bucket, in dem die Datei mit der Bestandsliste gespeichert ist, wird als Ziel-Bucket bezeichnet.

Quell-Bucket

Der Bestand listet die Objekte auf, die im Quell-Bucket gespeichert sind. Sie können Bestandslisten für einen ganzen Bucket oder nach einem Präfix (Objektschlüsselname) gefiltert erhalten.

Der Quell-Bucket:

- Enthält die Objekte, die im Bestand aufgelistet sind.
- Enthält die Konfiguration für den Bestand.

Ziel-Bucket

Amazon S3 Inventory listet Dateien auf, die in den Ziel-Bucket geschrieben werden. Um alle Bestandslistendateien an einem gemeinsamen Standort im Ziel-Bucket zu gruppieren, können Sie ein Ziel-Präfix (Objektschlüsselname) in der Bestandskonfiguration angeben.

Der Ziel-Bucket:

- Enthält die Dateilisten für den Bestand.
- Enthält die Manifestdateien, die alle Bestandslisten der Datei auflisten, die im Ziel-Bucket gespeichert sind. Weitere Informationen finden Sie unter Was ist ein Bestandsmanifest? (p. 509)
- Braucht eine Bucket-Richtlinie, um Amazon S3 die Berechtigung zu erteilen, das Eigentum am Bucket zu überprüfen, ebenso wie die Berechtigung, Dateien in den Bucket zu schreiben.
- Muss sich in derselben AWS-Region wie der Quell-Bucket befinden.
- Kann gleich dem Quell-Bucket sein.
- Kann einem anderen AWS-Konto gehören als das Konto, dem der Quell-Bucket gehört.

Einrichten von Amazon S3 Inventory

Amazon S3 Inventory hilft Ihnen, Ihren Speicher zu verwalten, indem nach einem definierten Zeitplan Listen der Objekte in einem S3-Bucket erstellt werden. Sie können mehrere Bestandslisten für einen Bucket konfigurieren. Die Bestandslisten werden in CSV-, ORC,- oder Parquet-Dateien in einem Ziel-Bucket veröffentlicht.

Die einfachste Möglichkeit, einen Bestand einzurichten, ist die Verwendung der AWS Management Console, Sie können aber auch REST-API, AWS CLI oder AWS SDKs verwenden. Die Konsole führt den ersten Schritt des folgenden Verfahrens für Sie durch: Hinzufügen einer Bucket-Richtlinie zum Ziel-Bucket.

So richten Sie einen Amazon S3-Bestand für meinen S3-Bucket ein:

1. Fügen Sie eine Bucket-Richtlinie für den Ziel-Bucket hinzu.

   Sie müssen eine Bucket-Richtlinie für den Ziel-Bucket einrichten, um Amazon S3 die Berechtigung zu erteilen, Objekte in den Bucket an dem definierten Standort zu schreiben. Eine Beispielrichtlinie finden Sie unter Gewähren von Berechtigungen für Amazon S3-Bestand und Amazon S3-Analysen (p. 453).
2. Konfigurieren Sie eine Bestandsliste, um die Objekte in einem Quell-Bucket aufzulisten und die Liste in einem Ziel-Bucket zu veröffentlichen.

   Wenn Sie eine Bestandsliste für einen Quell-Bucket konfigurieren, geben Sie den Ziel-Bucket an, in dem die Liste gespeichert werden soll, und ob Sie möchten, dass die Liste täglich oder wöchentlich generiert werden soll. Sie können auch konfigurieren, welche Objekt-Metadaten aufgenommen werden sollen, und ob alle Objektversionen oder nur die aktuellen Versionen aufgelistet werden sollen.

   Sie können angeben, dass die Bestandslistendatei verschlüsselt wird, indem Sie einen von Amazon S3 verwalteten Schlüssel (SSE-S3) oder einen vom Kunden verwalteten AWS Key Management Service (AWS KMS) Kundenhauptschlüssel (Customer Master Key, CMK) verwenden. Weitere Informationen zu SSE-S3 und SSE-KMS finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290). Wenn Sie die SSE-KMS-Verschlüsselung verwenden möchten, siehe Schritt 3.
   - Informationen zur Verwendung der Konsole zum Konfigurieren einer Bestandsliste finden Sie unter Wie konfiguriere ich Amazon S3 Inventory? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
   - Um mit der Amazon S3-API eine Bestandsliste zu konfigurieren, verwenden Sie die REST-API PUT Bucket-Bestandskonfiguration oder das Äquivalent aus der AWS CLI oder den AWS SDKs.
3. Um die Bestandsdatei mit SSE-KMS zu verschlüsseln, erteilen Sie Amazon S3 die Berechtigung, den in AWS KMS gespeicherten CMK zu verwenden.

   Sie können die Verschlüsselung für die Bestandslistendatei konfigurieren, indem Sie die AWS Management Console, die REST-API, AWS CLI oder AWS SDKs verwenden. Egal, welche Möglichkeit Sie wählen, Sie müssen Amazon S3 die Berechtigung gewähren, den vom Kunden verwalteten AWS KMS-CMK zum Verschlüsseln der Bestandsdatei zu verwenden. Sie erteilen Amazon S3 die Berechtigung, indem Sie die Schlüsselrichtlinie für den kundenverwalteten CMK ändern, den Sie zur Verschlüsselung der Bestandsdatei verwenden möchten. Weitere Informationen finden Sie im folgenden

Abschnitt, Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres AWS KMS CMK für die Verschlüsselung (p. 506).

Erteilen der Berechtigung an Amazon S3 zur Verwendung Ihres AWS KMS CMK für die Verschlüsselung

Um Amazon S3 die Berechtigung zum Verschlüsseln mit einem kundenverwalteten AWS Key Management Service-(AWS KMS)-Kundenhauptschlüssel (CMK) zu erteilen, müssen Sie eine Schlüsselrichtlinie verwenden. Gehen Sie folgendermaßen vor, um Ihre Schlüsselrichtlinie zu aktualisieren, damit Sie einen kundenverwalteten AWS KMS CMK zur Verschlüsselung der Bestandsdatei verwenden können.

So erteilen Sie Berechtigungen für die Verschlüsselung mit Ihrem AWS KMS-CMK:

1. Melden Sie sich mit dem AWS-Konto, das zu dem vom Kunden verwalteten CMK gehört, bei der AWS Management Console an.
2. Öffnen Sie die AWS KMS-Konsole unter https://console.aws.amazon.com/kms.
3. Um die AWS-Region zu ändern, verwenden Sie die Regionenauswahl in der oberen rechten Ecke der Seite.
4. Klicken Sie im linken Navigationsbereich auf Customer managed keys (Vom Kunden verwaltete Schlüssel).
5. Wählen Sie unter Customer managed keys (Vom Kunden verwaltete Schlüssel) den vom Kunden verwalteten CMK aus, den Sie zum Verschlüsseln der Bestandsdatei verwenden möchten.
6. Wählen Sie unter Key policy (Schlüsselrichtlinie) die Option Switch to policy view (Zur Richtlinienansicht wechseln) aus.
7. Um die Schlüsselrichtlinie zu aktualisieren, wählen Sie Edit (Bearbeiten).
8. Fügen Sie unter Edit key policy (Schlüsselrichtlinie bearbeiten) die folgende Schlüsselrichtlinie zu der vorhandenen Schlüsselrichtlinie hinzu.

```
{
    "Sid": "Allow Amazon S3 use of the CMK",
    "Effect": "Allow",
    "Principal": {
        "Service": "s3.amazonaws.com"
    },
    "Action": [
        "kms:GenerateDataKey"
    ],
    "Resource": "*"
}
```

9. Wählen Sie Save Changes.

   Weitere Informationen zum Erstellen von kundenverwalteten AWS KMS CMKs und zum Verwenden von Schlüsselrichtlinien finden Sie in den folgenden Themen im AWS Key Management Service Developer Guide:

   - Erste Schritte
   - Verwenden von Schlüsselrichtlinien in AWS KMS

   Sie können auch die AWS KMS-PUT-Schlüsselrichtlinien-API PutKeyPolicy verwenden, um die Schlüsselrichtlinie in den vom Kunden verwalteten CMK zu kopieren, den Sie zur Verschlüsselung der Bestandsdatei verwenden möchten.

Was ist in einem Amazon S3 Inventory enthalten?

Eine Datei mit einer Bestandsliste enthält eine Liste der Objekte im Quell-Bucket sowie die Metadaten für jedes Objekt. Die Bestandslisten werden im Ziel-Bucket als mit GZIP komprimierte CSV-Datei, als für Apache optimierte und mit ZLIB komprimierte ORC-Datei oder als mit Snappy komprimierte Apache Parquet-Datei gespeichert.

Die Bestandsliste enthält eine Liste der Objekte in einem S3-Bucket sowie die folgenden Metadaten für jedes aufgelistete Objekt:

- Bucket name (Bucket-Name) – Der Name des Buckets, für den der Bestand gilt.
- Key name (Schlüsselname) – Name des Objektschlüssels (oder Schlüssel), der das Objekt in dem Bucket eindeutig identifiziert. Bei Verwendung des CSV-Dateiformats ist der Schlüsselname URL-kodiert und muss dekodiert werden, bevor Sie ihn verwenden können.
- Version ID (Versions-ID) – ID der Objektversion. Wenn Sie das Versioning für einen Bucket aktivieren, weist Amazon S3 allen Objekten, die dem Bucket hinzugefügt werden, eine Versionsnummer zu. Weitere Informationen finden Sie unter Objekt-Versioning (p. 127). (Dieses Feld ist nicht enthalten, wenn die Liste nur für die aktuelle Version der Objekte erstellt wird.)
- IsLatest – Auf `True` gesetzt, wenn es sich bei dem Objekt um die aktuelle Version des Objekts handelt. (Dieses Feld ist nicht enthalten, wenn die Liste nur für die aktuelle Version der Objekte erstellt wird.)
- Size (Größe) – Objektgröße in Bytes.
- Last modified date (Letztes Änderungsdatum) – Datum der Erstellung oder der letzten Änderung des Objekts, je nachdem, welches Datum später liegt.
- ETag – Der Entitäts-Tag ist ein Hashwert des Objekts. Das ETag gibt nur Änderungen am Inhalt eines Objekts wieder, nicht an seinen Metadaten. Das ETag kann ein MD5 Digest der Objektdaten sein, muss aber nicht. Dies hängt davon ab, wie das Objekt erstellt und verschlüsselt wurde.
- Storage class (Speicherklasse) – Die für die Speicherung des Objekts verwendete Speicherklasse. Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122).
- Intelligent-Tiering-Zugriff – Zugriffsebene (häufig oder selten) des Objekts bei Speicherung im Intelligent Tiering. Weitere Informationen finden Sie unter Intelligent-Tiering bei Amazon S3.
- Multipart upload flag (Markierung für mehrteiligen Upload) – Auf `True` gesetzt, wenn das Objekt als mehrteiliger Upload hochgeladen wurde. Weitere Informationen finden Sie unter Überblick Multipart Upload (p. 199).
- Delete marker (Löschmarkierung) – Auf `True` gesetzt, wenn es sich bei dem Objekt um eine Löschmarkierung handelt. Weitere Informationen finden Sie unter Objekt-Versioning (p. 127). (Dieses Feld wird dem Bericht automatisch hinzugefügt, wenn Sie den Bericht so konfiguriert haben, dass alle Versionen der Objekte aufgenommen werden.)
- Replication status (Replikationsstatus) – auf `PENDING`, `COMPLETED`, `FAILED` oder `REPLICA.` gesetzt. Für weitere Informationen siehe Informationen zum Replikationsstatus (p. 721).
- Encryption status (Verschlüsselungsstatus) – Auf `SSE-S3`, `SSE-C`, `SSE-KMS` oder `NOT-SSE` gesetzt. Der serverseitige Verschlüsselungsstatus für SSE-S3, SSE-KMS und SSE mit vom Benutzer bereitgestellten Schlüsseln (SSE-C). Der Status `NOT-SSE` bedeutet, dass das Objekt nicht mit serverseitiger Verschlüsselung verschlüsselt ist. Weitere Informationen finden Sie unter Datenschutz durch Verschlüsselung (p. 290).
- Beibehaltungsdatum für S3 Objektsperre – Das Datum, bis zu dem das gesperrte Objekt nicht gelöscht werden kann. Weitere Informationen finden Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).
- S3 Objektsperre Mode (S3 Objektsperre-Modus) – Festgelegt auf `Governance` oder `Compliance` für Objekte, die gesperrt sind. Weitere Informationen finden Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).
- S3 Objektsperre Legal hold status (Rechtlicher Aufbewahrungsstatus von S3 Objektsperre) – Festgelegt auf `On`, wenn für ein Objekt eine rechtliche Aufbewahrungsfrist gilt; andernfalls auf `Off` gesetzt. Weitere Informationen finden Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).

Wir empfehlen, eine Lebenszyklusrichtlinie einzurichten, die alte Bestandslisten löscht. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Bestandskonsistenz

Möglicherweise erscheinen nicht alle Ihre Objekte in jeder Bestandsliste. Die Bestandsliste bietet letztendliche Datenkonsistenz für PUTs für neue Objekte und überschriebene Objekte, ebenso wie für DELETE-Vorgänge. Bestandslisten sind fortlaufende Momentaufnahmen von Bucket-Elemente, die letztendlich konsistent sind (d. h. die Liste enthält möglicherweise keine vor kurzem hinzugefügten oder gelöschten Objekte).

Um den Status des Objekts zu überprüfen, bevor Sie eine Aktion dafür ausführen, empfehlen wir, eine `HEAD Object`-REST API-Anforderung zu stellen, um Metadaten für das Objekt abzurufen, oder die Objekteigenschaften in der Amazon S3-Konsole zu überprüfen. Sie können Objektmetadaten auch mit der AWS CLI oder den AWS SDKs überprüfen. Weitere Informationen finden Sie unter HEAD Object im Amazon Simple Storage Service API Reference.

Wo befinden sich die Bestandslisten?

Wenn eine Bestandsliste veröffentlicht wird, werden die Manifestdateien am folgenden Standort im Ziel-Bucket veröffentlicht.

```
destination-prefix/source-bucket/config-ID/YYYY-MM-DDTHH-MMZ/manifest.json
destination-prefix/source-bucket/config-ID/YYYY-MM-DDTHH-MMZ/manifest.checksum
destination-prefix/source-bucket/config-ID/hive/dt=YYYY-MM-DD-HH-MM/symlink.txt
```

- *destination-prefix* ist das Ziel-Präfix (Objektschlüsselname), das in der Bestandskonfiguration angegeben ist, und das verwendet werden kann, um alle Bestandslistendateien an einem gemeinsamen Speicherort im Ziel-Bucket zu gruppieren.
- *source-bucket* ist der Quell-Bucket, für den die Bestandsliste erstellt wird. Er wird hinzugefügt, um Kollisionen zu vermeiden, wenn mehrere Bestandsberichte von mehreren Quell-Buckets an denselben Ziel-Bucket gesendet werden.
- *config-ID* wird hinzugefügt, um Kollisionen zu vermeiden, wenn mehrere Bestandsberichte vom selben Quell-Bucket an denselben Ziel-Bucket gesendet werden. *config-ID* kommt aus der Bestandsberichtkonfiguration und ist der Name des Berichts, der bei der Einrichtung festgelegt wurde.
- *YYYY-MM-DDTHH-MMZ* ist der Zeitstempel, der sich aus der Startzeit und dem Datum zusammensetzt, an dem die Bestandsberichtserstellung mit dem Scannen des Buckets beginnt, z. B. `2016-11-06T21-32Z`.
- `manifest.json` ist die Manifestdatei.
- `manifest.checksum` ist das MD5 des Inhalts der `manifest.json`-Datei.
- `symlink.txt` ist die Apache Hive-kompatible Manifestdatei.

Die Bestandslisten werden täglich oder wöchentlich am folgenden Standort im Ziel-Bucket veröffentlicht.

```
destination-prefix/source-bucket/config-ID/example-file-name.csv.gz
...
destination-prefix/source-bucket/config-ID/example-file-name-1.csv.gz
```

- *Ziel-Präfix* ist das (Objektschlüsselname-) Präfix, das in der Bestandskonfiguration festgelegt ist. Es kann verwendet werden, um alle Bestandslistendateien an einem gemeinsamen Standort im Ziel-Bucket zu gruppieren.

- *source-bucket* ist der Quell-Bucket, für den die Bestandsliste erstellt wird. Er wird hinzugefügt, um Kollisionen zu vermeiden, wenn mehrere Bestandsberichte von mehreren Quell-Buckets an denselben Ziel-Bucket gesendet werden.
- *example-file-name*`.csv.gz` ist eine der CSV-Bestandsdateien. ORC-Bestandsnamen enden mit der Dateinamenserweiterung `.orc` und Parquet-Bestandsnamen enden mit der Dateinamenserweiterung `.parquet`.

Was ist ein Bestandsmanifest?

Die Manifest-Dateien `manifest.json` und `symlink.txt` beschreiben, wo sich die Bestandsdateien befinden. Wenn eine neue Bestandsliste geliefert wird, wird sie durch eine neue Reihe von Manifestdateien begleitet.

Jedes in der `manifest.json`-Datei enthaltene Manifest bietet Metadaten und andere grundlegende Informationen zu einem Bestand. Diese Informationen beinhalten Folgendes:

- Name des Quell-Buckets
- Name des Ziel-Buckets
- Bestandsversion
- Erstellungszeitstempel im Epochen-Datumsformat, der aus der Startzeit und dem Datum besteht, an dem die Bestandberichtserstellung beginnt, den Bucket zu scannen
- Format und Schema der Bestandsdateien
- Tatsächliche Liste der Bestandsdateien, die im Ziel-Bucket enthalten sind

Immer wenn eine `manifest.json`-Datei geschrieben wird, wird sie von einer `manifest.checksum`-Datei begleitet, die das MD5 des Inhalts der `manifest.json`-Datei ist.

Nachfolgend finden Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen CSV-formatierten Bestand.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-inventory-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "CSV",
    "fileSchema": "Bucket, Key, VersionId, IsLatest, IsDeleteMarker, Size,
 LastModifiedDate, ETag, StorageClass, IsMultipartUploaded, ReplicationStatus,
 EncryptionStatus, ObjectLockRetainUntilDate, ObjectLockMode, ObjectLockLegalHoldStatus",
    "files": [
        {
            "key": "Inventory/example-source-bucket/2016-11-06T21-32Z/
files/939c6d46-85a9-4ba8-87bd-9db705a579ce.csv.gz",
            "size": 2147483647,
            "MD5checksum": "f11166069f1990abeb9c97ace9cdfabc"
        }
    ]
}
```

Nachfolgend finden Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen ORC-formatierten Bestand.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-destination-bucket",
    "version": "2016-11-30",
```

```
    "creationTimestamp" : "1514944800000",
    "fileFormat": "ORC",
    "fileSchema":
 "struct<bucket:string,key:string,version_id:string,is_latest:boolean,is_delete_marker:boolean,size:big
    "files": [
        {
            "key": "inventory/example-source-bucket/data/
d794c570-95bb-4271-9128-26023c8b4900.orc",
            "size": 56291,
            "MD5checksum": "5925f4e78e1695c2d020b9f6eexample"
        }
    ]
}
```

Nachfolgend finden Sie ein Beispiel eines Manifests in einer `manifest.json`-Datei für einen Parquet-formatierten Bestand.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "Parquet",
    "fileSchema": "message s3.inventory { required binary bucket (UTF8); required binary
 key (UTF8); optional binary version_id (UTF8); optional boolean is_latest; optional
 boolean is_delete_marker;  optional int64 size;  optional int64 last_modified_date
 (TIMESTAMP_MILLIS);  optional binary e_tag (UTF8);  optional binary storage_class (UTF8);
  optional boolean is_multipart_uploaded;  optional binary replication_status (UTF8);
 optional binary encryption_status (UTF8);}"
  "files": [
        {
           "key": "inventory/example-source-bucket/data/
d754c470-85bb-4255-9218-47023c8b4910.parquet",
            "size": 56291,
            "MD5checksum": "5825f2e18e1695c2d030b9f6eexample"
        }
    ]
}
```

Die `symlink.txt`-Datei ist eine Apache Hive-kompatible Manifestdatei, mit der Hive Bestandsdateien und die zugehörigen Datendateien automatisch entdecken kann. Das Hive-kompatible Manifest funktioniert mit den Hive-kompatiblen Diensten Athena und Amazon Redshift Spectrum.. Außerdem funktioniert dies mit Hive-kompatiblen Anwendungen, einschließlich Presto, Apache Hive, Apache Spark und vielen anderen.

> **Important**
>
> Die Apache Hive-kompatible Manifestdatei `symlink.txt` funktioniert derzeit nicht mit AWS Glue. Das Lesen von `symlink.txt` mit Apache Hive und Apache Spark wird für ORC- und Parquet--formatierte Bestandsdateien nicht unterstützt.

Woran erkenne ich, ob eine Bestandsliste fertig ist?

Sie können eine Amazon S3-Ereignisbenachrichtigung einrichten, um informiert zu werden, wenn die Prüfsummendatei für das Manifest erstellt wird, woran zu erkennen ist, dass dem Ziel-Bucket eine Bestandsliste hinzugefügt wurde. Das Manifest ist eine aktuelle Liste aller Bestandslisten am Zielstandort.

Amazon S3 kann Ereignisse in einem Amazon Simple Notification Service (Amazon SNS)-Thema, einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange oder einer AWS Lambda-Funktion veröffentlichen. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).

Die folgende Benachrichtigungskonfiguration definiert, dass alle `manifest.checksum`-Dateien, die dem Ziel-Bucket hinzugefügt werden, von der AWS Lambda `cloud-function-list-write`. verarbeitet werden.

```
<NotificationConfiguration>
  <QueueConfiguration>
      <Id>1</Id>
      <Filter>
          <S3Key>
              <FilterRule>
                  <Name>prefix</Name>
                  <Value>destination-prefix/source-bucket</Value>
              </FilterRule>
              <FilterRule>
                  <Name>suffix</Name>
                  <Value>checksum</Value>
              </FilterRule>
          </S3Key>
     </Filter>
     <Cloudcode>arn:aws:lambda:us-west-2:222233334444:cloud-function-list-write</Cloudcode>
     <Event>s3:ObjectCreated:*</Event>
  </QueueConfiguration>
  </NotificationConfiguration>
```

Weitere Informationen finden Sie unter Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda Developer Guide.

Abfragen eines Bestands mit Amazon Athena

Sie können Amazon S3-Bestände mithilfe von Standard-SQL abfragen, indem Sie Amazon Athena in allen Regionen verwenden, in denen Athena verfügbar ist. Um die Verfügbarkeit in Ihrer AWS-Region zu prüfen, sehen Sie sich die Tabelle der AWS-Regionen an.

Athena kann Amazon S3-Bestandsdateien im ORC-, Parquet- oder CSV-Format abfragen. Wenn Sie Athena für die Abfrage des Bestands verwenden, empfehlen wir, dass Sie ORC- oder Parquet-formatierte Bestandsdateien anstelle von CSV verwenden. Die Formate ORC und Parquet bieten eine schnellere Abfrageleistung und niedrigere Abfragekosten. ORC und Parquet sind selbstbeschreibende, typerkennende und spaltenbasierte Datenformate, die für Apache Hadoop entwickelt wurden. Das spaltenbasierte Format lässt den Leser nur die Spalten lesen, entpacken und verarbeiten, die für die aktuelle Abfrage benötigt werden. Die ORC- und Parquet-Formate für Amazon S3-Bestände sind in allen AWS-Regionen verfügbar.

Einstieg in die Verwendung von Athena, um Amazon S3-Bestände abzufragen

1. Erstellen einer Athena-Tabelle. Informationen zum Erstellen einer neuen Tabelle finden Sie unter Erstellen von Tabellen in Amazon Athena im Amazon Athena-Benutzerhandbuch.

   Die folgende Beispielabfrage umfasst alle optionalen Felder in einem nach ORC formatierten Bestandsbericht. Entfernen Sie alle optionalen Felder, die Sie für Ihren Bestand nicht ausgewählt haben, sodass die Abfrage den für Ihren Bestand gewählten Feldern entspricht. Außerdem müssen Sie den Bucket-Namen und den Standort verwenden. Der Standort zeigt auf Ihren Bestandszielpfad, z. B. `s3://destination-prefix/source-bucket/config-ID/hive/`.

   ```
   CREATE EXTERNAL TABLE your_table_name(
     `bucket` string,
     key string,
     version_id string,
     is_latest boolean,
     is_delete_marker boolean,
     size bigint,
   ```

```
  last_modified_date timestamp,
  e_tag string,
  storage_class string,
  is_multipart_uploaded boolean,
  replication_status string,
  encryption_status string,
  object_lock_retain_until_date timestamp,
  object_lock_mode string,
  object_lock_legal_hold_status string
  )
  PARTITIONED BY (dt string)
  ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
  STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
  OUTPUTFORMAT  'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
  LOCATION 's3://destination-prefix/source-bucket/config-ID/hive/';
```

Bei Verwendung von Athena für die Abfrage eines nach Parquet formatierten Bestandsberichts verwenden Sie das folgende SerDe anstelle des ORC SerDe in der `ROW FORMAT SERDE`-Anweisung.

```
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
```

Bei Verwendung von Athena für die Abfrage eines CSV-formatierten Bestandsberichts verwenden Sie in der `ROW FORMAT SERDE`-Anweisung das folgende Parquet SerDe anstelle des ORC SerDe.

```
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
```

2. Um neue Bestandslisten Ihrer Tabelle hinzuzufügen, verwenden Sie den folgenden `MSCK REPAIR TABLE`-Befehl.

```
MSCK REPAIR TABLE your-table-name;
```

3. Nachdem Sie die ersten beiden Schritte ausgeführt haben, können Sie für Ihren Bestand Ad-hoc-Abfragen wie in den folgenden Beispielen veranschaulicht durchführen.

```
# Get list of latest inventory report dates available
SELECT DISTINCT dt FROM your-table-name ORDER BY 1 DESC limit 10;

# Get encryption status for a provided report date.
SELECT encryption_status, count(*) FROM your-table-name WHERE dt = 'YYYY-MM-DD-HH-MM'
 GROUP BY encryption_status;

# Get encryption status for report dates in the provided range.
SELECT dt, encryption_status, count(*) FROM your-table-name
WHERE dt > 'YYYY-MM-DD-HH-MM' AND dt < 'YYYY-MM-DD-HH-MM' GROUP BY dt,
 encryption_status;
```

Weitere Informationen zur Verwendung von Athena finden Sie unter Amazon Athena-Benutzerhandbuch.

Amazon S3 Inventory REST APIs

Die folgenden REST-Operationen werden für Amazon S3-Lagerbestandslisten verwendet.

- DELETE Bucket Inventory
- GET Bucket Inventory
- List Bucket Inventory
- PUT Bucket Inventory

Ausfallsicherheit in Amazon S3

Im Zentrum der globalen AWS-Infrastruktur stehen Regionen und Availability Zones. AWS-Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante Netzwerke mit niedriger Latenz und hohen Durchsätzen verbunden sind. Diese Availability Zones bieten Ihnen eine effektive Methode zum Entwerfen und Betreiben von Anwendungen und Datenbanken. Availability Zones sind in noch größeren Ausmaß hochverfügbar und sie sind fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren. Wenn Sie Ihre Daten über größere geografische Distanzen replizieren müssen, können Sie Replikation (p. 669) verwenden, was ein automatisches, asynchrones Kopieren von Objekten über Buckets in verschiedenen AWS-Regionen hinweg ermöglicht.

Jede AWS-Region verfügt über mehrere Availability Zones. Sie können Ihre Anwendungen über mehrere Availability Zones in derselben Region bereitstellen, um eine bessere Fehlertoleranz und niedriger Latenz zu erzielen. Availability Zones sind mit schnellen, privaten Glasfasernetzwerken verbunden. Dies ermöglicht Ihnen die Nutzung von Anwendungen, für die ein automatischer, unterbrechungsfreier Failover zwischen den Availability Zones eingerichtet ist.

Weitere Informationen über zu AWS-Regionen und Availability Zones finden Sie unter Weltweite AWS-Infrastruktur.

Neben der globalen AWS-Infrastruktur stellt Amazon S3 verschiedene Funktionen bereit, um Ihren Anforderungen an Ausfallsicherheit und Datensicherung gerecht zu werden.

Lebenszykluskonfiguration

Eine Lebenszykluskonfiguration besteht aus einer Reihe von Regeln, mit denen Aktionen definiert werden, die Amazon S3 auf eine Gruppe von Objekten anwendet. Mithilfe der Konfigurationsregeln für den Lebenszyklus können Sie Amazon S3 anweisen, Objekte in kostengünstigere Speicherklassen zu übergeben bzw. zu archivieren oder zu löschen. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Versioning

Das Versioning ermöglicht Ihnen, mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Sie können Versioning verwenden, um sämtliche Versionen aller Objekte in Ihrem Amazon S3 Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen. Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514).

S3 Objektsperre

Mit S3 Objektsperre können Sie Objekte anhand des Modells Write Once Read Many (WORM) speichern. Mit S3 Objektsperre können Sie für eine bestimmte Zeit oder permanent verhindern, dass ein Objekt gelöscht oder überschrieben wird. S3 Objektsperre hilft Ihnen dabei, regulatorische Bestimmungen einzuhalten, die eine WORM-Speicherung verlangen oder einfach nur eine weitere Schutzebene gegen das Ändern und Löschen von Objekten einzurichten. Weitere Informationen finden Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).

Speicherklassen

Amazon S3 bietet eine Reihe von Speicherklassen für die von Ihnen gespeicherten Objekte an. Zwei dieser Speicherklassen (STANDARD_IA und ONEZONE_IA) sind für langlebige Daten konzipiert, auf die selten zugegriffen wird, z. B. Sicherungen. Sie können auch die Speicherklasse S3 Glacier verwenden, um Objekte, auf die Sie nicht in Echtzeit zugreifen müssen, zu archivieren. Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122).

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls die Ausfallsicherheit:

- Aktivieren von Versioning
- Erwägen Sie die regionsübergreifende Replikation von Amazon S3
- Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets

Verschlüsselung von Amazon S3-Sicherungen

Wenn Sie Sicherungen mit Amazon S3 speichern, hängt die Verschlüsselung Ihrer Sicherungen von der Konfiguration dieser Buckets ab. Mit Amazon S3 können Sie die standardmäßige Verschlüsselung für einen S3-Bucket festlegen. Sie können die Standardverschlüsselung in einem Bucket festlegen, sodass alle Objekte beim Speichern im Bucket verschlüsselt werden. Die standardmäßige Verschlüsselung unterstützt in AWS KMS gespeicherte Schlüssel (SSE-KMS). Weitere Informationen finden Sie unter Amazon S3-Standardverschlüsselung für S3-Buckets (p. 72).

Verwenden von Versioning

Das Versioning ermöglicht Ihnen, mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Sie können Versioning verwenden, um sämtliche Versionen aller Objekte in Ihrem Amazon S3 Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen. Wenn Sie das Versioning für einen Bucket aktivieren und Amazon S3 mehrere Schreibanforderungen für dasselbe Objekt gleichzeitig empfängt, werden alle Objekte gespeichert.

Wenn Sie das Versioning für einen Bucket aktivieren, generiert Amazon S3 automatisch eine eindeutige Versions-ID für das Objekt, das gespeichert wird. Beispielsweise könnten Sie in einem Bucket zwei Objekte mit demselben Schlüssel haben, aber mit unterschiedlichen Versions-IDs, wie beispielsweise `photo.gif` (Version 111111) und `photo.gif` (Version 121212).

Versionsfähige Bucktes erlauben Ihnen, Objekte nach einem versehentlichen Löschen oder Überschreiben wiederherzustellen. Beispiel:

- Wenn Sie ein Objekt löschen, fügt Amazon S3 eine Löschmarkierung hinzu, statt des dauerhaft zu entfernen. Die Löschmarkierung wird zur aktuellen Objektversion. Sie können die vorherige Version immer wieder herstellen. Weitere Informationen finden Sie unter Löschen von Objektversionen (p. 527).
- Wenn Sie ein Objekt überschreiben, entsteht eine neue Objektversion im Bucket. Sie können die vorherige Version immer wieder herstellen.

> **Important**
>
> Wenn in dem nicht-versionierten Bucket eine Lebenszyklus-Richtlinie für den Ablauf von Objekten vorhanden ist und Sie dasselbe Verhalten hinsichtlich einer dauerhaften Löschung beim Aktivieren des Versionings beibehalten möchten, müssen Sie eine langfristige Ablaufrichtlinie hinzufügen. Die Lebenszyklus-Richtlinie für den langfristigen Ablauf verwaltet das Löschen der langfristigen Objektversionen im versionsfähigen Bucket. (Ein versionsfähiges Bucket behält eine kurzfristige und null oder mehr langfristige Objektversionen.) Weitere Informationen finden Sie unter How Do I Create a Lifecycle Policy for an S3 Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Buckets können einen von drei Status haben: nicht versionsfähig (Standard), versionsfähig oder mit ausgesetztem Versioning.

> **Important**
>
> Nachdem Sie einen Bucket versionsfähig gemacht haben, kann er nicht in einen nicht versionsfähigen Status zurückgesetzt werden. Sie können das Versioning für diesen Bucket jedoch aussetzen.

Der Versioning-Status gilt für alle (niemals für eine Untermenge) der Objekte in diesem Bucket. Wenn Sie das Versioning für einen Bucket zum ersten Mal aktivieren, werden die Objekte darin anschließend immer als Versionen gespeichert und erhalten eine Versions-ID. Beachten Sie Folgendes:

- Objekte, die in Ihrem Bucket gespeichert waren, bevor Sie den Versioning-Status einrichten, haben die Versions-ID `null`. Wenn Sie das Versioning aktivieren, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht mehr. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anforderungen verarbeitet. Weitere Informationen finden Sie unter Verwalten von Objekten in einem versionsfähigen Bucket (p. 519).
- Der Bucket-Eigentümer (oder ein anderer Benutzer mit geeigneten Berechtigungen) kann das Versioning aussetzen, um zu verhindern, dass sich weitere Objektversionen ansammeln. Wenn Sie das Versioning aussetzen, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anforderungen verarbeitet. Weitere Informationen finden Sie unter Verwalten von Objekten in einem Bucket mit ausgesetztem Versioning (p. 534).

Vorgehensweise zum Konfigurieren des Versionings für einem Bucket

Sie können das Versioning für einen Bucket unter Verwendung einer der folgenden Methoden konfigurieren:

- Konfigurieren des Versionings mithilfe der Amazon S3-Konsole.
- Programmgesteuertes Konfigurieren des Versionings mithilfe der AWS SDKs.

  Sowohl die Konsole als auch die SDKs rufen die REST-API auf, die Amazon S3 für die Verwaltung des Versionings bereitstellt.

  > **Note**
  >
  > Falls nötig, können Sie auch die REST API-Aufrufe in Amazon S3 direkt von Ihrem Code aus durchführen. Allerdings kann dies aufwändig sein, weil Sie Code zur Authentifizierung Ihrer Anforderungen schreiben müssen.

  Jedem von Ihnen erstellten Bucket ist eine Versioning-Subressource zugeordnet (siehe Optionen für die Bucket-Konfiguration (p. 61)). Standardmäßig ist Ihr Bucket nicht versionsfähig, und dementsprechend speichert die Versioning-Subressource eine leere Versioning-Konfiguration.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</VersioningConfiguration>
```

  Um das Versioning zu aktivieren, senden Sie eine Anforderung an Amazon S3 mit einer Versionskonfiguration, die einen Status enthält.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

  Um das Versioning auszusetzen, setzen Sie den Statuswert auf `Suspended`.

Der Bucket-Eigentümer, ein AWS-Konto, das den Bucket erstellt hat (Root-Konto), sowie autorisierte Benutzer können den Versioning-Status eines Buckets konfigurieren. Weitere Informationen zu Berechtigungen finden Sie unter Identity and Access Management in Amazon S3 (p. 329).

Ein Beispiel für die Konfiguration des Versionings finden Sie unter Beispiele für das Aktivieren des Bucket-Versionings (p. 517).

MFA Delete

Sie können optional eine weitere Sicherheitsschicht einführen, indem Sie einen Bucket mit MFA (Multi-Factor Authentication) Delete konfigurieren, sodass für jede der folgenden Operationen eine zusätzliche Authentifizierung erforderlich ist:

- Ändern des Versioning-Status Ihres Buckets
- Dauerhaftes Löschen einer Objektversion

MFA Delete fordert zwei Authentifizierungsformen in Kombination:

- Ihre Sicherheitsanmeldeinformationen
- Die Verkettung einer gültigen Seriennummer, eines Leerzeichens und des sechsstelligen Codes, der auf einem zugelassenen Authentifizierungsgerät angezeigt wird

MFA Delete bietet damit zusätzliche Sicherheit, wenn beispielsweise Ihre Sicherheitsanmeldeinformationen nicht mehr vertrauenswürdig sind.

Für das Aktivieren oder Deaktivieren von MFA Delete verwenden Sie dieselbe API, die Sie für die Konfiguration des Versionings für einen Bucket verwenden. Amazon S3 speichert die MFA Delete-Konfiguration in derselben Versioning-Subressource, in der auch der Versioning-Status des Buckets gespeichert ist.

Die MFA-Löschfunktion kann folgendermaßen ein unbeabsichtigtes Löschen von Buckets verhindern:

- Der Benutzer, der die Aktion startet, muss nachweisen, dass er ein physisches MFA-Gerät mit einem MFA-Code besitzt.
- Es wird eine zusätzliche Ebene der Reibung und Sicherheit zum Löschvorgang hinzugefügt.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>VersioningState</Status>
  <MfaDelete>MfaDeleteState</MfaDelete>
</VersioningConfiguration>
```

> **Note**
>
> Der Bucket-Eigentümer, das AWS-Konto, das den Bucket erstellt hat (Root-Konto), sowie alle autorisierten IAM-Benutzer können das Versioning aktivieren, aber nur der Bucket-Eigentümer (Root-Konto) kann MFA Delete aktivieren. Weitere Informationen finden Sie im AWS-Blog-Beitrag MFA Delete and Versioning.

Um MFA Delete zu nutzen, verwenden Sie ein Hardwaregerät oder ein virtuelles MFA-Gerät, um einen Authentifizierungscode zu generieren. Das folgende Beispiel zeigt einen generierten Authentifizierungscode, angezeigt auf einem Hardwaregerät.

Note

MFA Delete und ein durch MFA geschützter API-Zugriff sind Funktionen, die Schutz in bestimmten Situationen bieten sollen. Sie konfigurieren MFA Delete für einen Bucket, um sicherzustellen, dass die Daten in Ihrem Bucket nicht versehentlich gelöscht werden können. Der durch MFA geschützte API-Zugriff wird verwendet, um einen zusätzlichen Authentifizierungsfaktor (MFA-Code) einzuführen, wenn Sie auf sensible Amazon S3-Ressourcen zugreifen. Sie können für alle Operationen für Amazon S3-Ressourcen fordern, dass sie mit temporären Anmeldeinformationen ausgeführt werden, die mit MFA erstellt wurden. Ein Beispiel finden Sie unter Hinzufügen einer Bucket-Richtlinie zur Anforderung einer MFA (p. 451).
Weitere Informationen dazu, wie Sie ein Authentifizierungsgerät kaufen und aktivieren, finden Sie unter https://aws.amazon.com/iam/details/mfa/.

Verwandte Themen

Weitere Informationen finden Sie unter den folgenden Themen:

- Beispiele für das Aktivieren des Bucket-Versionings (p. 517)
- Verwalten von Objekten in einem versionsfähigen Bucket (p. 519)
- Verwalten von Objekten in einem Bucket mit ausgesetztem Versioning (p. 534)
- Deutliche Zunahme von HTTP 503-Antworten auf Amazon S3-Anfragen an Buckets mit aktiviertem Versioning (p. 773)

Beispiele für das Aktivieren des Bucket-Versionings

Themen

- Verwenden der Amazon S3-Konsole (p. 517)
- Using the AWS SDK for Java (p. 517)
- Using the AWS SDK for .NET (p. 518)
- Verwenden anderer AWS SDKs (p. 519)

Dieser Abschnitt zeigt Beispiele, wie das Versioning für einen Bucket aktiviert werden kann. Die Beispiele aktivieren zuerst das Versioning für einen Bucket und rufen dann den Versioning-Status ab. Eine Einführung finden Sie unter Verwenden von Versioning (p. 514).

Verwenden der Amazon S3-Konsole

Weitere Informationen über das Aktivieren des Versionings unter Verwendung der Amazon S3-Konsole finden Sie unter Wie aktiviere ich Versioning für einen S3-Bucket oder setze dieses aus? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Using the AWS SDK for Java

Example

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Region;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.AmazonS3Exception;
```

```
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.SetBucketVersioningConfigurationRequest;

public class BucketVersioningConfigurationExample {
    public static String bucketName = "*** bucket name ***";
    public static AmazonS3Client s3Client;

    public static void main(String[] args) throws IOException {
        s3Client = new AmazonS3Client(new ProfileCredentialsProvider());
        s3Client.setRegion(Region.getRegion(Regions.US_EAST_1));
        try {

            // 1. Enable versioning on the bucket.
         BucketVersioningConfiguration configuration =
           new BucketVersioningConfiguration().withStatus("Enabled");

   SetBucketVersioningConfigurationRequest setBucketVersioningConfigurationRequest =
     new SetBucketVersioningConfigurationRequest(bucketName,configuration);

   s3Client.setBucketVersioningConfiguration(setBucketVersioningConfigurationRequest);

   // 2. Get bucket versioning configuration information.
   BucketVersioningConfiguration conf =
 s3Client.getBucketVersioningConfiguration(bucketName);
    System.out.println("bucket versioning configuration status:    " + conf.getStatus());

        } catch (AmazonS3Exception amazonS3Exception) {
            System.out.format("An Amazon S3 error occurred. Exception: %s",
 amazonS3Exception.toString());
        } catch (Exception ex) {
            System.out.format("Exception: %s", ex.toString());
        }
    }
}
```

Using the AWS SDK for .NET

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using System;
using Amazon.S3;
using Amazon.S3.Model;

namespace s3.amazon.com.docsamples
{
    class BucketVersioningConfiguration
    {
        static string bucketName = "*** bucket name ***";

        public static void Main(string[] args)
        {
            using (var client = new AmazonS3Client(Amazon.RegionEndpoint.USEast1))
            {
                try
                {
                    EnableVersioningOnBucket(client);
                    string bucketVersioningStatus =
 RetrieveBucketVersioningConfiguration(client);
                }
                catch (AmazonS3Exception amazonS3Exception)
                {
```

```
                    if (amazonS3Exception.ErrorCode != null &&
                        (amazonS3Exception.ErrorCode.Equals("InvalidAccessKeyId")
                        ||
                        amazonS3Exception.ErrorCode.Equals("InvalidSecurity")))
                    {
                        Console.WriteLine("Check the provided AWS Credentials.");
                        Console.WriteLine(
                        "To sign up for service, go to http://aws.amazon.com/s3");
                    }
                    else
                    {
                        Console.WriteLine(
                         "Error occurred. Message:'{0}' when listing objects",
                         amazonS3Exception.Message);
                    }
                }
            }

            Console.WriteLine("Press any key to continue...");
            Console.ReadKey();
        }

        static void EnableVersioningOnBucket(IAmazonS3 client)
        {

                PutBucketVersioningRequest request = new PutBucketVersioningRequest
                {
                    BucketName = bucketName,
                    VersioningConfig = new S3BucketVersioningConfig
                    {
                        Status = VersionStatus.Enabled
                    }
                };

                PutBucketVersioningResponse response = client.PutBucketVersioning(request);
        }


        static string RetrieveBucketVersioningConfiguration(IAmazonS3 client)
        {
                GetBucketVersioningRequest request = new GetBucketVersioningRequest
                {
                    BucketName = bucketName
                };

                GetBucketVersioningResponse response = client.GetBucketVersioning(request);
                return response.VersioningConfig.Status;
            }
    }
}
```

Verwenden anderer AWS SDKs

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Verwalten von Objekten in einem versionsfähigen Bucket

Themen

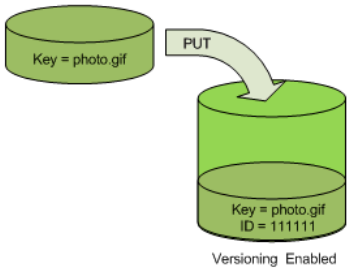
- Hinzufügen von Objekten zu versionierungsfähigen Buckets (p. 520)
- Auflisten von Objekten in einem versionsfähigen Bucket (p. 521)
- Abrufen von Objektversionen (p. 525)

Objekte, die in Ihrem Bucket gespeichert waren, bevor Sie den Versioning-Status einrichten, haben die Versions-ID `null`. Wenn Sie das Versioning aktivieren, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht mehr. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anforderungen verarbeitet. Die Themen in diesem Abschnitt erklären verschiedene Objektoperationen in einem Bucket mit aktiviertem Versioning.

Hinzufügen von Objekten zu versionierungsfähigen Buckets

Wenn Sie das Versioning für einen Bucket aktivieren, fügt Amazon S3 jedem in dem Bucket gespeicherten Objekt automatisch eine eindeutige Versions-ID hinzu (mit `PUT`, `POST` oder `COPY`).

Die folgende Abbildung zeigt, dass Amazon S3 jedem Objekt automatisch eine eindeutige Versions-ID hinzufügt, wenn es einem versionsfähigen Bucket hinzugefügt wird.



Themen

- Verwenden der Konsole (p. 520)
- Verwenden der AWS SDKs (p. 520)
- Verwenden der REST API (p. 520)

Verwenden der Konsole

Anleitungen finden Sie unter Dateien und Ordner in einen S3-Bucket hochladen im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für das Hochladen von Objekten mit den AWS SDKs for Java, .NET und PHP finden Sie unter Objekte hochladen (p. 193). Die Beispiele für das Hochladen von Objekten in versionsfähige und nicht versionsfähige Buckets sind gleich, aber bei versionsfähigen Buckets weist Amazon S3 eine Versionsnummer zu. Andernfalls ist die Versionsnummer null.

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Verwenden der REST API

Hinzufügen von Objekten zu versionierungsfähigen Buckets

1	Aktivieren des Versionings für einen Bucket mit einer `PUT Bucket versioning`-Anforderung. Weitere Informationen finden Sie unter PUT Bucket-Versioning.

2	Senden Sie eine `PUT`-, `POST`- oder `COPY`-Anforderung, um ein Objekt im Bucket zu speichern.

Wenn Sie einem versionsfähigen Bucket ein Objekt hinzufügen, gibt Amazon S3 die Versions-ID des Objekts im Antwort-Header `x-amz-version-id` zurück, z. B.:

```
x-amz-version-id: 3/L4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY
```

> **Note**
>
> Für jede Version eines gespeicherten und übertragenen Objekts gelten die Standardpreise von Amazon S3. Jede Version eines Objekts besteht aus dem vollständigen Objekt, nicht nur aus einem Delta gegenüber der vorherigen Version. Wenn Sie also drei Versionen eines Objekts gespeichert haben, fallen Gebühren für die drei Objekte an.

> **Note**
>
> Die von Amazon S3 zugewiesenen Versions-ID-Werte sind URL-sicher (können als Teil einer URI angegeben werden).

Auflisten von Objekten in einem versionsfähigen Bucket

Themen

- Verwenden der Konsole (p. 521)
- Verwenden der AWS SDKs (p. 521)
- Verwenden der REST API (p. 524)

Dieser Abschnitt zeigt ein Beispiel, wie Objektversionen aus einem versionsfähigen Bucket aufgelistet werden. Amazon S3 speichert Objektversionsinformationen in der Subressource Versionen (vgl. Optionen für die Bucket-Konfiguration (p. 61)), die mit dem Bucket verbunden ist.

Verwenden der Konsole

Informationen zum Auflisten von Objektversionen in der Amazon S3-Konsole finden Sie unter Wie erkenne ich die Versionen eines S3-Objekts? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwenden der AWS SDKs

Die Beispiele in diesem Abschnitt veranschaulichen, wie Sie eine Objektliste aus einem versionsfähigen Bucket abrufen. Jede Anforderung gibt bis zu 1000 Versionen zurück, sofern Sie keine kleinere Anzahl festlegen. Wenn die Versionen im Bucket dieses Limit überschreiten, senden Sie eine Reihe von Anforderungen, um die Liste aller Versionen abzurufen. Dieser Vorgang zur „seitenweisen“ Rückgabe von Ergebnissen wird als Paginierung bezeichnet. Um zu veranschaulichen, wie Paginierung funktioniert, limitieren die Beispiele jede Antwort auf zwei Objektversionen. Nachdem die erste Seite mit Ergebnissen abgerufen wurde, wird in jedem Beispiel überprüft, ob die Versionsliste abgeschnitten wurde. Wurde sie abgeschnitten, dann wird im Beispiel mit dem seitenweisen Abruf fortgefahren, bis alle Versionen abgerufen wurden.

> **Note**
>
> Die folgenden Beispiele funktionieren auch bei einem Bucket, der versionsfähig ist, oder bei Objekten ohne individuelle Versionen. In solchen Fällen gibt Amazon S3 die Objektliste mit der Versions-ID `null` zurück.

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Verwendung von AWS SDK for Java

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListVersionsRequest;
import com.amazonaws.services.s3.model.S3VersionSummary;
import com.amazonaws.services.s3.model.VersionListing;

public class ListKeysVersioningEnabledBucket {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Retrieve the list of versions. If the bucket contains more versions
            // than the specified maximum number of results, Amazon S3 returns
            // one page of results per request.
            ListVersionsRequest request = new ListVersionsRequest()
                    .withBucketName(bucketName)
                    .withMaxResults(2);
            VersionListing versionListing = s3Client.listVersions(request);
            int numVersions = 0, numPages = 0;
            while (true) {
                numPages++;
                for (S3VersionSummary objectSummary :
                        versionListing.getVersionSummaries()) {
                    System.out.printf("Retrieved object %s, version %s\n",
                            objectSummary.getKey(),
                            objectSummary.getVersionId());
                    numVersions++;
                }
                // Check whether there are more pages of versions to retrieve. If
                // there are, retrieve them. Otherwise, exit the loop.
                if (versionListing.isTruncated()) {
                    versionListing = s3Client.listNextBatchOfVersions(versionListing);
                } else {
                    break;
                }
            }
            System.out.println(numVersions + " object versions retrieved in " + numPages +
 " pages");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
```

```
        }
    }
}
```

Using the AWS SDK for .NET

Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ListObjectsVersioningEnabledBucketTest
    {
        static string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main(string[] args)
        {
            s3Client = new AmazonS3Client(bucketRegion);
            GetObjectListWithAllVersionsAsync().Wait();
        }

        static async Task GetObjectListWithAllVersionsAsync()
        {
            try
            {
                ListVersionsRequest request = new ListVersionsRequest()
                {
                    BucketName = bucketName,
                    // You can optionally specify key name prefix in the request
                    // if you want list of object versions of a specific object.

                    // For this example we limit response to return list of 2 versions.
                    MaxKeys = 2
                };
                do
                {
                    ListVersionsResponse response = await
 s3Client.ListVersionsAsync(request);
                    // Process response.
                    foreach (S3ObjectVersion entry in response.Versions)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }

                    // If response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.KeyMarker = response.NextKeyMarker;
                        request.VersionIdMarker = response.NextVersionIdMarker;
                    }
```

```
                    else
                    {
                        request = null;
                    }
                } while (request != null);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Verwenden der REST API

Zur Auflistung aller Versionen aller Objekte in einem Bucket verwenden Sie die Subressource `versions` in einer `GET Bucket`-Anforderung. Amazon S3 kann maximal 1000 Objekte abrufen. Jede Objektversion zählt als vollständiges Objekt. Wenn ein Bucket also zwei Schlüssel enthält (z. B. `photo.gif` und `picture.jpg`) und der erste Schlüssel 990 Versionen und der zweite Schlüssel 400 Versionen hat, ruft eine einzelne Abfrage alle 990 Versionen von `photo.gif` und nur die 10 neuesten Versionen von `picture.jpg` ab.

Amazon S3 gibt Objektversionen in der Reihenfolge zurück, in der sie gespeichert wurden, wobei die zuletzt gespeicherte zuerst zurückgegeben wird.

Alle Objektversionen in einem Bucket auflisten

- Geben Sie in einer `GET Bucket`-Anforderung die `versions`-Subressource an.

```
GET /?versions HTTP/1.1
Host: bucketName.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 +0000
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Abrufen einer Untermenge von Objekten in einem Bucket

Dieser Abschnitt beschreibt die beiden folgenden Beispielszenarien:

- Sie wollen eine Untermenge aller Objektversionen in einem Bucket abrufen, z. B. alle Versionen eines bestimmten Objekts.
- Die Anzahl der Objektversionen in der Antwort überschreitet den Wert für `max-key` (standardmäßig 1000), sodass Sie eine zweite Anforderung senden müssen, um die restlichen Objektversionen abzurufen.

Um eine Untermenge von Objektversionen abzurufen, verwenden Sie die Anforderungsparameter für GET Bucket. Weitere Informationen finden Sie unter GET Bucket.

Beispiel 1: Abrufen aller Versionen eines spezifischen Objekts

Sie können mit der `versions`-Subressource und dem `prefix`-Anforderungparameter unter Verwendung des folgenden Prozesses alle Versionen eines Objekts abrufen. Weitere Informationen zu `prefix` finden Sie unter GET Bucket.

Abruf aller Versionen eines Schlüssels

1	Setzen Sie den Parameter `prefix` auf den Schlüssel des Objekts, das Sie abrufen wollen.
2	Senden Sie eine `GET Bucket`-Anforderung unter Verwendung der `versions`-Subressource und `prefix`. `GET /?versions&prefix=objectName HTTP/1.1`

Example Abrufen von Objekten unter Verwendung eines Präfix

Das folgende Beispiel ruft Objekte ab, deren Schlüssel `myObject` ist oder damit beginnt.

```
GET /?versions&prefix=myObject HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Sie können die anderen Anforderungsparameter verwenden, um eine Untermenge aller Versionen des Objekts abzurufen. Weitere Informationen finden Sie unter GET Bucket.

Beispiel 2: Abrufen einer Liste zusätzlicher Objekte, falls die Antwort gekürzt wurde

Wenn die Anzahl der Objekte, die in einer `GET`-Anforderung zurückgegeben werden können, den Wert `max-keys` überschreitet, enthält die Antwort `<isTruncated>true</isTruncated>`, ebenso wie den ersten Schlüssel (in `NextKeyMarker`) und die erste Versions-ID (in `NextVersionIdMarker`), die die Anforderungskriterien erfüllen, aber nicht zurückgegeben wurden. Diese zurückgegebenen Werte verwenden Sie als Ausgangspunkt in einer nachfolgenden Anforderung, um die zusätzlichen Objekte abzurufen, die die `GET`-Anforderung erfüllen.

Gehen Sie wie folgt vor, um zusätzlichen Objekte abzurufen, die die ursprüngliche `GET Bucket versions`-Anforderung von einem Bucket erfüllen. Weitere Informationen zu `key-marker`, `version-id-marker`, `NextKeyMarker` und `NextVersionIdMarker` finden Sie unter GET Bucket.

Abrufen zusätzlicher Antworten, die die ursprüngliche GET-Anforderung erfüllen

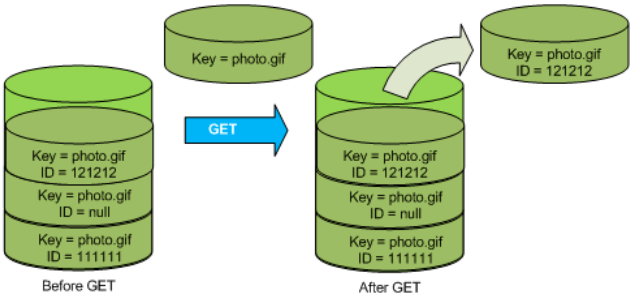
1	Setzen Sie den Wert von `key-marker` auf den Schlüssel, der in `NextKeyMarker` in der vorherigen Antwort zurückgegeben wurde.
2	Setzen Sie den Wert von `version-id-marker` auf die Versions-ID, die in `NextVersionIdMarker` in der vorherigen Antwort zurückgegeben wurde.
3	Senden Sie eine `GET Bucket versions`-Anforderung mit `key-marker` und `version-id-marker`.

Example Abrufen von Objekten ab einem bestimmten Schlüssel und einer bestimmten Versions-ID

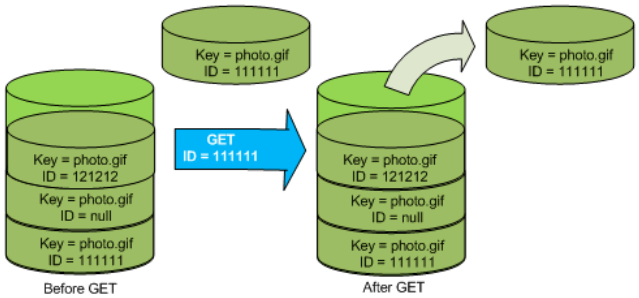
```
GET /?versions&key-marker=myObject&version-id-marker=298459348571 HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Abrufen von Objektversionen

Eine einfache `GET`-Anforderung ruft die aktuelle Version eines Objekts ab. Die folgende Abbildung zeigt, wie `GET` die aktuelle Version des Objekts zurückgibt, `photo.gif`.



Um eine spezifische Version abzurufen, müssen Sie ihre Versions-ID angeben. Die folgende Abbildung zeigt, dass eine `GET versionId`-Anforderung die angegebene Version des Objekts zurückgibt (nicht unbedingt die aktuelle).



Verwenden der Konsole

Anleitungen finden Sie unter Wie erkenne ich die Versionen eines S3-Objekts? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für das Hochladen von Objekten mit den AWS SDKs for Java, .NET und PHP finden Sie unter Objekte abrufen (p. 184). Die Beispiele für das Hochladen von Objekten in versionsfähige und nicht versionsfähige Buckets sind gleich, aber bei versionsfähigen Buckets weist Amazon S3 eine Versionsnummer zu. Andernfalls ist die Versionsnummer null.

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Verwenden von REST

Abrufen einer spezifischen Objektversion

1. Setzen Sie den Parameter `versionId` auf die ID der Version des Objekts, die Sie abrufen wollen.
2. Senden Sie eine `GET Object versionId` -Anforderung.

Example Abrufen eines versionierten Objekts

Die folgende Anforderung ruft die Version L4kqtJlcpXroDTDmpUMLUo von `my-image.jpg` ab.

```
GET /my-image.jpg?versionId=L4kqtJlcpXroDTDmpUMLUo HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Verwandte Themen

Abrufen der Metadaten einer Objektversion (p. 527)

Abrufen der Metadaten einer Objektversion

Wenn Sie nur die Metadaten eines Objekts abrufen wollen (nicht seinen Inhalt), verwenden Sie die `HEAD`-Operation. Standardmäßig erhalten Sie die Metadaten der aktuellsten Version. Um die Metadaten einer spezifischen Objektversion abzurufen, müssen Sie ihre Versions-ID angeben.

Abruf der Metadaten einer Objektversion

1. Setzen Sie den Parameter `versionId` auf die ID der Version des Objekts, dessen Metadaten Sie abrufen wollen.
2. Senden Sie eine `HEAD Object versionId`-Anforderung.

Example Abrufen der Metadaten eines versionierten Objekts

Die folgende Anforderung ruft die Metadaten der Version 3HL4kqCxf3vjVBH40Nrjfkd von `my-image.jpg` ab.

```
HEAD /my-image.jpg?versionId=3HL4kqCxf3vjVBH40Nrjfkd HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Nachfolgend ist eine Beispielantwort angezeigt.

```
HTTP/1.1 200 OK
x-amz-id-2: ef8yU9AS1ed4OpIszj7UDNEHGran
x-amz-request-id: 318BC8BC143432E5
x-amz-version-id: 3HL4kqtJlcpXroDTDmjVBH40Nrjfkd
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2006 12:00:00 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234
Content-Type: text/plain
Connection: close
Server: AmazonS3
```

Löschen von Objektversionen

Sie können jederzeit Objektversionen löschen. Darüber hinaus können Sie Lebenszykluskonfigurationsregeln für Objekte definieren, die einen definieren Lebenszyklus haben, um Amazon S3 aufzufordern, die aktuellen Objektversionen ablaufen zu lassen, oder die nicht aktuellen Objektversionen ständig zu entfernen. Wenn Ihr Bucket versionsfähig ist oder das Versioning ausgesetzt ist, funktionieren die Lebenszykluskonfigurationsaktionen wie folgt:

- Die `Expiration`-Aktion wird auf die aktuelle Objektversion angewendet, und statt die aktuelle Objektversion zu löschen, behält Amazon S3 die aktuelle Version als nicht aktuelle Version bei, indem es eine Löschmarkierung hinzufügt, die anschließend zur aktuellen Version wird.
- Die `NoncurrentVersionExpiration`-Aktion wird nur auf nicht aktuelle Objektversionen angewendet, und Amazon S3 entfernt diese Objektversionen dauerhaft. Dauerhaft entfernte Objekte können nicht wiederhergestellt werden.

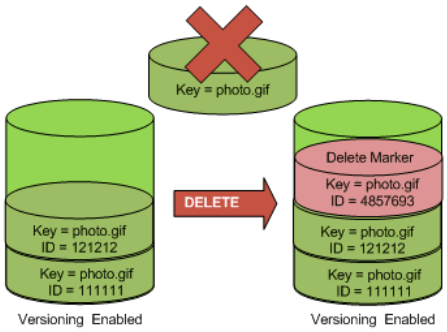
Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Eine `DELETE`-Anforderung hat die folgenden Anwendungsfälle:

- Wenn das Versioning aktiviert ist, kann ein einfaches `DELETE` ein Objekt nicht dauerhaft löschen.

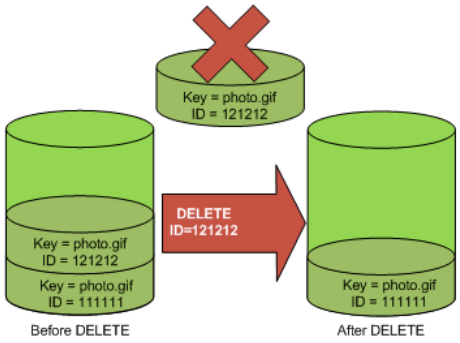
  Stattdessen fügt Amazon S3 eine Löschmarkierung in den Bucket ein, und die Löschmarkierung wird zur aktuellen Objektversion mit einer neuen ID. Wenn Sie versuchen, `GET` für ein Objekt auszuführen, dessen aktuelle Version eine Löschmarkierung ist, verhält sich Amazon S3 so, als wäre das Objekt gelöscht worden (obwohl es nicht vollständig entfernt wurde), und gibt einen 404-Fehler zurück.

  Die folgende Abbildung zeigt, dass ein einfaches `DELETE` das spezifizierte Objekt nicht wirklich löscht. Stattdessen fügt Amazon S3 eine Löschmarkierung ein.



- Um Objektversionen dauerhaft zu löschen, müssen Sie `DELETE Object versionId` verwenden.

  Die folgende Abbildung zeigt, dass das beim Löschen einer angegebenen Objektversion dieses Objekt dauerhaft gelöscht wird.



Verwenden der Konsole

Anleitungen finden Sie unter Wie erkenne ich die Versionen eines S3-Objekts? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwenden der AWS SDKs

Beispiele für das Löschen von Objekten mit den AWS SDKs for Java, .NET und PHP finden Sie unter Löschen von Objekten (p. 250). Die Beispiele für das Löschen von Objekten in versionsfähigen und nicht versionsfähigen Buckets sind gleich, aber bei versionsfähigen Buckets weist Amazon S3 eine Versionsnummer zu. Andernfalls ist die Versionsnummer null.

Weitere Informationen zur Verwendung anderer AWS SDKs finden Sie unter Beispiel-Code und Bibliotheken.

Verwenden von REST

Löschen einer spezifischen Objektversion

- Geben Sie in einem `DELETE` eine Versions-ID ein.

Example Löschen einer spezifischen Version

Das folgende Beispiel zeigt, wie Version UIORUnfnd89493jJFJ von `photo.gif` gelöscht wird.

```
DELETE /photo.gif?versionId=UIORUnfnd89493jJFJ HTTP/1.1

Host: bucket.s3.amazonaws.com
Date: Wed, 12 Oct 2009 17:50:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:xQE0diMbLRepdf3YB+FIEXAMPLE=
Content-Type: text/plain
Content-Length: 0
```

Verwandte Themen

Verwenden von MFA Delete (p. 529)

Arbeiten mit Löschmarkierungen (p. 529)

Entfernen von Löschmarkierungen (p. 531)

Verwenden von Versioning (p. 514)

Verwenden von MFA Delete

Wenn die Versioning-Konfiguration MFA Delete unterstützt, muss der Bucket-Eigentümer den `x-amz-mfa`-Anforderungs-Header in Anforderungen aufnehmen, um eine Objektversion dauerhaft zu löschen oder den Versioning-Status des Buckets zu ändern. Anforderung mit `x-amz-mfa` müssen HTTPS verwenden. Der Wert des Headers ist die Verkettung der Seriennummer Ihres Authentifizierungsgeräts, eines Leerzeichens und des darauf angezeigten Authentifizierungscodes. Wenn sie dies im Anforderungs-Header nicht angeben, schlägt die Anforderung fehl.

Weitere Informationen über Authentifizierungsgeräte finden Sie unter https://aws.amazon.com/iam/details/mfa/.

Example Löschen eines Objekts aus einem MFA Delete-fähigen Bucket

Das folgende Beispiel zeigt, wie `my-image.jpg` (in der angegebenen Version) gelöscht wird, ein mit MFA Delete konfigurierter Bucket. Achten Sie auf das Leerzeichen zwischen *`[SerialNumber]`* und *`[AuthenticationCode]`*. Weitere Informationen hierzu finden Sie unter DELETE Object.

```
DELETE /my-image.jpg?versionId=3HL4kqCxf3vjVBH40Nrjfkd HTTPS/1.1
Host: bucketName.s3.amazonaws.com
x-amz-mfa: 20899872 301749
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Weitere Informationen zur Aktivierung von MFA Delete finden Sie unter MFA Delete (p. 516).

Arbeiten mit Löschmarkierungen

Eine Löschmarkierung ist ein Platzhalter (Markierung) für ein versionsfähiges Objekt, das in einer einfachen `DELETE`-Anforderung angegeben wurde. Weil sich das Objekt in einem versionsfähigen Bucket befunden hat, wurde das Objekt nicht gelöscht. Die Löschmarkierung bewirkt jedoch, dass sich Amazon S3 verhält, als wäre es gelöscht worden.

Eine Löschmarkierung hat einen Schlüsselnamen (oder Schlüssel) und eine Versions-ID, so wie jedes andere Objekt. Eine Löschmarkierung unterscheidet sich jedoch wie folgt von anderen Objekten:

- Es sind ihm keine Daten zugeordnet.
- Es ist keinem ACL-Wert (Zugriffskontrollliste) zugeordnet.
- Eine `GET`-Anforderung kann nichts dafür anfordern, weil es keine Daten enthält. Sie erhalten den Fehler 404.
- Die einzige Operation, die Sie für eine Löschmarkierung verwenden können, ist ein Amazon S3-API-`DELETE`-Aufruf. Dazu müssen Sie die `DELETE`-Anforderung unter Verwendung eines AWS Identity and Access Management-(IAM-)Benutzers oder einer -Rolle mit den entsprechenden Berechtigungen durchführen.

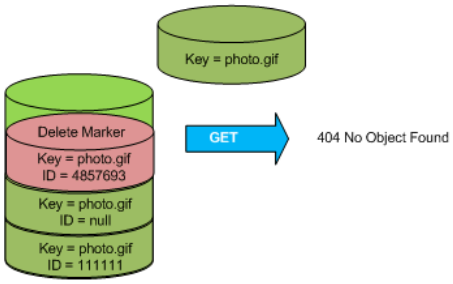
Für Löschmarkierungen fällt eine Gebühr für die Speicherung in Amazon S3 an. Der Speicherbedarf einer Löschmarkierung ist gleich der Größe des Schlüsselnamens der Löschmarkierung. Ein Schlüsselname ist eine Folge von Unicode-Zeichen. Die UTF-8-Codierung fügt Ihrem Bucket für jedes Zeichen im Namen 1 bis 4 Bytes Speicherbedarf hinzu. Weitere Informationen zu Schlüsselnamen finden Sie unter Objektschlüssel (p. 117). Informationen zum Löschen von Löschmarkierungen finden Sie unter Entfernen von Löschmarkierungen (p. 531). .

Nur Amazon S3 kann eine Löschmarkierung erstellen. Dies erfolgt immer, wenn Sie eine `DELETE Object`-Anforderung für ein Objekt in einem Bucket mit Versioning oder ausgesetztem Versioning stellen. Das Objekt in der `DELETE`-Anforderung wird nicht wirklich gelöscht. Stattdessen wird die Löschmarkierung zur aktuellen Version des Objekts. (Der Schlüsselname (oder Schlüssel) des Objekts wird zum Schlüssel der Löschmarkierung.) Wenn Sie versuchen, ein Objekt abzurufen, und seine aktuelle Version eine Löschmarkierung ist, antwortet Amazon S3 mit:

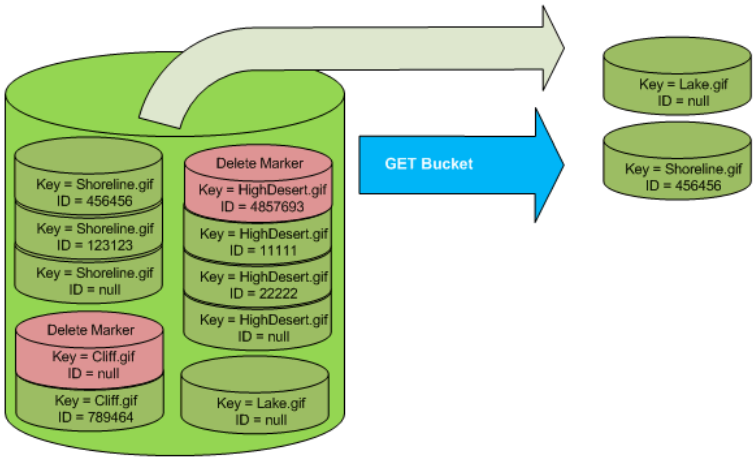
- Einem 404-Fehler (Object not found)
- Einem Antwort-Header, x-amz-delete-marker: true

Der Antwort-Header teilt Ihnen mit, dass das Objekt, auf das Sie zugegriffen haben, eine Löschmarkierung war. Dieser Antwort-Header gibt nie `false` zurück. Wenn der Wert `false` ist, nimmt Amazon S3 diesen Antwort-Header nicht in die Antwort auf.

Die folgende Abbildung zeigt, wie ein einfaches `GET` für ein Objekt, dessen aktuelle Version eine Löschmarkierung ist, den Fehler 404 No Object Found zurückgibt.



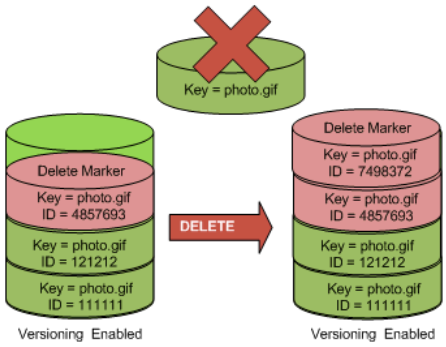
Die einzige Methode, Löschmarkierungen (und andere Versionen eines Objekts) aufzulisten, ist die Verwendung der `versions`-Subressource in einer `GET Bucket versions`-Anforderung. Ein einfaches `GET` ruft keine Löschmarkierungsobjekte ab. Die folgende Abbildung zeigt, dass ein einfaches `GET Bucket` keine Objekte zurückgibt, deren aktuelle Version eine Löschmarkierung ist.



Entfernen von Löschmarkierungen

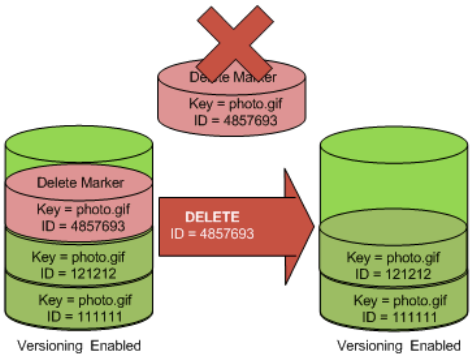
Um eine Löschmarkierung zu löschen, müssen Sie Ihre Versions-ID in einer `DELETE Object versionId`-Anforderung angeben. Wenn Sie eine `DELETE`-Anforderung verwenden, um eine Löschmarkierung zu löschen (ohne die Versions-ID der Löschmarkierung anzugeben), löscht Amazon S3 die Löschmarkierung nicht, sondern fügt stattdessen eine weitere Löschmarkierung ein.

Die folgende Abbildung zeigt, wie ein einfaches `DELETE` für eine Löschmarkierung nichts zurückgibt, sondern eine neue Löschmarkierung in einen Bucket einfügt.



In einem versionsfähigen Bucket hat diese neue Löschmarkierung eine eindeutige Versions-ID. Es ist also möglich, mehrere Löschmarkierungen desselben Objekts in einem Bucket zu haben.

Um eine Löschmarkierung dauerhaft zu löschen, müssen Sie Ihre Versions-ID in einer `DELETE Object versionId`-Anforderung angeben. Die folgende Abbildung zeigt, dass ein einfaches `DELETE Object versionId` eine Löschmarkierung nicht dauerhaft entfernt. Nur der Eigentümer eines Buckets kann eine Löschmarkierung dauerhaft entfernen.



Das Entfernen der Löschmarkierung bewirkt, dass eine einfache `GET`-Anforderung jetzt die aktuelle Version (121212) des Objekts abruft.

Eine Löschmarkierung dauerhaft entfernen

1. Setzen Sie den Parameter `versionId` auf die ID der Version der Löschmarkierung, die Sie entfernen wollen.
2. Senden Sie eine `DELETE Object versionId`-Anforderung.

Example Entfernen einer Löschmarkierung

Das folgende Beispiel entfernt die Löschmarkierung für `photo.gif` Version 4857693.

```
DELETE /photo.gif?versionId=4857693 HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Wenn Sie eine Löschmarkierung löschen, nimmt Amazon S3 folgendes in die Antwort auf:

```
204 NoContent
x-amz-version-id: versionID
x-amz-delete-marker: true
```

Übergang von Objektversionen

Sie können Lebenszykluskonfigurationsregeln für Objekte definieren, die einen definieren Lebenszyklus haben, um Objektversionen zu einem bestimmten Zeitpunkt in der Lebensdauer des Objekts in die `S3 Glacier`-Speicherklasse übergehen zu lassen. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Wiederherstellen früherer Versionen

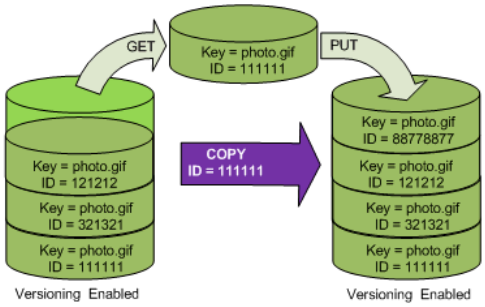
Einer der Vorteile des Versionings ist die Möglichkeit, frühere Versionen eines Objekts wiederherzustellen. Hierfür gibt es zwei Ansätze:

- Kopieren einer vorhergehenden Version des Objekts in denselben Bucket.

  Das kopierte Objekt wird zur aktuellen Version dieses Objekts, und alle Objektversionen werden beibehalten.
- Dauerhaftes Löschen der aktuellen Version des Objekts

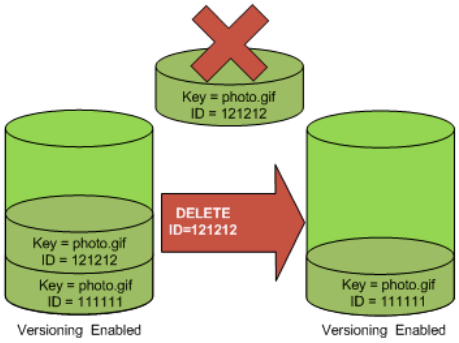
  Wenn Sie die aktuelle Objektversion löschen, wandeln Sie letztlich die nicht vorherige Version in die aktuelle Version dieses Objekts um.

Alle Objektversionen werden aufbewahrt, deshalb können Sie eine frühere Version zur aktuellen Version machen, indem Sie eine spezifische Version des Objekts in denselben Bucket kopieren. In der folgenden Abbildung wird das Quellobjekt (ID = 111111) in denselben Bucket kopiert. Amazon S3 stellt eine neue ID (88778877) bereit, und diese wird zur aktuellen Version des Objekts. Der Bucket enthält jetzt also die ursprüngliche Objektversion (111111) und kopiert seine Kopie (88778877).



Ein nachfolgendes `GET` ruft Version 88778877 ab.

Die folgende Abbildung zeigt, wie die aktuelle Version (121212) eines Objekts gelöscht wird, sodass die vorhergehende Version (111111) zum aktuellen Objekt wird.



Ein nachfolgendes `GET` ruft Version 111111 ab.

Versionierte Objekteberechtigungen

Berechtigungen werden auf Versionsebene eingerichtet. Jede Version hat ihren eigenen Objekt-Eigentümer. Ein AWS-Konto, das die Objektversion erstellt, ist der Eigentümer. Sie können also unterschiedliche Berechtigungen für unterschiedliche Versionen desselben Objekts einrichten. Dazu geben Sie die Versions-ID des Objekts an, dessen Berechtigungen Sie in einer `PUT Object versionId acl`-Anforderung setzen wollen. Eine detaillierte Beschreibung sowie Anweisungen zur Verwendung von ACLs finden Sie unter Identity and Access Management in Amazon S3 (p. 329)

Example Einrichtung von Berechtigungen für eine Objektversion

Die folgende Anforderung setzt die Berechtigung für den Empfänger, `BucketOwner@amazon.com`, auf `FULL_CONTROL` für den Schlüssel, `my-image.jpg`, Versions-ID, 3HL4kqtJvjVBH40Nrjfkd.

```
PUT /my-image.jpg?acl&versionId=3HL4kqtJvjVBH40Nrjfkd HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
Content-Length: 124

<AccessControlPolicy>
  <Owner>
    <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
    <DisplayName>mtd@amazon.com</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
```

```
        <ID>a9a7b886d6fd24a52fe8ca5bef65f89a64e0193f23000e241bf9b1c61be666e9</ID>
        <DisplayName>BucketOwner@amazon.com</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
  </AccessControlPolicy>
```

Analog dazu müssen Sie, um die Berechtigungen für eine spezifische Objektversion zu erhalten, ihre Versions-ID in einer `GET Object versionId acl`-Anforderung angeben. Sie müssen diese Versions-ID angeben, weil `GET Object acl` standardmäßig die Berechtigungen für die aktuelle Version des Objekts zurückgibt.

Example Abrufen der Berechtigungen für eine bestimmte Objektversion

Im folgenden Beispiel gibt Amazon S3 die Berechtigungen für den Schlüssel, `my-image.jpg`, Versions-ID, DVBH40Nr8X8gUMLUo, zurück.

```
GET /my-image.jpg?versionId=DVBH40Nr8X8gUMLUo&acl HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU
```

Weitere Informationen hierzu finden Sie unter GET Object acl.

Verwalten von Objekten in einem Bucket mit ausgesetztem Versioning

Themen

- Hinzufügen von Objekten zu Buckets mit ausgesetztem Versioning (p. 534)
- Abrufen von Objekten aus Buckets mit ausgesetztem Versioning (p. 535)
- Löschen von Objekten aus Buckets mit ausgesetztem Versioning (p. 536)

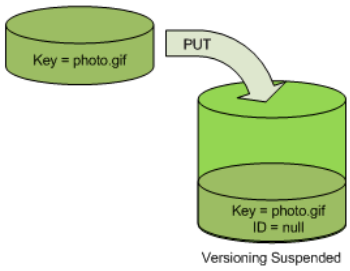
Sie setzen das Versioning aus, um zu verhindern, dass sich neue Versionen desselben Objekts in einem Bucket ansammeln. Das könnte notwendig sein, wenn Sie nur eine Version eines Objekts in einem Bucket wollen,oder wenn Sie nicht wollen, dass Gebühren für mehrere Versionen anfallen.

Wenn Sie das Versioning aussetzen, ändern sich die in Ihrem Bucket vorhandenen Objekte nicht. Was sich ändert, ist, wie Amazon S3 die Objekte in zukünftigen Anforderungen verarbeitet. Die Themen in diesem Abschnitt erklären verschiedene Objektoperationen in einem Bucket mit ausgesetztem Versioning.

Hinzufügen von Objekten zu Buckets mit ausgesetztem Versioning

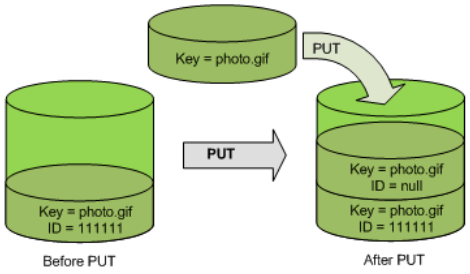
Wenn Sie das Versioning für einen Bucket ausgesetzt haben, fügt Amazon S3 automatisch jedem nachfolgend (mit `PUT`, `POST` oder `COPY`) in diesem Bucket gespeicherten Objekt die Versions-ID `null` hinzu

Die folgende Abbildung zeigt, wie Amazon S3 jedem Objekt automatisch die Versions-ID `null` hinzufügt, wenn es einem Bucket mit ausgesetztem Versioning hinzugefügt wird.

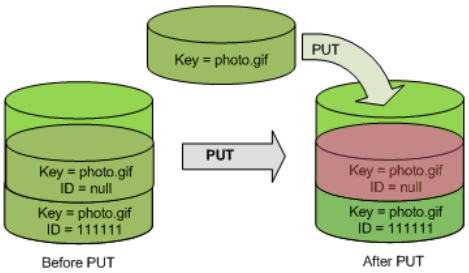


Wenn sich bereits eine Null-Version im Bucket befindet und Sie ein weiteres Objekt mit demselben Schlüssel hinzufügen, überschreibt das hinzugefügte Objekt die ursprüngliche Null-Version.

Wenn es versionsfähige Objekte im Bucket gibt, wird die Version, die Sie mit `PUT` hinzufügen, zur aktuellen Version des Objekts. Die folgende Abbildung zeigt, wie das Hinzufügen eines Objekts in einen Bucket, der versionsfähige Objekte enthält, das bereits im Bucket enthaltene Objekt nicht überschreibt. In diesem Fall befand sich Version 111111 bereits im Bucket. Amazon S3 weist dem Objekt, das hinzugefügt und im Bucket gespeichert wird, die Versions-ID null zu. Version 111111 wird nicht überschrieben.



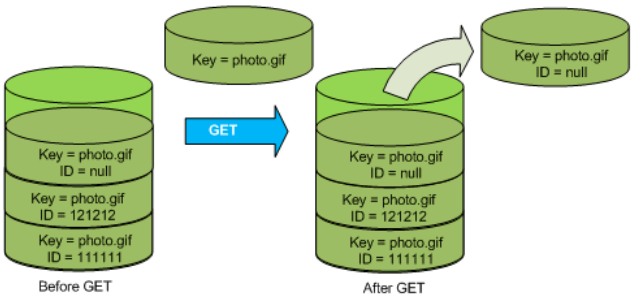
Wenn bereits eine Null-Version in einem Bucket vorhanden ist, wird die Null-Version überschrieben, wie in der folgenden Abbildung gezeigt.



Beachten Sie, dass der Schlüssel und die Versions-ID (`null`) der Null-Version vor und nach dem `PUT` gleich sind, der Inhalt der ursprünglich im Bucket gespeicherten Version jedoch durch den Inhalt des Objekts ersetzt wird, das mit `PUT` in den Bucket geschrieben wurde.

Abrufen von Objekten aus Buckets mit ausgesetztem Versioning

Eine `GET Object`-Anforderung gibt die aktuelle Version eines Objekts zurück, unabhängig davon, ob Sie das Versioning für einen Bucket aktiviert haben oder nicht. Die folgende Abbildung zeigt, wie ein einfaches `GET` die aktuelle Version des Objekts zurückgibt.



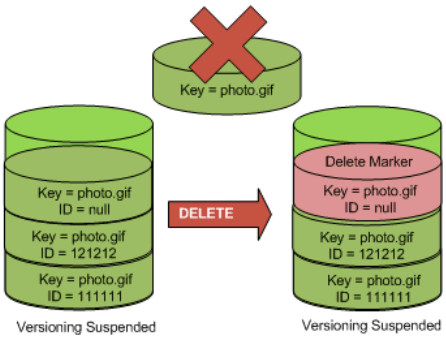
Löschen von Objekten aus Buckets mit ausgesetztem Versioning

Wenn das Versioning ausgesetzt ist, gilt für eine `DELETE`-Anforderung:

- Kann nur ein Objekt entfernen, dessen Versions-ID `null` ist

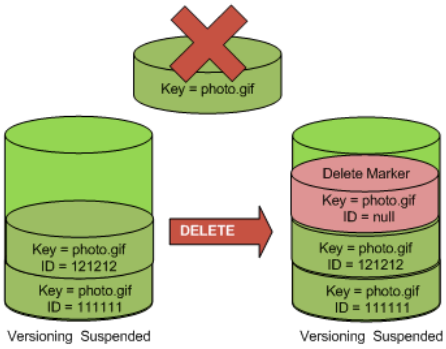
  Entfernt nichts, wenn es keine Nullversion des Objekts im Bucket gibt.
- Fügt eine Löschmarkierung in den Bucket ein.

Die folgende Abbildung zeigt, wie ein einfaches `DELETE` eine Nullversion entfernt und Amazon S3 stattdessen eine Löschmarkierung mit der Versions-ID `null` einfügt.

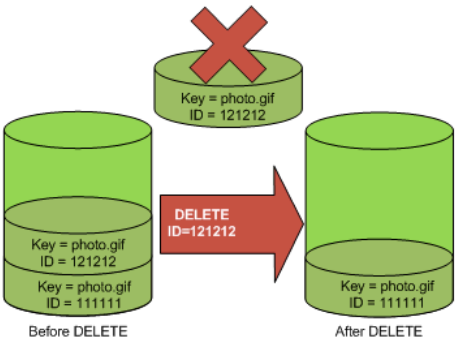


Beachten Sie, dass eine Löschmarkierung keinen Inhalt hat, Sie verlieren also den Inhalt der Nullversion, wenn sie durch eine Löschmarkierung ersetzt wird.

Die folgende Abbildung zeigt einen Bucket, der keine Nullversion enthält. In diesem Fall entfernt `DELETE` nichts, sondern Amazon S3 fügt einfach eine Löschmarkierung ein.



Selbst in einem Bucket mit ausgesetztem Versioning kann der Bucket-Eigentümer eine spezifische Version dauerhaft löschen. Die folgende Abbildung zeigt, dass das beim Löschen einer angegebenen Objektversion dieses Objekt dauerhaft gelöscht wird. Nur der Bucket-Eigentümer kann eine spezifische Objektversion löschen.



Sperren von Objekten mit S3 Objektsperre

Mit S3 Objektsperre können Sie Objekte anhand des Modells Write Once Read Many (WORM) speichern. Sie können damit für einen bestimmten Zeitraum verhindern, dass ein Objekt gelöscht oder überschrieben wird. Objektsperre hilft Ihnen dabei, regulatorische Bestimmungen einzuhalten, die die WORM-Speicherung verlangen, oder einfach nur eine weitere Schutzebene gegen das Ändern und Löschen von Objekten einzurichten.

S3 Objektsperre wurde von Cohasset Associates zur Verwendung in Umgebungen zugelassen, die den Regulierungen SEC 17a-4, CTCC und FINRA unterliegen. Weitere Informationen zu Objektsperre im Zusammenhang mit diesen Bestimmungen finden Sie unter Compliance-Bewertung durch Cohasset Associates.

Objektsperre stellt zwei Optionen für die Verwaltung der Aufbewahrung von Objekten bereit: Aufbewahrungszeiträume und rechtliche Fristen.

- Ein Aufbewahrungszeitraum gibt einen festen Zeitraum an, über den ein Objekt gesperrt bleibt. Während dieses Zeitraums ist Ihr Objekt WORM-geschützt und kann weder überschrieben noch gelöscht werden.
- Eine rechtliche Aufbewahrungsfrist bietet denselben Schutz wie ein Aufbewahrungszeitraum, hat jedoch kein Ablaufdatum. Stattdessen bleibt eine rechtliche Aufbewahrungsfrist gültig, bis Sie diese ausdrücklich entfernen. Rechtliche Aufbewahrungsfristen sind von Aufbewahrungszeiträumen unabhängig.

Für eine Objektversion kann es sowohl einen Aufbewahrungszeitraum als auch eine rechtliche Aufbewahrungsfrist geben, eins von beiden oder keins von beiden. Weitere Informationen finden Sie unter Übersicht über S3 Objektsperre (p. 538).

Objektsperre funktioniert nur in versionsgesteuerten Buckets. Aufbewahrungszeiträume und rechtliche Aufbewahrungsfristen gelten für einzelne Objektversionen. Wenn Sie eine Objektversion sperren, speichert Amazon S3 die Sperrinformationen in den Metadaten für diese Objektversion. Die Platzierung eines Aufbewahrungszeitraums oder einer rechtlichen Aufbewahrungsfrist für ein Objekt schützt ausschließlich die in der Anfrage angegebene Version. Sie verhindert nicht die Erstellung neuer Versionen des Objekts. Wenn Sie ein Objekt in einem Bucket platzieren, der denselben Schlüsselnamen wie ein vorhandenes, geschütztes Objekt hat, erstellt Amazon S3 eine neue Version dieses Objekts, speichert es im Bucket wie angefragt und meldet die Anfrage als erfolgreich abgeschlossen. Die vorhandene, geschützte Version des Objekts bleibt entsprechend seiner Aufbewahrungskonfiguration gesperrt.

Befolgen Sie zur Verwendung von S3 Objektsperre diese grundlegenden Schritte:

1. Erstellen Sie einen neuen Bucket, für den Objektsperre aktiviert ist.
2. (Optional) Konfigurieren Sie einen Standardaufbewahrungszeitraum für Objekte, die in diesem Bucket platziert werden.
3. Platzieren Sie die Objekte, die Sie sperren möchten, im Bucket.

4. Wenden Sie auf die Objekte, die Sie schützen möchten, einen Aufbewahrungszeitraum, eine rechtliche Aufbewahrungsfrist oder beides an.

Informationen zur Verwendung von Objektsperre auf dem AWS Management Console finden Sie unter Wie sperre ich ein Amazon S3-Objekt? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Themen

Übersicht über S3 Objektsperre

Mit S3 Objektsperre können Sie Objekte anhand des Modells Write Once Read Many (WORM) speichern. Dadurch können Sie für einen festen Zeitraum oder unbegrenzt verhindern, dass Objekte gelöscht oder überschrieben werden. Sie können S3 Objektsperre verwenden, um regulatorische Anforderungen einzuhalten, die die WORM-Speicherung verlangen, oder um eine zusätzliche Schutzebene gegen Objektänderungen und -löschungen einzurichten.

Informationen zur Verwaltung des Sperrungsstatus Ihrer Amazon S3-Objekte finden Sie unter the section called “Verwalten von Objektsperren” (p. 541).

> **Note**
>
> S3-Buckets mit S3 Objektsperre können nicht als Ziel-Buckets für Server access logging (Server-Zugriffsprotokollierung) (p. 777) verwendet werden

Die folgenden Abschnitte beschreiben die Hauptfunktionen von S3 Objektsperre.

Themen

Aufbewahrungsmodi

S3 Objektsperre bietet zwei Aufbewahrungsmodi:

- Governance-Modus
- Compliance-Modus

Diese Aufbewahrungsmodi wenden unterschiedliche Schutzgrade auf Ihre Objekte an. Sie können auf jede von Objektsperre geschützte Objektversion einen der beiden Aufbewahrungsmodi anwenden.

Im Governance-Modus können Benutzer eine Objektversion nicht überschreiben oder löschen oder ihre Sperreinstellungen ändern, wenn sie keine speziellen Berechtigungen besitzen. Mit dem Governance-Modus schützen Sie Objekte dagegen, von den meisten Benutzern gelöscht zu werden, Sie können jedoch weiterhin einigen Benutzern die Berechtigung geben, die Aufbewahrungseinstellungen zu ändern oder das Objekt bei Bedarf zu löschen. Sie können den Governance-Modus auch verwenden, um die Einstellungen für Aufbewahrungszeiträume zu testen, bevor Sie einen Aufbewahrungszeitraum im Compliance-Modus erstellen. Um Aufbewahrungseinstellungen im Governance-Modus zu überschreiben oder zu entfernen, muss ein Benutzer die Berechtigung `s3:BypassGovernanceRetention` besitzen und in jede Anfrage,

die ein Überschreiben des Governance-Modus erfordert, ausdrücklich `x-amz-bypass-governance-retention:true` als Anfrage-Header einfügen.

Note

Die Amazon S3-Konsole enthält standardmäßig den Header`x-amz-bypass-governance-retention:true`. Ein Löschen von Objekten, die durch den Governance-Modus geschützt sind und über die Berechtigungen`s3:BypassGovernanceRetention` oder `s3:GetBucketObjectLockConfiguration` verfügen, kann durchgeführt werden.

Im Compliance-Modus kann eine geschützte Objektversion von keinem Benutzer überschrieben oder gelöscht werden. Dies schließt den Root-Benutzer in Ihrem AWS-Konto ein. Wenn ein Objekt im Compliance-Modus gesperrt wurde, können der Aufbewahrungsmodus nicht geändert und der Aufbewahrungszeitraum nicht verkürzt werden. Der Compliance-Modus stellt sicher, dass eine Objektversion während des Aufbewahrungszeitraums weder überschrieben noch gelöscht werden.

Note

Die Aktualisierung der Metadaten einer Objektversion, wenn Sie eine Objektsperre platzieren oder ändern, überschreibt weder die Objektversion noch setzt sie den Zeitstempel für `Last-Modified` zurück.

Aufbewahrungszeiträume

Ein Aufbewahrungszeitraum schützt eine Objektversion für einen festen Zeitraum. Wenn Sie für eine Objektversion einen Aufbewahrungszeitraum festlegen, speichert Amazon S3 einen Zeitstempel in den Metadaten der Objektversion, um anzugeben, wann der Aufbewahrungszeitraum abläuft. Nach Ablauf des Aufbewahrungszeitraums kann die Objektversion überschrieben oder gelöscht werden, vorausgesetzt, Sie haben für die Objektversion nicht auch eine rechtliche Aufbewahrungsfrist festgelegt.

Ein Aufbewahrungszeitraum kann ausdrücklich oder über eine Bucket-Standardeinstellung für ein Objekt festgelegt werden. Wenn Sie einen Aufbewahrungszeitraum ausdrücklich auf eine Objektversion anwenden, geben Sie ein Ablaufdatum für die Aufbewahrung an. Amazon S3 speichert dieses Datum in den Metadaten der Objektversion und schützt die Objektversion, bis der Aufbewahrungszeitraum abläuft.

Wenn Sie Bucket-Standardeinstellungen verwenden, geben Sie kein Ablaufdatum für die Aufbewahrung an. Sie geben stattdessen eine Dauer in Tagen oder Jahren an, für die jede im Bucket platzierte Objektversion geschützt werden soll. Wenn Sie ein Objekt in den Bucket legen, berechnet Amazon S3 ein „Aufbewahren bis“-Datum für die Objektversion, indem es die angegebene Dauer dem Zeitstempel der Erstellung der Objektversion hinzufügt. Dieses Datum wird in den metadaten der Objektversion gespeichert. Die Objektversion wird anschließend so geschützt, als ob Sie für die Objektversion ausdrücklich eine Sperre mit diesem Aufbewahrungszeitraum angegeben hätten.

Note

Wenn Ihre Anfrage für die Platzierung einer Objektversion in einem Bucket ausdrücklich einen Aufbewahrungsmodus und -zeitraum angibt, überschreiben diese Einstellungen alle Bucket-Standardeinstellungen für diese Objektversion.

Wie alle anderen Objektsperre-Einstellungen gelten Aufbewahrungszeiträume für einzelne Objektversionen. Für verschiedene Versionen desselben Objekts können verschiedene Aufbewahrungsmodi und -zeiträume gelten.

Zum Beispiel: Angenommen, Sie haben ein Objekt, von dessen 30-tägigem Aufbewahrungszeitraum 15 Tage abgelaufen sind, und Sie führen die Aktion `PUT` zu Amazon S3 für ein Objekt mit demselben Namen und einem 60-tägigen Aufbewahrungszeitraum durch. In diesem Fall ist die `PUT`-Aktion erfolgreich, und Amazon S3 erstellt eine neue Version des Objekts mit einem 60-tägigen Aufbewahrungszeitraum. Die ältere Version behält den ursprünglichen Aufbewahrungszeitraum und kann in 15 Tagen gelöscht werden.

Sie können einen Aufbewahrungszeitraum verlängern, nachdem Sie eine Aufbewahrungseinstellung auf eine Objektversion angewendet haben. Hierzu senden Sie eine neue Sperranfrage für die Objektversion

ab, die ein `Retain Until Date` für die Aufbewahrung enthält, das nach dem aktuell für die Objektversion konfigurierten Ablaufdatum liegt. Amazon S3 ersetzt den vorhandenen Aufbewahrungszeitraum durch den neuen, längeren Zeitraum. Alle Benutzer, die die Berechtigung zur Festlegung von Aufbewahrungszeiträumen für Objekte besitzen, können Aufbewahrungszeiträume für Objektversionen verlängern, die in einem der beiden Modi gesperrt wurden.

Rechtliche Aufbewahrungsfristen

Objektsperre ermöglicht Ihnen auch die Festlegung einer rechtlichen Aufbewahrungsfrist für eine Objektversion. Wie Aufbewahrungszeiträume verhindern auch rechtliche Aufbewahrungsfristen das Überschreiben oder Löschen von Objektversionen. Mit rechtlichen Aufbewahrungsfristen sind jedoch keine Aufbewahrungszeiträume verknüpft. Sie sind gültig, bis sie entfernt werden. Alle Benutzer mit der Berechtigung `s3:PutObjectLegalHold` können rechtliche Aufbewahrungsfristen festlegen und entfernen. Die vollständige Liste der Amazon S3-Berechtigungen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Rechtliche Aufbewahrungsfristen sind von Aufbewahrungszeiträumen unabhängig. So lange für den Bucket, der das Objekt enthält, Objektsperre aktiviert ist, können Sie rechtliche Aufbewahrungsfristen festlegen und entfernen. Dies gilt unabhängig davon, ob für die angegebene Objektversion ein Aufbewahrungszeitraum festgelegt wurde. Die Festlegung einer rechtlichen Aufbewahrungsfrist für eine Objektversion hat keine Auswirkungen auf den Aufbewahrungsmodus oder -zeitraum für die betreffende Objektversion. Zum Beispiel: Angenommen, Sie setzen eine rechtliche Aufbewahrungsfrist für eine Objektversion fest, während diese Objektversion gleichzeitig durch einen Aufbewahrungszeitraum geschützt it. Wenn der Aufbewahrungszeitaum abgläuft, verliert das Objekt seinen WORM-Schutz nicht. Stattdessen wird das Objekt durch die rechtliche Aufbewahrungsfrist weiter geschützt, bis ein autorisierter Benutzer diese ausdrücklich entfernt. Wenn Sie eine rechtliche Aufbewahrungsfrist entfernen, während für eine Objektversion ein Aufbewahrungszeitraum gilt, bleibt die Objektversion geschützt, bis der Aufbewahrungszreitraum abläuft.

Bucket-Konfiguration

Zur Verwendung von Objektsperre muss dies für einen Bucket aktiviert werden. Sie können optional auch einen Standardaufbewahrungsmodus und -zeitraum konfigurieren, der für neue Objekte gilt, die in den Bucket platziert werden.

Aktivieren von S3 Objektsperre

Damit Sie Objekte sperren können, müssen Sie einen Bucket für die Verwendung von S3 Objektsperre konfigurieren. Dazu geben Sie an, wann Sie den Bucket erstellen, der Objektsperre aktivieren soll. Wenn Sie einen Bucket für Objektsperre konfiguriert haben, können Sie Objekte in diesem Bucket durch Aufbewahrungszeiträume, rechtliche Aufbewahrungsfristen oder beides sperren.

Note

- Sie können Objektsperre ausschließlich für neue Buckets aktivieren. Wenn Sie Objektsperre für einen bereits bestehenden Bucket aktivieren möchten, wenden Sie sich an AWS Support.
- Wenn Sie einen Bucket mit aktiviertem Objektsperre erstellen, aktiviert Amazon S3 automatisch die Versionssteuerung für diesen Bucket.
- Wenn Sie einen Bucket mit aktivierter Objektsperre erstellen, können Sie Objektsperre nicht deaktivieren oder die Versionssteuerung für diesen Bucket aussetzen.

Informationen zur Aktivierung von Objektsperre auf der Konsole finden Sie unter Wie sperre ich ein Amazon S3-Objekt? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service

Standardaufbewahrungseinstellungen

Wenn Sie Objektsperre für einen Bucket aktivieren, kann der Bucket geschützte Objekte speichern. Die Einstellung schützt jedoch nicht automatisch Objekte, die in den Bucket gelegt werden. Wenn Sie

die Objekte, die Sie in diesen Bucket platzieren, automatisch schützen möchten, können Sie einen Standardaufbewahrungszeitraum konfigurieren. Die Standardeinstellungen gelten für alle neuen Objekte, die im Bucket platziert werden, wenn Sie nicht ausdrücklich während der Erstellung eines Objekts einen anderen Aufbewahrungsmodus und -zeitraum für dieses Objekt angeben.

Tip

Wenn Sie den Standaufbewahrungsmodus und -zeitraum für alle neuen Objektversionen durchsetzen möchten, die in einem Bucket platziert werden, können Sie die Bucket-Standardeinstellungen festlegen und Benutzern die Berechtigung zum Festlegen der Aufbewahrungseinstellungen von Objekten verweigern. Amazon S3 wendet anschließend den Standardaufbewahrungsmodus und -zeitraum auf neue Objektversionen an, die im Bucket platziert werden, und weist alle Anfragen für das Platzieren von Objekten zurück, die einen Aufbewahrungsmodus und -zeitraum enthalten.

In den Bucket-Standardeinstellungen müssen ein Modus und ein Zeitraum festgelegt werden. Der Bucket-Standardmodus kann entweder Governance oder Compliance sein. Weitere Informationen finden Sie unter Aufbewahrungsmodi (p. 538).

Ein Standardaufbewahrungszeitraum wird nicht als Zeitstempel, sondern als Zeitraum von Tagen oder Jahren beschrieben. Wenn Sie ein Objekt in einen Bucket mit einem Standard-Aufbewahrungszeitraum legen, berechnet Objektsperre einen Aufbewahren bis-Zeitraum. Dazu wird der Standard-Aufbewahrungszeitraum dem Zeitstempel des Erstellungszeitpunkts der Objektversion hinzugefügt. Amazon S3 speichert den resultierenden Zeitstempel als das „Aufbewahren bis“-Datum der Objektversion so, als hätten Sie ihn manuell berechnet und selbst auf die Objektversion gesetzt.

Standardeinstellungen gelten nur für neu im Bucket platzierte Objekte. Durch die Festlegung von Standardaufbewahrungseinstellungen für einen Bucket werden keine Aufbewahrungseinstellungen für Objekte festgelegt, die im Bucket bereits vorhanden sind.

Important

Objektsperren gelten ausschließlich für einzelne Objektversionen. Wenn Sie ein Objekt in einem Bucket platzieren, für den ein Standardaufbewahrungszeitraum festgelegt wurde, und für dieses Objekt nicht ausdrücklich einen Aufbewahrungszeitraum angeben, erstellt Amazon S3 das Objekt mit dem Standardaufbewahrungszeitraum für den Bucket. Nach der Erstellung des Objekts ist dessen Aufbewahrungszeitraum vom Standardaufbewahrungszeitraum des Buckets unabhängig. Die Änderung des Standardaufbewahrungszeitraums für einen Bucket hat keine Auswirkungen auf den vorhandenen Aufbewahrungszeitraum für Objekte in diesem Bucket.

Note

Wenn Sie einen Standardaufbewahrungszeitraum für einen Bucket konfigurieren, müssen Anforderungen zum Hochladen von Objekten in einen solchen Bucket den `Content-MD5`-Header enthalten. Weitere Informationen finden Sie unter PUT Object im Amazon Simple Storage Service API Reference.

Erforderliche Berechtigungen

Objektsperre-Operationen erfordern bestimmte Berechtigungen. Weitere Informationen zu erforderlichen Berechtigungen finden Sie unter Beispiel – Objektoperationen (p. 380). Informationen zur Verwendung von Bedingungen mit Berechtigungen finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382).

Verwalten von Amazon S3-Objektsperren

Mit S3 Objektsperre können Sie Objekte in Amazon S3 mit dem Modell Write Once Read Many (WORM) speichern. Sie können damit den Objektsperrungsstatus Ihrer Amazon S3-Objekte anzeigen, konfigurieren

und verwalten. Weitere Informationen zu S3 Objektsperre-Funktionen finden Sie unter Übersicht über S3 Objektsperre (p. 538).

Themen

Anzeigen der Sperrinformationen für ein Objekt

Sie können den Objektsperre-Status einer Amazon S3-Objektversion mithilfe der Befehle `GET Object` und `HEAD Object` anzeigen. Beide Befehle geben den Aufbewahrungsmodus, `Retain Until Date`, und den Status in Bezug auf eine rechtliche Aufbewahrungsfrist für die angegebene Objektversion an.

Um Aufbewahrungsmodus und Aufbewahrungszeitraum für eine Objektversion anzuzeigen, müssen Sie die Berechtigung `s3:GetObjectRetention` besitzen. Um den Status eines Objekts in Bezug auf rechtliche Aufbewahrungsfristen anzuzeigen, müssen Sie die Berechtigung `s3:GetObjectLegalHold` besitzen. Für die Aktionen `GET` oder `HEAD` für eine Objektversion ohne die erforderlichen Berechtigungen zum Anzeigen des Sperrungsstatus ist die Anforderung erfolgreich. Es werden jedoch keine Informationen zurückgegeben, für deren Anzeige Sie nicht berechtigt sind.

Sie können die Standardaufbewahrungskonfiguration eines Buckets (wenn vorhanden) anzeigen, indem Sie die Objektsperre-Konfiguration des Buckets anfordern. Dazu müssen Sie über die `s3:GetBucketObjectLockConfiguration`-Berechtigung verfügen. Wenn Sie eine Objektsperre-Konfiguration für einen Bucket anfordern, für den kein S3 Objektsperre aktiviert ist, gibt Amazon S3 einen Fehler zurück. Weitere Informationen zu Berechtigungen finden Sie unter Beispiel – Objektoperationen (p. 380).

Sie können Amazon S3-Bestandsberichte für Ihre Buckets so konfigurieren, dass sie `Retain Until Date`, `object lock Mode` und `Legal Hold Status` für alle Objekte in einem Bucket enthalten. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503).

Umgehen des Governance-Modus

Sie können für im Governance-Modus gesperrte Objektversionen, Operationen ausführen, als ob sie nicht geschützt wären, wenn Sie die Berechtigung `s3:BypassGovernanceRetention` besitzen. Zu diesen Operationen gehören das Löschen einer Objektversion, das Verkürzen des Aufbewahrungszeitraums oder das Entfernen von Objektsperre durch die Platzierung einer neuen Sperre mit leeren Parametern. Um den Governance-Modus umgehen zu können, müssen Sie in Ihrer Anforderung ausdrücklich angeben, dass Sie den Governance-Modus umgehen möchten. Sie tun dies, indem Sie den Header `x-amz-bypass-governance-retention:true` in Ihre Anforderung einfügen oder für Anforderungen, die über die SDKs AWS CLI oder AWS ausgeführt werden, den entsprechenden Parameter verwenden. Die AWS Management Console übernimmt diesen Header automatisch für über die Konsole gestellte Anforderungen, wenn Sie über die erforderliche Berechtigung verfügen, den Governance-Modus zu umgehen.

> Note
>
> Die Umgehung des Governance-Modus hat keine Auswirkungen auf den Status einer Objektversion in Bezug auf rechtliche Aufbewahrungsfristen. Wenn für eine Objektversion eine rechtliche Aufbewahrungsfrist aktiviert ist, bleibt die rechtliche Aufbewahrungsfrist in Kraft und verhindert, dass die Objektversion durch Anforderungen überschrieben oder gelöscht wird.

Konfigurieren von Ereignissen und Benachrichtigungen

Amazon S3-Ereignisse können in einem S3-Bucket, für den S3 Objektsperre aktiviert wurde, für Operationen auf Objektebene konfiguriert werden. Wenn `PUT Object`-, `HEAD Object`- und `GET Object`-Aufrufe Objektsperre-Metadaten enthalten, enthalten Ereignisse für diese Aufrufe diese Metadatenwerte. Wenn Objektsperre-Metadaten einem Objekt hinzugefügt werden oder für ein Objekt aktualisiert werden, lösen diese Aktionen außerdem Ereignisse aus. Diese Ereignisse treten ein, wenn Sie `PUT`- oder `GET`-Anforderungen für Informationen zur Objektaufbewahrung oder rechtlichen Aufbewahrungsfristen ausführen.

Weitere Informationen über Amazon S3-Ereignisse finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).

Sie können mit Amazon S3-Ereignisbenachrichtigungen Zugriffe auf Ihre Objektsperre-Konfigurationen und -daten sowie Änderungen unter Verwendung von AWS CloudTrail nachverfolgen. Weitere Informationen über CloudTrail finden Sie in der AWS CloudTrail-Dokumentation.

Sie können auch Amazon CloudWatch verwenden, um auf der Basis dieser Daten Alarme zu generieren. Weitere Informationen über CloudWatch finden Sie in der Amazon CloudWatch-Dokumentation.

Festlegen von Aufbewahrungslimits

Sie können mittels einer Bucket-Richtlinie mindestens erforderliche und maximal zulässige Aufbewahrungszeiträume für einen Bucket festlegen. Hierzu verwenden Sie den Bedingungsschlüssel `s3:object-lock-remaining-retention-days`. Das folgende Beispiel zeigt eine Bucket-Richtlinie, die den Bedingungsschlüssel `s3:object-lock-remaining-retention-days` verwendet, um einen maximalen Aufbewahrungszeitraum von 10 Tagen festzulegen.

```
{
    "Version": "2012-10-17",
    "Id": "<Policy1436912751980>",
    "Statement": [
        {
            "Sid": "<Stmt1436912698057>",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:PutObjectRetention"
            ],
            "Resource": "arn:aws:s3:::<example-bucket>/*",
            "Condition": {
                "NumericGreaterThan": {
                    "s3:object-lock-remaining-retention-days": "10"
                }
            }
        }
    ]
}
```

> **Note**
>
> Wenn es sich bei Ihrem Bucket um den Ziel-Bucket einer Replikationsrichtlinie handelt und Sie minimal und maximal zulässige Aufbewahrungszeiträume für Objektreplikate einrichten möchten, die mittels Replikation erstellt werden, müssen Sie in Ihre Bucket-Richtlinie die Aktion `s3:ReplicateObject` einschließen.

Weitere Informationen finden Sie unter den folgenden Themen:

- Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394)
- Beispiel – Objektoperationen (p. 380)

- Amazon S3-Bedingungsschlüssel (p. 382)

Verwalten von Löschmarkierungen und Objektlebenszyklen

Sie können eine geschützte Objektversion zwar nicht löschen, aber eine Löschmarkierung für das betreffende Objekt erstellen. Durch das Setzen einer Löschmarkierung auf einem Objekt wird keine Objektversion gelöscht. Amazon S3 verhält sich dadurch aber zumeist so, als ob das Objekt gelöscht worden wäre. Weitere Informationen finden Sie unter Arbeiten mit Löschmarkierungen (p. 529).

> Note
>
> Löschmarkierungen sind nicht WORM-geschützt, unabhängig vom Aufbewahrungszeitraum oder der rechtlichen Aufbewahrungsfrist für das zugrunde liegende Objekt.

Konfigurationen für die Verwaltung des Objektlebenszyklus funktionieren für geschützte Objekte weiterhin normal. Dies schließt die Platzierung von Löschmarkierungen ein. Geschützte Objektversionen werden jedoch davor geschützt, durch eine Lebenszykluskonfiguration gelöscht oder überschrieben zu werden. Weitere Informationen zur Verwaltung von Objektlebenszyklen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Verwenden von S3 Objektsperre mit Replikation

Sie können S3 Objektsperre mit der Replikation verwenden, um automatisches, asynchrones Kopieren von gesperrten Objekten und deren Aufbewahrungsmetadaten über S3-Buckets hinweg in derselben oder in verschiedenen AWS-Regionen zu aktivieren. Wenn Sie die Replikation verwenden, werden Objekte aus einem Quell-Bucket in einen Ziel-Bucket repliziert. Weitere Informationen finden Sie unter Replikation (p. 669).

Zur Einrichtung von S3 Objektsperre mit der Replikation können Sie eine der folgenden Optionen auswählen.

Option 1: Aktivieren Sie Objektsperre zuerst.

1. Aktivieren Sie Objektsperre für den Ziel-Bucket oder für den Quell- und Ziel-Bucket.
2. Richten Sie die Replikation zwischen den Quell- und Ziel-Buckets ein.

Option 2: Richten Sie zuerst die Replikation ein.

1. Richten Sie die Replikation zwischen den Quell- und Ziel-Buckets ein.
2. Aktivieren Sie Objektsperre für den Ziel-Bucket oder für den Quell- und Ziel-Bucket.

Zum Abschließen von Schritt 2 in den vorhergehenden Optionen müssen Sie sich an den AWS Support wenden. Dies ist erforderlich, damit die Replikation ordnungsgemäß konfiguriert wird.

Bevor Sie sich an den AWS Support wenden, sehen Sie sich die folgenden Anforderungen zur Einrichtung von Objektsperre mit einer Replikation an:

- Für den Amazon S3-Ziel-Bucket muss Objektsperre aktiviert sein.
- Sie müssen zwei neue Berechtigungen für den S3-Quell-Bucket in der AWS Identity and Access Management (IAM)-Rolle erteilen, mit der Sie die Replikation eingerichtet haben. Die zwei neuen Berechtigungen lauten `s3:GetObjectRetention` und `s3:GetObjectLegalHold`. Wenn die Rolle über eine `s3:Get*`-Berechtigung verfügt, ist die Anforderung erfüllt. Weitere Informationen finden Sie unter Einrichten von Berechtigungen für die Replikation (p. 684).

Mehr über S3 Objektsperre erfahren Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).

Infrastruktursicherheit in Amazon S3

Als verwalteter Service ist Amazon S3 durch die globalen Verfahren zur Gewährleistung der Netzwerksicherheit von AWS geschützt, die im Whitepaper Amazon Web Services: Übersicht über die Sicherheitsprozesse beschrieben sind.

Der Zugriff auf Amazon S3 über das Netzwerk erfolgt durch veröffentlichte AWS-APIS. Clients müssen Transport Layer Security (TLS) 1.0 unterstützen. Wir empfehlen TLS 1.2. Clients müssen außerdem Cipher Suites mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Außerdem müssen Anfragen mit der AWS Signature V4 oder der AWS Signature V2 unterzeichnet werden, wobei diese gültige Anmeldeinformationen erfordern.

Diese APIs lassen sich von einem beliebigen Netzwerkstandort aus aufrufen. Da Amazon S3 jedoch ressourcenbasierte Zugriffsrichtlinien unterstützt, kann es zu Einschränkungen bezüglich der Quell-IP-Adresse kommen. Sie können den Zugriff auf Buckets von bestimmten Amazon Virtual Private Cloud (Amazon VPC)-Endpunkten oder bestimmten VPCs über Amazon S3-Bucket-Richtlinien steuern. So wird der Netzwerkzugriff auf einen angegebenen Amazon S3-Bucket von ausschließlich der spezifischen VPC innerhalb des AWS-Netzwerks effizient isoliert. Weitere Informationen finden Sie unter Beispiel für Bucket-Richtlinien für VPC-Endpunkte für Amazon S3 (p. 454).

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls die Sicherheit der Infrastruktur von Amazon S3:

- Erwägen Sie VPC-Endpunkte für den Amazon S3-Zugriff
- Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets

Konfiguration und Schwachstellenanalyse in Amazon S3

AWS übernimmt grundlegende Sicherheitsaufgaben wie Betriebssystemsicherheit und Datenbank-Patching, Firewall-Konfiguration und Notfallwiederherstellung. Diese Verfahren wurden von qualifizierten Dritten überprüft und zertifiziert. Weitere Informationen finden Sie in den folgenden Ressourcen:

- Compliance-Validierung für Amazon S3 (p. 503)
- Modell der übergreifenden Verantwortlichkeit
- Amazon Web Services: Übersicht über Sicherheitsverfahren (Whitepaper)

Die folgenden bewährten Sicherheitsmethoden umfassen ebenfalls die Adresskonfiguration und Schwachstellenanalyse in Amazon S3:

- Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets
- Aktivieren von AWS Config

Bewährte Sicherheitsmethoden für Amazon S3

Amazon S3 bietet eine Reihe von Sicherheitsfunktionen, die Sie beim Entwickeln und Implementieren Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden sind allgemeine Richtlinien und keine vollständige Sicherheitslösung. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

Themen

Bewährte Methoden für vorbeugende Amazon S3-Sicherheitsmaßnahmen

Die folgenden bewährten Methoden für Amazon S3 können dabei helfen, Sicherheitsvorfälle zu verhindern.

Achten Sie darauf, dass für Ihre Amazon S3-Buckets die korrekten Richtlinien gelten und dass sie nicht öffentlich zugänglich sind.

Sofern es nicht unbedingt notwendig ist, dass jemand Ihren S3-Bucket im Internet lesen oder darin schreiben kann, sollten Sie sicherstellen, dass der S3-Bucket nicht öffentlich ist. Im Folgenden werden einige Maßnahmen aufgeführt:

- Blockieren des öffentlichen Zugriffs in Amazon S3 Mit Amazon S3 Block Public Access können Administratoren und Bucket-Eigentümer problemlos zentrale Kontrollen zur Begrenzung des öffentlichen Zugriffs auf ihre Amazon S3-Ressourcen einrichten. Diese Kontrollen werden unabhängig davon durchgesetzt, wie die Ressourcen erstellt wurden. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493).
- Ermitteln Sie, welche Amazon S3-Bucket-Richtlinien eine Platzhalter-ID wie den Prinzipal „*“ erlauben (was so viel wie „jeder“ bedeutet) oder eine Platzhalteraktion „*“ zulassen (was es einem Benutzer ermöglicht, jede beliebige Aktion im Amazon S3-Bucket durchzuführen).
- Achten Sie ebenfalls auf Zugriffskontrolllisten (ACLs, Access Control Lists) für Amazon S3-Buckets, die „Jedem“ oder „Jedem authentifizierten AWS-Benutzer“ Lese-, Schreib- oder Vollzugriffsberechtigungen gewähren.
- Verwenden Sie die `ListBuckets`-API, um alle Ihre Amazon S3-Buckets zu scannen. Ermitteln Sie dann mit `GetBucketAcl`, `GetBucketWebsite` und `GetBucketPolicy`, ob der Bucket über richtlinienkonforme Zugriffskontrollen und Konfiguration verfügt.
- Verwenden Sie AWS Trusted Advisor, um Ihre Amazon S3-Implementierung zu prüfen.
- Sie sollten auch mithilfe der verwalteten AWS Config-Regeln-Regeln s3-bucket-public-read-prohibited und s3-bucket-public-write-prohibited laufende detektive Kontrollen implementieren.

Weitere Informationen finden Sie unter Festlegen von Zugriffsberechtigungen für Buckets und Objekte im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Implementieren der geringstmöglichen Zugriffsrechte

Beim Erteilen von Berechtigungen entscheiden Sie, wer welche Berechtigungen für welche Amazon S3-Ressourcen erhält. Sie aktivieren die spezifischen Aktionen, die daraufhin für die betreffenden Ressourcen erlaubt sein sollen. Aus diesem Grund sollten Sie nur Berechtigungen gewähren, die zum Ausführen einer Aufgabe erforderlich sind. Die Implementierung der geringstmöglichen Zugriffsrechte ist eine grundlegende Voraussetzung zum Reduzieren des Sicherheitsrisikos und der Auswirkungen, die aufgrund von Fehlern oder böswilligen Absichten entstehen könnten.

Die folgenden Tools stehen zur Implementierung der geringstmöglichen Zugriffsrechte zur Verfügung:

- IAM-Benutzerrichtlinien und Berechtigungsgrenzen für IAM-Entities
- Amazon S3-Bucket-Richtlinien
- Amazon S3-Zugriffskontrolllisten (ACLs)
- Service-Kontrollrichtlinien

Hilfreiche Informationen zur Entscheidung, welche der vorherigen Mechanismen Sie auswählen sollten, finden Sie unter Einführung in das Verwalten des Zugriffs auf Amazon S3-Ressourcen (p. 330).

Verwenden Sie IAM-Rollen für Anwendungen und AWS-Services, die Amazon S3-Zugriff benötigen

Für Anwendungen auf Amazon EC2 oder andere AWS-Services zum Zugriff auf Amazon S3 - Ressourcen müssen gültige AWS-Anmeldeinformationen in den AWS-API-Anforderungen enthalten sein. Sie sollten AWS-Anmeldeinformationen nicht direkt in der Anwendung oder einer Amazon EC2-Instance speichern. Dabei handelt es sich um langfristige Anmeldeinformationen, die nicht automatisch rotiert werden und bedeutende geschäftliche Auswirkungen haben könnten, wenn sie kompromittiert werden.

Stattdessen sollten Sie mithilfe einer IAM-Rolle temporäre Anmeldeinformationen für Anwendungen und Services verwalten, die Zugriff auf Amazon S3 benötigen. Wenn Sie eine Rolle verwenden, müssen Sie keine langfristigen Anmeldeinformationen (wie Benutzername und Passwort oder Zugriffsschlüssel) an eine Amazon EC2-Instance oder einen AWS-Service wie AWS Lambda ausgeben. Stattdessen stellt die Rolle temporäre Berechtigungen bereit, die von den Anwendungen beim Aufrufen anderer AWS-Ressourcen genutzt werden können.

Weitere Informationen finden Sie unter den folgenden Themen im IAM-Benutzerhandbuch:

- IAM-Rollen
- Gängige Szenarien für Rollen: Benutzer, Anwendungen und Services

Aktivieren der Multifaktor-Authentifizierung (MFA)-Löschfunktion

Die MFA-Löschfunktion kann ein unbeabsichtigtes Löschen von Buckets verhindern. Wenn die MFA-Löschfunktion nicht aktiviert ist, kann jeder Benutzer, der über das Passwort und eine Root mit ausreichenden Berechtigungen verfügt oder jeder IAM-Benutzer ein Amazon S3-Objekt endgültig löschen.

Die MFA-Löschfunktion erfordert eine zusätzliche Authentifizierung für die folgenden Operationen:

- Ändern des Versioning-Status Ihres Buckets
- Dauerhaftes Löschen einer Objektversion

Weitere Informationen finden Sie unter MFA Delete (p. 516).

Erwägen Sie die Verschlüsselung von Daten im Ruhezustand

Mithilfe der folgenden Optionen können Sie Daten im Ruhezustand in Amazon S3 schützen:

- Serverseitige Verschlüsselung – Sie fordern Amazon S3 auf, Ihr Objekt zu verschlüsseln, bevor es dieses Objekt auf Datenträger in seinen Rechenzentren schreibt, und es wieder zu entschlüsseln, wenn Sie die Objekte herunterladen. Die serverseitige Verschlüsselung kann eine Gefährdung Ihrer Daten durch Verschlüsseln der Daten mit einem Schlüssel verringern, der durch einen anderen Mechanismus gespeichert wird, als der Mechanismus, mit dem die Daten gespeichert werden.

  Amazon S3 bietet folgende serverseitige Verschlüsselungsoptionen:

  - Serverseitige Verschlüsselung mit Amazon S3--verwalteten Schlüsseln (SSE-S3).
  - Serverseitige Verschlüsselung mit Kundenmasterschlüsseln (CMKs), die in AWS Key Management Service (SSE-KMS) gespeichert sind.
  - Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)

  Weitere Informationen finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290).
- Verwendung der clientseitigen Verschlüsselung – Sie können Daten clientseitig verschlüsseln und die verschlüsselten Daten auf Amazon S3 hochladen. In diesem Fall verwalten Sie den Verschlüsselungsprozess, die Verschlüsselungsschlüssel und die zugehörigen Tools. Genau wie die serverseitige Verschlüsselung kann die client-seitige Verschlüsselung das Risiko reduzieren, indem die Daten mit einem Schlüssel verschlüsselt werden, der durch einen anderen Mechanismus gespeichert wird als der, der die Daten selbst speichert

Amazon S3 bietet mehrere serverseitige Verschlüsselungsoptionen. Weitere Informationen finden Sie unter Schützen von Daten mithilfe clientseitiger Verschlüsselung (p. 322).

Erzwingen der Verschlüsselung von Daten bei der Übertragung

Mithilfe von HTTPS (TLS) kann das Abhören oder eine Manipulation des Netzwerkverkehrs durch Person-in-the-Middle-Angriffe verhindert werden. Sie sollten nur verschlüsselte Verbindungen über HTTPS (TLS) unter Anwendung der Bedingung aws:SecureTransport auf Amazon S3-Bucketrichtlinien zulassen.

Sie sollten auch mithilfe der verwalteten AWS Config-Regel s3-bucket-ssl-requests-only laufende detektive Kontrollen implementieren.

Beispielsweise S3 Objektsperre

S3 Objektsperre ermöglicht es Ihnen, Objekte mit einem WORM-Modell („Write Once Read Many“) zu speichern. S3 Objektsperre kann dazu beitragen, versehentliches oder unangemessenes Löschen von Daten zu verhindern. Beispielsweise können Sie S3 Objektsperre zum Schutz Ihrer AWS CloudTrail-Protokolle verwenden.

Aktivieren von Versioning

Das Versioning ermöglicht Ihnen, mehrere Versionen eines Objekts im selben Bucket aufzubewahren. Sie können Versioning verwenden, um sämtliche Versionen aller Objekte in Ihrem Amazon S3 Bucket zu speichern, abzurufen oder wiederherzustellen. Daten lassen sich dank Versioning nach unbeabsichtigten Benutzeraktionen und Anwendungsfehlern leicht wiederherstellen.

Sie sollten auch mithilfe der verwalteten AWS Config-Regel s3-bucket-versioning-enabled laufende detektive Kontrollen implementieren.

Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514).

Erwägen Sie die regionsübergreifende Replikation von Amazon S3

Auch wenn Amazon S3 Ihre Daten standardmäßig in mehreren entfernten Availability Zones speichert, machen es die Compliance-Anforderungen möglicherweise erforderlich, Daten in noch größeren Entfernungen zu speichern. Die regionsübergreifende Replikation (CRR) ermöglicht es, Daten zwischen entfernten AWS-Regionen zu replizieren, um diese Anforderungen zu erfüllen. CRR ermöglicht automatisches, asynchrones Kopieren von Objekten in den Buckets verschiedener AWS-Regionen. Weitere Informationen finden Sie unter Replikation (p. 669).

Note

CRR erfordert, dass Versioning in Quell- und Ziel-S3-Buckets aktiviert ist.

Sie sollten auch mithilfe der verwalteten AWS Config-Regel s3-bucket-replication-enabled laufende detektive Kontrollen implementieren.

Erwägen Sie VPC-Endpunkte für den Amazon S3-Zugriff

Ein VPC-Endpunkt für Amazon S3 ist eine logische Entity innerhalb einer Amazon Virtual Private Cloud (Amazon VPC), die ausschließlich Verbindungen mit Amazon S3 zulässt. Sie können den Zugriff auf Buckets von bestimmten Amazon VPC-Endpunkten oder bestimmten VPCs über Amazon S3-Bucket-Richtlinien steuern. Ein VPC-Endpunkt kann verhindern, dass Datenverkehr ins offene Internet gelangt.

VPC-Endpunkte für Amazon S3 unterstützen die Kontrolle des Zugriffs auf Ihre Amazon S3-Daten auf zwei Arten:

- Sie können steuern, welche Anfragen, Benutzer oder Gruppen durch einen spezifischen VPC-Endpunkt erlaubt sind.
- Mit S3 Bucket-Richtlinien können Sie steuern, welche VPCs oder VPC-Endpunkte Zugriff auf Ihre S3-Buckets haben.

- Sie können eine Daten-Exfiltration verhindern, indem Sie eine VPC verwenden, die kein Internet-Gateway hat.

Weitere Informationen finden Sie unter Beispiel für Bucket-Richtlinien für VPC-Endpunkte für Amazon S3 (p. 454).

Bewährte Methoden zur Überwachung und Prüfung von Amazon S3

Mithilfe der folgenden bewährten Methoden für Amazon S3 können Sie potenzielle Sicherheitsschwächen und Vorfälle erkennen.

Identifizieren und prüfen Sie alle Ihre Amazon S3-Buckets

Die Identifikation Ihrer IT-Assets ist ein wichtiger Aspekt von Governance und Sicherheit. Es ist erforderlich, dass Sie alle Ihre Amazon S3-Ressourcen sehen, um ihren Sicherheitsstatus beurteilen und Maßnahmen gegen potenzielle Schwachstellen ergreifen zu können.

Verwenden Sie den Tag-Editor, um sicherheits- und prüfungsrelevante Ressourcen zu identifizieren. Verwenden Sie dann diese Tags zur Suche nach den entsprechenden Ressourcen. Weitere Informationen finden Sie unter Suchen nach zu markierenden Ressourcen.

Verwenden Sie Amazon S3-Inventar für die Prüfung und Meldung des Replikations- und Verschlüsselungsstatus Ihrer Objekte für Unternehmens-, Compliance- und regulatorische Anforderungen. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503).

Erstellen Sie Ressourcengruppen für Ihre Amazon S3-Ressourcen. Weitere Informationen finden Sie unter Was sind AWS-Ressourcengruppen?

Implementieren Sie die Überwachung mithilfe von AWS-Überwachungstools

Die Überwachung ist wichtig, um Zuverlässigkeit, Sicherheit, Verfügbarkeit und Leistung von Amazon S3 und Ihrer AWS-Lösungen aufrechtzuerhalten. AWS bietet mehrere Tools und Services, mit deren Hilfe Sie Amazon S3 und Ihre anderen AWS-Services überwachen können. Sie können beispielsweise CloudWatch-Metriken für Amazon S3, insbesondere `PutRequests`, `GetRequests`, `4xxErrors` und `DeleteRequests` überwachen. Weitere Informationen erhalten Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739) und Überwachung von Amazon S3 (p. 738).

Ein zweites Beispiel finden Sie unter Beispiel: Amazon S3-Bucket-Aktivität. In diesem Beispiel wird beschrieben, wie Sie einen Amazon CloudWatch-Alarm erstellen, der ausgelöst wird, wenn ein Amazon S3-API-Aufruf für eine PUT- oder DELETE-Bucket-Richtlinie, ein Bucket-Lebenszyklus, eine Bucket-Replikation oder ein PUT für eine Bucket-ACL ausgeführt wird.

Aktivieren der Amazon S3-Server-Zugriffsprotokollierung

Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an einen Bucket gestellt wurden. Server-Zugriffsprotokolle können Sie bei Sicherheits- und Zugriffsprüfungen unterstützen, sie liefern Informationen über Ihren Kundenstamm und helfen beim Verständnis der Amazon S3-Rechnung. Weitere Informationen zur Aktivierung der Server-Zugriffsprotokollierung finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).

Sie sollten auch mithilfe der verwalteten AWS Config-Regel s3-bucket-logging-enabled laufende detektive Kontrollen implementieren.

Verwendung von AWS CloudTrail

AWS CloudTrail liefert Aufzeichnungen der Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in Amazon S3. Anhand der von CloudTrail gesammelten Informationen können Sie die an Amazon S3 gesendete Anforderung, die IP-Adresse, von der die Anforderung gesendet wurde,

den Initiator sowie den Zeitpunkt der Anforderung und weitere Angaben bestimmen. Sie können beispielsweise CloudTrail-Einträge für PUT-Aktionen identifizieren, die eine Auswirkung auf den Datenzugriff haben, insbesondere `PutBucketAcl`, `PutObjectAcl`, `PutBucketPolicy` und `PutBucketWebsite`. Wenn Sie Ihr AWS-Konto einrichten, ist CloudTrail standardmäßig aktiviert. Sie können die letzten Ereignisse in der CloudTrail-Konsole anzeigen. Um eine laufende Aufzeichnung von Aktivitäten und Ereignissen für Ihre Amazon S3-Buckets zu erstellen, können Sie einen Trail in der CloudTrail-Konsole erstellen. Weitere Informationen finden Sie unter Protokollieren von Datenereignissen für Trails im AWS CloudTrail User Guide.

Wenn Sie eine Trail erstellen, können Sie CloudTrail zum Protokollieren von Datenereignissen konfigurieren. Datenereignisse sind Datensätze von Ressourcen-Operationen, die auf oder innerhalb einer Ressource ausgeführt werden. In Amazon S3 erfassen Datenereignisse API-Aktivitäten auf Objektebene für einzelne Buckets. CloudTrail unterstützt eine Teilmenge von Amazon S3-API-Operationen auf Objektebene wie `GetObject`, `DeleteObject` und `PutObject`. Weitere Information zur Funktionsweise von CloudTrail mit Amazon S3 finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail (p. 750). In der Amazon S3-Konsole können Sie Ihre S3-Buckets auch so konfigurieren, dass die Protokollierung auf Objektebene für CloudTrail aktiviert wird.

AWS Config stellt eine verwaltete Regel (`cloudtrail-s3-dataevents-enabled`) bereit, mit der Sie bestätigen können, dass mindestens ein CloudTrail-Trail Datenereignisse für Ihre S3-Buckets protokolliert. Weitere Informationen finden Sie unter `cloudtrail-s3-dataevents-enabled` im AWS Config Developer Guide.

Aktivieren von AWS Config

Mehrere der in diesem Thema aufgeführten bewährten Methoden schlagen die Erstellung von AWS Config-Regeln vor. Mit AWS Config können Sie die Konfigurationen Ihrer AWS-Ressourcen bewerten und überprüfen. AWS Config überwacht Ressourcenkonfigurationen und ermöglicht Ihnen einen Vergleich der aufgezeichneten Konfigurationen mit gewünschten sicheren Konfigurationen. Mit AWS Config können Sie Änderungen an Konfigurationen von und Beziehungen zwischen AWS-Ressourcen überprüfen, detaillierte Verläufe der Ressourcenkonfiguration analysieren und die generelle Konformität mit den in Ihren internen Richtlinien festgelegten Konfigurationen überprüfen. Dadurch können Sie die Compliance-Prüfung, die Sicherheitsanalyse, das Änderungsmanagement und die Fehlerbehebung bei Betriebsabläufen vereinfachen. Weitere Informationen finden Sie unter Einrichten von AWS Config mit der Konsole im AWS Config-Entwicklerhandbuch. Achten Sie bei der Angabe der Ressourcentypen, die aufgezeichnet werden sollen, darauf, dass Amazon S3-Ressourcen enthalten sind.

Ein Beispiel zur Verwendung von AWS Config, um Amazon S3-Buckets, die öffentlichen Zugriff gestatten, zu überwachen und auf diese zu reagieren, finden Sie unter Verwendung von AWS Config, um Amazon S3-Buckets, die öffentlichen Zugriff gestatten, zu überwachen und auf diese zu reagieren im Blog zur AWS-Sicherheit.

Erwägen Sie die Verwendung von Amazon Macie mit Amazon S3

Macie verwendet Methoden des maschinellen Lernens, um sensible Daten in AWS automatisch zu erkennen, zu klassifizieren und zu schützen. Macie erkennt sensible Daten, wie persönlich identifizierbare Informationen (PII) oder geistiges Eigentum. Sie erhalten Dashboards und Warnungen, die sichtbar machen, wie auf Daten zugegriffen wird und wie diese verschoben werden. Weitere Informationen finden Sie unter Was ist Amazon Macie?

Überwachen von AWS-Sicherheitshinweisen

Sie sollten die in Trusted Advisor für Ihr AWS-Konto geposteten Sicherheitsempfehlungen regelmäßig überprüfen. Achten Sie dabei besonders auf Warnungen zu Amazon S3-Buckets mit „offenen Zugriffsberechtigungen“. Sie können dies auch programmgesteuert mit describe-trusted-advisor-checks durchführen.

Überwachen Sie außerdem die primäre E-Mail-Adresse, mit der Ihre einzelnen AWS-Konten registriert sind. AWS sendet Nachrichten zu auftretenden Sicherheitsproblemen, die Sie betreffen könnten, auf diese E-Mail-Adresse.

Operative AWS-Probleme mit weitreichenden Auswirkungen werden auf dem AWS Service Health Dashboard gepostet. Operative Probleme werden ebenfalls über das Personal Health Dashboard in den einzelnen Konten gepostet. Weitere Informationen finden Sie in der AWS Health-Dokumentation.

Durchführen von S3 Stapeloperationen

Mit S3 Stapeloperationen können Sie umfassende Stapeloperationen für Amazon S3-Objekte ausführen. S3 Stapeloperationen kann eine Operation für von Ihnen angegebene Listen von Amazon S3-Objekten ausführen. Ein einziger Auftrag kann die festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen. Amazon S3 verfolgt den Fortschritt, versendet Benachrichtigungen und speichert einen detaillierten Abschlussbericht zu allen Aktionen. So profitieren Sie von einem vollständig verwalteten, prüfbaren und serverlosen Erlebnis. Sie können S3 Stapeloperationen über die AWS Management Console, die AWS CLI, die AWS-SDKs oder die REST-API verwenden.

Mit S3 Stapeloperationen können Sie Objekte kopieren, Objekttags oder Zugriffskontrolllisten (ACLs) festlegen. Darüber hinaus können Sie Objektwiederherstellungen von Amazon S3 Glacier initiieren oder eine AWS Lambda-Funktion aufrufen, um benutzerdefinierte Aktionen mit Ihren Objekten durchzuführen. Sie können diese Operationen für eine benutzerdefinierten Liste von Objekten ausführen oder mit einem Amazon S3-Bestandsbericht auch größte Objektlisten generieren. Amazon S3 Stapeloperationen verwendet dieselben Amazon S3-APIs, die Sie bereits mit Amazon S3verwenden – Sie kennen die Schnittstelle also bereits.

Themen

Terminologie

In diesem Abschnitt werden die Begriffe Aufträge, Operationen und Aufgaben erwähnt, ihre Bedeutung ist wie folgt zu verstehen:

Job

Ein Auftrag ist die grundlegende Arbeitseinheit für S3 Stapeloperationen. Ein Auftrag umfasst alle Informationen, die erforderlich sind, um die angegebene Operation für die im Manifest aufgeführten Objekte auszuführen. Sobald Sie diese Informationen bereitgestellt und den Beginn des Auftrags angefordert haben, führt der Auftrag die Operation für alle Objekte im Manifest durch.

Operation

Die Operation stellt die Art der API-Aktion dar, z. B. das Kopieren von Objekten, die der Auftrag Stapeloperationen ausführen soll. Jeder Auftrag führt einen einzelnen Typ von Operation für alle Objekte aus, die im Manifest angegeben sind.

Aufgabe

Eine Aufgabe ist die Ausführungseinheit für einen Auftrag. Eine Aufgabe steht für einen einzelnen Aufruf eines Amazon S3- oder AWS Lambda-API-Vorgangs, um die Operation des Auftrags für ein einzelnes Objekt durchzuführen. Während der Lebensdauer eines Auftrag erstellt S3 Stapeloperationen eine Aufgabe für jedes Objekt, das im Manifest angegeben ist.

Die Grundlagen: Amazon S3-Aufträge für die Stapeloperation

Mit S3 Stapeloperationen können Sie umfassende Stapeloperationen für Amazon S3-Objekte ausführen. S3 Stapeloperationen kann eine Operation oder Aktion für von Ihnen angegebene Listen von Amazon S3-Objekten ausführen.

Themen

Funktionsweise eines S3 Stapeloperationen-Auftrags

Ein Auftrag ist die grundlegende Arbeitseinheit für S3 Stapeloperationen. Ein Auftrag umfasst alle Informationen, die erforderlich sind, um die angegebene Operation für eine Liste von Objekten auszuführen.

Um einen Auftrag zu erstellen, übergeben Sie S3 Stapeloperationen eine Liste von Objekten und geben die Aktion an, die für diese Objekte ausgeführt werden soll. S3 Stapeloperationen unterstützt die folgenden Operationen:

- PUT copy object
- PUT Object Tagging
- PUT object ACL
- Initiieren einer S3 Glacier-Wiederherstellung
- Aufrufen einer AWS Lambda-Funktion

Die Objekte, für die Sie einen Auftrag ausführen möchten, sind in einem Manifest-Objekt aufgelistet. Ein Auftrag führt die festgelegte Operation für jedes Objekt aus, das im Manifest enthalten ist. Sie können einen Amazon S3-Bestand (p. 503)-Bericht als Manifest nutzen, was die Erstellung umfassender Listen von Objekten in einem Bucket erleichtert. Ein Manifest können Sie auch in einem einfachen CSV-Format festlegen, mit dem Sie Stapeloperationen für eine benutzerdefinierte Liste an Objekten durchführen können, die in einem einzelnen Bucket enthalten sind.

Nachdem Sie einen Auftrag erstellt haben, verarbeitet Amazon S3 die Liste der Objekte im Manifest und führt die festgelegte Operation für jedes Objekt aus. Während ein Auftrag ausgeführt wird, können Sie den Fortschritt programmgesteuert oder über die Amazon S3-Konsole überwachen. Sie können einen Auftrag auch so konfigurieren, dass er nach Ende der Ausführung einen Abschlussbericht erzeugt. Der Abschlussbericht beschreibt die Ergebnisse jeder Aufgabe, die vom Auftrag durchgeführt wurde. Weitere allgemeine Informationen zur Überwachung von Aufträgen finden Sie unter Verwalten von S3 Stapeloperationen-Aufträgen (p. 571).

Angeben eines Manifests

Ein Manifest ist ein Amazon S3-Objekt, das Objektschlüssel auflistet, für die Amazon S3 eine Aktion ausführen soll. Um ein Manifest für einen Auftrag zu erstellen, geben Sie den Objektschlüssel des Manifests, das ETag und die optionale Versions-ID an. Die Inhalte des Manifests müssen per URL verschlüsselt sein. Manifeste, die eine serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C) und ein serverseitige Verschlüsselung mit AWS Key Management Service Kundenstammschlüsseln (Customer Master Key, CMK) (SSE-KMS) verwenden, werden nicht unterstützt. Ihr Manifest muss den Bucket-Namen, den Objektschlüssel und (optional) die Objektversion enthalten. Andere Felder im Manifest werden von S3 Stapeloperationen nicht verwendet.

Sie können ein Manifest in einer Auftragserstellungsanforderung in einem der beiden folgenden Formate erstellen.

- Amazon S3-Bestandsbericht — Muss ein CSV-formatierter Amazon S3-Bestandsbericht sein. Sie müssen die `manifest.json`-Datei angeben, die mit dem Bestandsbericht verknüpft ist. Weitere Informationen zu den Bestandsberichten finden Sie unter Amazon S3-Bestand (p. 503). Enthält der Bestandsbericht Versions-IDs, führt S3 Stapeloperationen die Aktionen auf den spezifischen Objektversionen aus.
- CSV-Datei: Jede Zeile in der Datei muss den Bucket-Namen, den Objektschlüssel und (optional) die Objektversion enthalten. Objektschlüssel müssen URL-codiert sein, wie in den folgenden Beispielen gezeigt. Das Manifest muss Versions-IDs für alle Objekte enthalten oder auslassen. Weitere Informationen über das CSV-Manifest-Format finden Sie unter JobManifestSpec in der Amazon Simple Storage Service API Reference.

  Im Folgenden finden Sie ein Beispiel für ein Manifest im CSV-Format ohne Versions-IDs.

```
Examplebucket,objectkey1
Examplebucket,objectkey2
Examplebucket,objectkey3
Examplebucket,photos/jpgs/objectkey4
Examplebucket,photos/jpgs/newjersey/objectkey5
Examplebucket,object%20key%20with%20spaces
```

  Im Folgenden finden Sie ein Beispiel für ein Manifest im CSV-Format mit Versions-IDs.

```
Examplebucket,objectkey1,PZ9ibn9D5lP6p298B7S9_ceqx1n5EJ0p
Examplebucket,objectkey2,YY_ouuAJByNW1LRBfFMfxMge7XQWxMBF
Examplebucket,objectkey3,jbo9_jhdPEyB4RrmOxWS0kU0EoNrU_oI
Examplebucket,photos/jpgs/objectkey4,6EqlikJJxLTsHsnbZbSRffn24_eh5Ny4
Examplebucket,photos/jpgs/newjersey/objectkey5,imHf3FAiRsvBW_EHB8GOu.NHunHO1gVs
Examplebucket,object%20key%20with%20spaces,9HkPvDaZY5MVbMhn6TMn1YTb5ArQAo3w
```

  Important

  Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, sollten Sie die Versions-IDs für die Objekte angeben. Wenn Sie einen Auftrag erstellen, analysiert S3 Stapeloperationen das gesamte Manifest vor Ausführung des Auftrags. Es wird jedoch kein „Snapshot“ des Bucket-Zustands aufgenommen.
  Da Manifeste Milliarden von Objekten enthalten können, kann die Ausführung von Aufträgen sehr lange dauern. Wenn Sie ein Objekt mit einer neuen Version überschreiben, während ein Auftrag ausgeführt wird, und keine Versions-ID für dieses Objekt angegeben haben, führt Amazon S3 die Operation für die neueste Version des Objekts durch und nicht für die Version, die bei der Erstellung des Auftrags vorlag. Die einzige Möglichkeit, dieses Verhalten zu verhindern, besteht darin, eine Versions-ID für die im Manifest enthaltenen Objekte anzugeben.

Erstellen eines S3 Stapeloperationen-Auftrags

Mit S3 Stapeloperationen können Sie Stapeloperationen an einer Liste spezifischer Amazon S3-Objekte im großen Umfang durchführen. Sie können S3 Stapeloperationen-Aufträge mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDKs oder REST-API erstellen.

In diesem Abschnitt werden die Informationen beschrieben, die Sie benötigen, um einen S3 Stapeloperationen-Auftrag zu erstellen. Darüber hinaus wird das Ergebnis einer `Create Job`-Anforderung beschrieben.

Note

Schritt-für-Schritt-Anleitungen zum Erstellen eines Auftrags mit der Amazon S3-Konsole finden Sie unter Erstellen eines S3 Stapeloperationen-Auftrags im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Erstellen einer Auftragsanforderung

Um einen S3-Stapeloperationen-Auftrag zu erstellen, müssen Sie folgende Informationen angeben:

Operation

Geben Sie die Operation an, die S3 Stapeloperationen für die Objekte im Manifest ausführen soll. Jeder Operationstyp akzeptiert Parameter, die für diesen Vorgang spezifisch sind. Auf diese Weise können Sie die gleichen Aufgaben ausführen, als ob Sie die Operation einzeln für jedes Objekt ausgeführt hätten.

Manifest

Das Manifest ist eine Liste aller Objekte, für die S3 Stapeloperationen die festgelegte Aktion ausführen soll. Sie können einen Amazon S3-Bestand (p. 503)-Bericht im CSV-Format als Manifest nutzen oder Ihre eigene benutzerdefinierte CSV-Liste von Objekten verwenden.

Weitere Informationen zu Manifesten finden Sie unter Angeben eines Manifests (p. 553).

Priorität

Mit Auftragsprioritäten können Sie angeben, welche relative Priorität dieser Auftrag gegenüber den anderen in Ihrem Konto ausgeführten Aufträgen besitzt. Eine höhere Nummer bedeutet eine höhere Priorität.

Auftragsprioritäten sind nur in Beziehung zu den für andere Aufträge in demselben Konto und in der derselben Region festgelegten Prioritäten bedeutsam. So können Sie wählen, welches Nummerierungssystem für Sie funktioniert. So können Sie beispielsweise allen `Initiate Restore Object`-Aufträgen die Priorität 1, allen `PUT Object Copy`-Aufträgen die Priorität 2 und allen `Put Object ACL`-Aufträgen die Priorität 3 zuweisen.

S3 Stapeloperationen priorisiert Aufträge gemäß den Prioritätszahlen, garantieren aber keine strikte Sortierung. Daher sind Auftragsprioritäten nicht dazu geeignet, sicherzustellen, dass ein Auftrag vor einem anderen Auftrag startet oder endet. Wenn Sie eine strikte Sortierung gewährleisten möchten, müssen Sie den Abschluss eines Auftrags abwarten, bevor Sie den nächsten starten.

RoleArn

Sie müssen eine AWS Identity and Access Management (IAM)-Rolle angeben, um den Auftrag auszuführen. Die verwendete IAM-Rolle muss über ausreichende Berechtigungen verfügen, um die im Auftrag festgelegte Operation durchzuführen. Um zum Beispiel einen `PUT Object Copy`-Auftrag auszuführen, benötigt die IAM-Rolle die `s3:GetObject`-Berechtigungen für den Quell-Bucket und die `s3:PutObject`-Berechtigungen für den Ziel-Bucket. Außerdem braucht die Rolle die Berechtigungen, das Manifest zu lesen und den Auftragsabschlussbericht zu schreiben.

Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

Weitere Informationen zu Amazon S3-Berechtigungen finden Sie unter Amazon S3-Aktionen (p. 380).

Bericht

Geben Sie an, ob S3 Stapeloperationen einen Auftragsabschlussbericht erstellen soll oder nicht. Wenn Sie einen Auftragsabschlussbericht anfordern, müssen Sie auch die Parameter für den Bericht in diesem Element angeben. Zu den notwendigen Angaben gehören der Bucket, in dem der Bericht gespeichert werden soll, das Format des Berichts, ob der Bericht die Details aller Aufgaben oder nur der fehlgeschlagenen Aufgaben enthalten soll, und eine optionale Präfix-Zeichenfolge.

Tags (optional)

Sie können den Zugriff auf Ihre S3 Stapeloperationen-Aufträge bezeichnen und steuern, indem Sie Tags hinzufügen. Anhand von Tags kann identifiziert werden, wer für einen Stapeloperationen-Job verantwortlich ist. Sie können Aufträge mit bereits zugeordneten Tags erstellen und Sie können bereits erstellten Aufträgen nachträglich Tags hinzufügen. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung zum Aufrufen von `CreateJob` erteilen, vorausgesetzt, der Auftrag wird mit dem Tag `"Department=Finance"` erstellt.

Weitere Informationen finden Sie unter the section called “Steuern von Zugriffs- und Labeling-Aufträgen mithilfe von Tags” (p. 572).

Beschreibung (optional)

Um die Nachverfolgung und Überwachung des Auftrags zu unterstützen, können Sie auch eine Beschreibung mit bis zu 256 Zeichen bereitstellen. Amazon S3 schließt diese Beschreibung ein, wenn Informationen zu einem Auftrag zurückgegeben oder Auftragsdetails in der Amazon S3-Konsole angezeigt werden. So können Sie Aufträge ganz einfach nach den ihnen zugewiesenen Beschreibungen sortieren und filtern. Beschreibungen müssen nicht eindeutig sein, sodass Sie Beschreibungen auch als Kategorien (z. B. "Wöchentliche Aufträge zum Kopieren von Protokollen") verwenden können, um Gruppen ähnlicher Aufträge nachzuverfolgen.

Erstellen einer Auftragsantwort

Wenn die `Create Job`-Anforderung erfolgreich ist, gibt Amazon S3 eine Auftrags-ID zurück. Die Auftrags-ID ist eine eindeutige Kennung, die Amazon S3 automatisch erstellt. Anhand dieser ID können Sie Ihren Stapeloperationen-Auftrag identifizieren und seinen Status überwachen.

Wenn Sie einen Auftrag über die AWS CLI, AWS SDKs oder REST API erstellen, können Sie die S3 Stapeloperationen so einrichten, dass automatisch mit der Verarbeitung des Auftrags begonnen wird. Der Auftrag wird ausgeführt, sobald er bereit ist und muss nicht hinter Aufträgen mit höherer Priorität warten.

Wenn Sie einen Auftrag über die AWS Management Console erstellen, müssen Sie die Auftragsdetails überprüfen und bestätigen, dass Sie ihn ausführen möchten. Erst dann beginnt Stapeloperationen mit der Verarbeitung. Nachdem Sie bestätigt haben, dass Sie den Auftrag ausführen möchten, wird er so verarbeitet und ausgeführt, als wäre er mit einer der anderen Methoden erstellt worden. Wenn ein Auftrag den Status „Ausgesetzt“ mehr als 30 Tage behält, schlägt er fehl.

Gewähren von Berechtigungen für Amazon S3-Stapeloperationen

In diesem Abschnitt wird beschrieben, wie die zur Erstellung und Ausführung von S3 Stapeloperationen-Aufträgen erforderlichen Berechtigungen gewährt werden.

Themen

Erforderliche Berechtigungen zum Erstellen eines S3 Stapeloperationen-Auftrags

Um einen Amazon S3 Stapeloperationen-Auftrag zu erstellen, ist die Berechtigung `s3:CreateJob` erforderlich. Dieselbe Entity, die den Auftrag erstellt, muss auch über die Berechtigung `iam:PassRole` verfügen, um die für den Auftrag festgelegte AWS Identity and Access Management (IAM)-Rolle an

Amazon S3 zu übergebenStapeloperationen. Weitere Informationen zum Erstellen dieser IAM-Rolle finden Sie im nächsten Thema Erstellen einer S3 Stapeloperationen IAM-Rolle (p. 557).

Erstellen einer S3 Stapeloperationen IAM-Rolle

Amazon S3 muss dazu berechtigt sein, S3 Stapeloperationen in Ihrem Namen auszuführen. Sie gewähren diese Berechtigungen über eine AWS Identity and Access Management (IAM)-Rolle. Dieser Abschnitt enthält Beispiele zu den Vertrauens- und Berechtigungsrichtlinien, die Sie beim Erstellen einer IAM-Rolle verwenden. Weitere Informationen finden Sie unter IAM-Rollen. Beispiele finden Sie unter Beispiel: Verwenden von Auftrags-Tags zum Steuern von Berechtigungen für S3 Stapeloperationen (p. 577).

Sie können in Ihren IAM-Richtlinien auch Bedingungsschlüssel verwenden, um Zugriffsberechtigungen für S3 Stapeloperationen-Aufgaben zu filtern. Weitere Informationen und die vollständige Liste der Amazon S3-spezifischen Bedingungsschlüssel finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Das folgende Video zeigt, wie Sie mithilfe der AWS Management Console IAM-Berechtigungen für Stapeloperationen-Aufträge einrichten.

Vertrauensrichtlinie

U dem S3 Stapeloperationen-Service-Prinzipal die Berechtigung zu erteilen, die IAM-Rolle anzunehmen, hängen Sie der Rolle die folgende Vertrauensrichtlinie an.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Berechtigungsrichtlinien

Je nach Art der Operation können Sie eine der folgenden Richtlinien anfügen.

Note

- Unabhängig von der Operation benötigt Amazon S3 Berechtigungen, um das Manifestobjekt aus Ihrem S3-Bucket zu lesen und optional einen Bericht in Ihren Bucket zu schreiben. Daher enthalten alle der folgenden Richtlinien diese Berechtigungen.
- Für Amazon S3-Bestandsberichtmanifeste benötigt S3 Stapeloperationen die Berechtigung zum Lesen des manifest.json-Objekts sowie der zugehörigen CSV-Datendateien.
- Versionsspezifische Berechtigungen wie `s3:GetObjectVersion` sind nur erforderlich, wenn Sie die Versions-ID der Objekte festlegen.
- Wenn Sie S3 Stapeloperationen auf verschlüsselten Objekten ausführen, benötigt die IAM-Rolle auch Zugriff auf die AWS KMS-Schlüssel, die für die Verschlüsselung verwendet werden.

- PUT copy object

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectTagging"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::{{DestinationBucket}}/*"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::{{SourceBucket}}/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::{{ReportBucket}}/*"
            ]
        }
    ]
}
```

- PUT Object Tagging

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectTagging",
        "s3:PutObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::{{TargetResource}}/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::{{ManifestBucket}}/*"
```

```
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:GetBucketLocation"
      ],
      "Resource":[
        "arn:aws:s3:::{{ReportBucket}}/*"
      ]
    }
  ]
}
```

- PUT object ACL

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectAcl",
        "s3:PutObjectVersionAcl"
      ],
      "Resource": "arn:aws:s3:::{{TargetResource}}/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::{{ManifestBucket}}/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:GetBucketLocation"
      ],
      "Resource":[
        "arn:aws:s3:::{{ReportBucket}}/*"
      ]
    }
  ]
}
```

- Initiieren einer S3 Glacier-Wiederherstellung

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
          "s3:RestoreObject"
      ],
      "Resource": "arn:aws:s3:::{{TargetResource}}/*"
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::{{ManifestBucket}}/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:GetBucketLocation"
      ],
      "Resource":[
        "arn:aws:s3:::{{ReportBucket}}/*"
      ]
    }
  ]
}
```

Zugehörige Ressourcen

- Die Grundlagen: Amazon S3-Aufträge für die Stapeloperation (p. 553)
- Operationen (p. 560)
- Verwalten von S3 Stapeloperationen-Aufträgen (p. 571)

Operationen

S3 Stapeloperationen unterstützt fünf verschiedene Operationen. Die Themen in diesem Abschnitt beschreiben die einzelnen Vorgänge und deren Funktionsweise mit Stapeloperationen.

Themen

- PUT-Objektkopie (p. 560)
- Initiieren eines Wiederherstellungsobjekts (p. 561)
- Aufrufen einer Lambda-Funktion von Amazon S3-Stapelvorgängen (p. 562)
- Put Object ACL (p. 570)
- Put Object Tagging (p. 570)

PUT-Objektkopie

Durch die Operation „PUT Object – Kopieren“ wird jedes im Manifest angegebene Objekt kopiert. Sie können Objekte in einen anderen Bucket in der gleichen AWS-Region oder in einen Bucket in einer anderen Region kopieren. S3 Stapeloperationen unterstützt die meisten durch Amazon S3 verfügbaren Optionen zum Kopieren von Objekten. Zu diesen Optionen gehören das Festlegen von Objekt-Metadaten, das Festlegen von Berechtigungen sowie das Ändern der Speicherklasse eines Objekts. Weitere Informationen über durch Amazon S3 verfügbare Funktionen zum Kopieren von Objekten finden Sie unter Objekte kopieren (p. 233).

Beschränkungen und Einschränkungen

- Alle Quellobjekte müssen sich in einem Bucket befinden.
- Alle Zielobjekte müssen sich in einem Bucket befinden.
- Sie benötigen Lese-Berechtigungen für den Quell-Bucket und Schreib-Berechtigungen für den Ziel-Bucket.
- Die maximale Größe eines Objekts, das kopiert werden soll, beträgt 5 GB.
- „PUT Object – Kopieren“-Aufträge sollten in der Zielregion erstellt werden, also in der Region, in die die Objekte kopiert werden sollen.
- Alle „PUT Object – Kopieren“-Optionen werden unterstützt, mit Ausnahme der bedingten Prüfungen von ETags und Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln.
- Wenn die Buckets nicht versioniert sind, werden die Objekte mit dem gleichen Schlüsselnamen überschrieben.
- Objekte werden nicht zwingend in derselben Reihenfolge kopiert, in der sie im Manifest aufgelistet sind. Für versionierte Buckets sollten Sie zuerst alle nicht aktuellen Versionen kopieren und später die aktuellen Versionen in einer darauf folgenden Aufgabe kopieren, nachdem die erste Aufgabe abgeschlossen ist, falls das Beibehalten der aktuellen/nicht aktuellen Versionsreihenfolge wichtig ist.
- Das Kopieren von Objekten in die Speicherklasse Reduced Redundancy (RRS) wird nicht unterstützt.

Initiieren eines Wiederherstellungsobjekts

Sie können S3 Stapeloperationen verwenden, um umfassende Stapeloperationen für Amazon S3-Objekte durchzuführen. S3 Stapeloperationen kann eine Operation für eine von Ihnen angegebene Liste von Amazon S3-Objekten ausführen, dazu zählt auch das Initiieren von Objektwiederherstellungen aus Amazon S3 Glacier. Weitere Informationen finden Sie unter S3 Stapeloperationen (p. 552).

Auf Objekte, die Sie in den Speicherklassen S3 Glacier oder S3 Glacier Deep Archive archivieren, kann nicht in Echtzeit zugegriffen werden. Durch die Verwendung der `InitiateRestore`-Operation in Ihrem S3 Stapeloperationen wird für jedes im Manifest angegebene Objekt eine Wiederherstellungsanforderung an S3 Glacier gesendet. Um einen Auftrag „Initiate Restore Object (Objekt wiederherstellen initiieren)“ zu erstellen, müssen Sie Ihrer Anforderung zwei Elemente hinzufügen:

- ExpirationInDays

  Wenn Sie ein Objekt aus S3 Glacier wiederherstellen, handelt es sich bei dem Objekt nur um eine temporäre Kopie, die Amazon S3 nach einem angegebenen Zeitraum wieder löscht. Dieses Element gibt an, wie lange die temporäre Kopie in Amazon S3 verfügbar sein wird. Nachdem die temporäre Kopie abgelaufen ist, können Sie das Objekt nur abrufen, indem Sie es erneut aus S3 Glacier wiederherstellen. Weitere Informationen zur Objekt-Wiederherstellung finden Sie unter Wiederherstellen archivierter Objekte (p. 272).
- GlacierJobTier

  Amazon S3 kann Objekte aus S3 Glacier anhand von drei verschiedenen Wiederherstellungsebenen wiederherstellen: Expedited, Standard und Bulk. S3 Stapeloperationen unterstützt nur die Standard- und Bulk-Ebenen. Weitere Informationen zu den Wiederherstellungsebenen von S3 Glacier finden Sie unter Archiv-Abrufoptionen (p. 273). Weitere Informationen zur Preisgestaltung für die einzelnen Ebenen finden Sie unter dem Abschnitt „Datenabrufpreise“ unter Amazon S3 Glacier-Preise.

  Important

  Der Auftrag „Initiate Restore Object (Objekt wiederherstellen initiieren)“ leitet nur die Anforderung, Objekte wiederherzustellen, ein. S3 Stapeloperationen meldet den Auftrag für jedes Objekt als abgeschlossen, nachdem die Anforderung für das jeweilige Objekt initiiert wurde. Amazon S3 aktualisiert weder den Auftrag, noch informiert es Sie darüber, dass das Objekt wiederhergestellt

wurde. Sie können jedoch Ereignisbenachrichtigungen verwenden, um Benachrichtigungen zu erhalten, wenn die Objekte in Amazon S3 verfügbar sind. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).

Überlappende Wiederherstellungen

Sollte der Auftrag „Objekt wiederherstellen initiieren“ versuchen, ein Objekt wiederherzustellen, das gerade wiederhergestellt wird, verhält sich S3 Stapeloperationen folgendermaßen:

Der Wiederherstellungsvorgang für das Objekt ist erfolgreich, wenn eine der folgenden Bedingungen erfüllt ist:

- Im Vergleich zu der bereits ausgeführten Wiederherstellungsanforderung ist `ExpirationInDays` für diesen Auftrag identisch und `GlacierJobTier` ist schneller.
- Die vorherige Wiederherstellungsanforderung wurde bereits abgeschlossen und das Objekt ist derzeit verfügbar. In diesem Fall aktualisiert S3 Stapeloperationen das Ablaufdatum des wiederhergestellten Objekts, um mit dem für diesen Auftrag angegebenen `ExpirationInDays` übereinzustimmen.

Der Wiederherstellungsvorgang für das Objekt schlägt fehl, wenn eine der folgenden Bedingungen erfüllt sind:

- Die bereits ausgeführte Wiederherstellungsanforderung wurde noch nicht abgeschlossen und die Wiederherstellungsdauer für diesen Auftrag (durch `ExpirationInDays` angegeben) unterscheidet sich von der Wiederherstellungsdauer, die in der bereits ausgeführten Wiederherstellungsanforderung angegeben wurde.
- Die Wiederherstellungsebene für diesen Auftrag (durch `GlacierJobTier` angegeben) ist identisch oder langsamer als die Wiederherstellungsebene, die in der bereits ausgeführten Wiederherstellungsanforderung angegeben wurde.

Einschränkungen

„Objekt wiederherstellen initiieren“-Aufträge haben folgende Einschränkungen:

- Sie müssen „Objekt wiederherstellen initiieren“-Aufträge in der gleichen Region erstellen, in der sich das archivierte Objekt befindet.
- S3 Stapeloperationen unterstützt S3 Glacier SELECT nicht.
- S3 Stapeloperationen unterstützt die Ebene „Expedited retrieval“ (Beschleunigter Abruf) nicht.

Aufrufen einer Lambda-Funktion von Amazon S3-Stapelvorgängen

S3 Stapeloperationen kann AWS Lambda-Funktionen zur Durchführung benutzerdefinierter Aktionen auf Objekte aufrufen, die in einem Manifest aufgeführt sind. Dieser Abschnitt beschreibt die Erstellung einer Lambda-Funktion zur Verwendung mit S3 Stapeloperationen sowie die Erstellung einer Aufgabe zum Aufruf der Funktion. Die S3 Stapeloperationen-Aufgabe verwendet den `LambdaInvoke`-Vorgang für die Ausführung einer Lambda-Funktion für jedes in einem Manifest aufgelistete Objekt.

Sie können mit S3 Stapeloperationen für Lambda unter Verwendung der AWS Management Console, von AWS Command Line Interface (AWS CLI), der AWS SDKs oder von REST APIs arbeiten. Weitere Informationen zur Verwendung von Lambda finden Sie unter Erste Schritte mit AWS Lambda im AWS Lambda Developer Guide.

In den folgenden Themen werden die ersten Schritte bei der Verwendung von S3 Stapeloperationen mit Lambda erläutert.

Themen

Verwendung von Lambda mit Amazon S3-Stapeloperationen

Bei der Verwendung von S3 Stapeloperationen mit AWS Lambda müssen Sie neue Lambda-Funktionen erstellen, die speziell für die Verwendung mit S3 Stapeloperationen ausgerichtet sind. Sie können keine vorhandenen ereignisbasierten Amazon S3-Funktionen mit S3 Stapeloperationen wiederverwenden. Ereignisfunktionen können Nachrichten lediglich erhalten und nicht zurückgeben. Die Lambda-Funktionen, die mit S3 Stapeloperationen verwendet werden, müssen Nachrichten annehmen und zurückgeben. Weitere Informationen zur Verwendung von Lambda mit Amazon S3-Ereignissen finden Sie unter Verwendung von AWS Lambda mit Amazon S3 im AWS Lambda Developer Guide.

Sie erstellen eine S3 Stapeloperationen-Aufgabe, die Ihre Lambda-Funktion aufruft. Die Aufgabe führt dieselbe Lambda-Funktion für alle in Ihrem Manifest aufgeführten Objekte aus. Sie können steuern, welche Versionen Ihrer Lambda-Funktion bei der Verarbeitung der Objekte in Ihrem Manifest verwendet werden. S3 Stapeloperationen unterstützt nicht qualifizierte Amazon-Ressourcennamen (ARNs), Alias und spezifische Versionen. Weitere Informationen finden Sie unter Einführung in das AWS Lambda-Versioning im AWS Lambda Developer Guide.

Wenn Sie die S3 Stapeloperationen-Aufgabe mit einem Funktions-ARN versehen, der ein Alias oder den Qualifizierer `$LATEST` verwendet und die Version aktualisieren, auf den einer von beiden verweist, beginnt S3 Stapeloperationen mit dem Aufruf der neuen Version Ihrer Lambda-Funktion. Dies kann nützlich sein, wenn Sie mitten in der Ausführung einer großen Aufgabe Funktionen aktualisieren möchten. Wenn S3 Stapeloperationen die verwendete Version nicht ändern soll, geben Sie die spezifische Version im Parameter `FunctionARN` an, wenn Sie die Aufgabe erstellen.

Antwort- und Ergebniscodes

Es gibt zwei Codeebenen, die S3 Stapeloperationen von Lambda-Funktionen erwartet. Die erste ist der Antwortcode für die gesamte Anforderung, und die zweite ist ein Ergebniscode pro Aufgabe. Die folgende Tabelle enthält die Antwortcodes.

Antwortcode	Beschreibung
`Succeeded`	Die Aufgabe wurde normal abgeschlossen. Wenn Sie einen Aufgabenabschlussbericht angefordert haben, ist die Ergebniszeichenfolge der Aufgabe in dem Bericht enthalten.
`TemporaryFailure`	Die Aufgabe unterlag einem vorübergehenden Fehler und wird erneut ausgeführt, bevor der Job abgeschlossen ist. Die Ergebniszeichenfolge wird ignoriert. Wenn dies der letzte Versuch ist, ist die Fehlermeldung im abschließenden Bericht enthalten.
`PermanentFailure`	Die Aufgabe unterlag einem dauerhaften Fehler. Wenn Sie einen Aufgabenabschlussbericht angefordert haben, ist die Aufgabe als `Failed` markiert und enthält die Fehlermeldungszeichenfolge.

Antwortcode	Beschreibung
	Ergebniszeichenfolge aus fehlgeschlagenen Aufgaben werden ignoriert.

Erstellen einer Lambda-Funktion zur Verwendung mit S3 Stapeloperationen

Dieser Abschnitt enthält Beispiel-AWS Identity and Access Management (IAM)-Berechtigungen, die Sie mit der Lambda-Funktion verwenden müssen. Er enthält auch ein Beispiel einer Lambda-Funktion für die Verwendung mit S3 Stapeloperationen. Wenn Sie noch nie eine Lambda-Funktion erstellt haben, sollten Sie sich das Tutorial: Verwendung von AWS Lambda mit Amazon S3 im AWS Lambda Developer Guide ansehen.

Sie müssen Lambda-Funktionen speziell zur Verwendung mit S3 Stapeloperationen erstellen. Sie können keine vorhandenen Lambda-Funktionen verwenden, die auf Amazon S3-Ereignissen basieren. Der Grund dafür ist, dass Lambda-Funktionen, die für S3 Stapeloperationenverwendet werden, bestimmte Datenfelder akzeptieren und zurückgeben müssen.

Beispiele von IAM-Berechtigungen

Nachfolgend sehen Sie Beispiele für die IAM-Berechtigungen, die zur Verwendung einer Lambda-Funktion mit S3 Stapeloperationen erforderlich sind.

Example — S3 Stapeloperationen-Vertrauensrichtlinie

Im Folgenden finden Sie ein Beispiel für eine Vertrauensrichtlinie, die Sie für die Stapeloperationen IAM-Rolle verwenden können. Diese IAM-Rolle wird beim Erstellen des Auftrags angegeben und erteilt Stapeloperationen die Berechtigung, die IAM-Rolle zu übernehmen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "batchoperations.s3.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Example — Lambda-IAM-Richtlinie

Nachfolgend sehen Sie ein Beispiel für eine IAM-Richtlinie, die S3 Stapeloperationen die Berechtigung erteilt, die Lambda-Funktion aufzurufen und das Eingabemanifest zu lesen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BatchOperationsLambdaPolicy",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:PutObject",
                "lambda:InvokeFunction"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Beispiel: Anforderung und Antwort

Dieser Abschnitt bietet Beispiele für Anforderungen und Antworten für die Lambda-Funktion.

Example Anfrage

Im Folgenden sehen Sie ein JSON-Beispiel für eine Anforderung für die Lambda-Funktion.

```
{
    "invocationSchemaVersion": "1.0",
    "invocationId": "YXNkbGZqYWRmaiBhc2RmdW9hZHNmZGpmaGFzbGtkaGZza2RmaAo",
    "job": {
        "id": "f3cc4f60-61f6-4a2b-8a21-d07600c373ce"
    },
    "tasks": [
        {
            "taskId": "dGFza2lkZ29lc2hlcmUK",
            "s3Key": "customerImage1.jpg",
            "s3VersionId": "1",
            "s3BucketArn": "arn:aws:s3:us-east-1:0123456788:awsexamplebucket1"
        }
    ]
}
```

Example Antwort

Im Folgenden sehen Sie ein JSON-Beispiel für eine Antwort für die Lambda-Funktion.

```
{
  "invocationSchemaVersion": "1.0",
  "treatMissingKeysAs" : "PermanentFailure",
  "invocationId" : "YXNkbGZqYWRmaiBhc2RmdW9hZHNmZGpmaGFzbGtkaGZza2RmaAo",
  "results": [
    {
      "taskId": "dGFza2lkZ29lc2hlcmUK",
      "resultCode": "Succeeded",
      "resultString": "[\"Mary Major", \"John Stiles\"]"
    }
  ]
}
```

Lambda-Beispielfunktion für S3 Stapeloperationen

Die folgende Python-Lambda-Beispielfunktion verarbeitet das Manifest schrittweise und kopiert dabei jedes Objekt und benennt es um.

Wie das Beispiel zeigt, sind Schlüssel aus S3 Stapeloperationen URL-codiert. Wenn Amazon S3 mit anderen AWS-Services verwendet werden soll, muss der aus S3 Stapeloperationen übergebene Schlüssel URL-codiert werden.

```
import boto3
import urllib
from botocore.exceptions import ClientError

def lambda_handler(event, context):
```

```
# Instantiate boto client
s3Client = boto3.client('s3')

# Parse job parameters from S3 Stapeloperationen
jobId = event['job']['id']
invocationId = event['invocationId']
invocationSchemaVersion = event['invocationSchemaVersion']

# Prepare results
results = []

# Parse Amazon S3 Key, Key Version, and Bucket ARN
taskId = event['tasks'][0]['taskId']
s3Key = urllib.unquote(event['tasks'][0]['s3Key']).decode('utf8')
s3VersionId = event['tasks'][0]['s3VersionId']
s3BucketArn = event['tasks'][0]['s3BucketArn']
s3Bucket = s3BucketArn.split(':::')[-1]

# Construct CopySource with VersionId
copySrc = {'Bucket': s3Bucket, 'Key': s3Key}
if s3VersionId is not None:
    copySrc['VersionId'] = s3VersionId

# Copy object to new bucket with new key name
try:
    # Prepare result code and string
    resultCode = None
    resultString = None

    # Construct New Key
    newKey = rename_key(s3Key)
    newBucket = 'destination-bucket-name'

    # Copy Object to New Bucket
    response = s3Client.copy_object(
        CopySource = copySrc,
        Bucket = newBucket,
        Key = newKey
    )

    # Mark as succeeded
    resultCode = 'Succeeded'
    resultString = str(response)
except ClientError as e:
    # If request timed out, mark as a temp failure
    # and S3 Stapeloperationen will make the task for retry. If
    # any other exceptions are received, mark as permanent failure.
    errorCode = e.response['Error']['Code']
    errorMessage = e.response['Error']['Message']
    if errorCode == 'RequestTimeout':
        resultCode = 'TemporaryFailure'
        resultString = 'Retry request to Amazon S3 due to timeout.'
    else:
        resultCode = 'PermanentFailure'
        resultString = '{}: {}'.format(errorCode, errorMessage)
except Exception as e:
    # Catch all exceptions to permanently fail the task
    resultCode = 'PermanentFailure'
    resultString = 'Exception: {}'.format(e.message)
finally:
    results.append({
        'taskId': taskId,
        'resultCode': resultCode,
        'resultString': resultString
    })
```

```
    return {
        'invocationSchemaVersion': invocationSchemaVersion,
        'treatMissingKeysAs': 'PermanentFailure',
        'invocationId': invocationId,
        'results': results
    }

def rename_key(s3Key):
    # Rename the key by adding additional suffix
    return s3Key + '_new_suffix'
```

Erstellen einer S3 Stapeloperationen-Aufgabe, die eine Lambda-Funktion aufruft

Beim Erstellen einer S3 Stapeloperationen-Aufgabe zum Aufruf einer Lambda-Funktion müssen Sie Folgendes angeben:

- Den ARN Ihrer Lambda-Funktion (möglicherweise mit dem Funktionsalias oder einer spezifischen Versionsnummer)
- Eine IAM-Rolle mit der Berechtigung zum Aufruf der Funktion
- Den Aktionsparameter`LambdaInvokeFunction`

Weitere Informationen zum Erstellen einer S3 Stapeloperationen-Aufgabe finden Sie unter Erstellen eines S3 Stapeloperationen-Auftrags (p. 554) und Operationen (p. 560).

Im folgenden Beispiel wird eine S3 Stapeloperationen-Aufgabe erstellt, die mit der AWS CLI eine Lambda-Funktion aufruft.

```
aws s3control create-job
    --account-id <AccountID>
    --operation  '{"LambdaInvoke": { "FunctionArn":
 "arn:aws:lambda:Region:AccountID:function:LambdaFunctionName" } }'
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":
{"ObjectArn":"arn:aws:s3:::ManifestLocation","ETag":"ManifestETag"}}'
    --report
 '{"Bucket":"arn:aws:s3:::awsexamplebucket1","Format":"Report_CSV_20180820","Enabled":true,"Prefix":"Re
    --priority 2
    --role-arn arn:aws:iam::AccountID:role/BatchOperationsRole
    --region Region
    --description “Lambda Function"
```

Bereitstellen von Informationen auf Aufgabenebene in Lambda-Manifesten

Wenn Sie AWS Lambda-Funktionen mit S3 Stapeloperationen verwenden, möchten Sie möglicherweise weitere Daten für die einzelnen Aufgaben/Schlüssel, die der Ausführung zugrunde liegen. Möglicherweise möchten Sie, dass sowohl ein Quellobjektschlüssel als auch ein neuer Objektschlüssel bereitgestellt wird. Ihre Lambda-Funktion kann in einem solchen Fall den Quellschlüssel in einen neuen S3-Bucket unter einem neuen Namen speichern. Standardmäßig können Sie mit Amazon S3-Stapelvorgängen nur den Ziel-Bucket und eine Liste von Quellschlüsseln im Eingabemanifest für Ihre Aufgabe angeben. Im Folgenden wird beschrieben, wie Sie weitere Daten in Ihr Manifest aufnehmen können, sodass Sie komplexere Lambda-Funktionen ausführen können.

Zum Angeben von Pro-Schlüssel-Parametern in Ihrem S3 Stapeloperationen-Manifest für die Verwendung im Code der Lambda-Funktion verwenden Sie das folgende URL-kodierte JSON-Format. Das Feld `key` wird an Ihre Lambda-Funktion übergeben, als wäre es ein Amazon S3-Objektschlüssel. Es kann aber von

der Lambda-Funktion so interpretiert werden, dass es andere Werte oder mehrere Schlüssel enthält, wie im Folgenden gezeigt:

> **Note**
>
> Die maximale Anzahl von Zeichen für das Feld `key` im Manifest beträgt 1.024.

Example – Manifest, dass die „Amazon S3-Schlüssel“ durch JSON-Zeichenfolgen ersetzt

Die URL-kodierte Version muss S3 Stapeloperationen zur Verfügung gestellt werden.

```
my-bucket,{"origKey": "object1key", "newKey": "newObject1Key"}
my-bucket,{"origKey": "object2key", "newKey": "newObject2Key"}
my-bucket,{"origKey": "object3key", "newKey": "newObject3Key"}
```

Example — URL-kodiertes Manifest

Diese URL-kodierte Version muss S3 Stapeloperationen zur Verfügung gestellt werden. Eine Version, die nicht URL-kodiert ist, wird nicht unterstützt.

```
my-bucket,%7B%22origKey%22%3A%20%22object1key%22%2C%20%22newKey%22%3A%20%22newObject1Key
%22%7D
my-bucket,%7B%22origKey%22%3A%20%22object2key%22%2C%20%22newKey%22%3A%20%22newObject2Key
%22%7D
my-bucket,%7B%22origKey%22%3A%20%22object3key%22%2C%20%22newKey%22%3A%20%22newObject3Key
%22%7D
```

Example — Lambda-Funktion im Manifest-Format, die Ergebnisse in den Aufgabenbericht schreibt.

Diese Lambda-Funktion zeigt, wie JSON analysiert wird, das im S3 Stapeloperationen Manifest codiert ist.

```
import json
from urllib.parse import unquote_plus


# This example Lambda function shows how to parse JSON that is encoded into the Amazon S3
 batch
# operations manifest containing lines like this:
#
# bucket,encoded-json
# bucket,encoded-json
# bucket,encoded-json
#
# For example, if we wanted to send the following JSON to this Lambda function:
#
# bucket,{"origKey": "object1key", "newKey": "newObject1Key"}
# bucket,{"origKey": "object2key", "newKey": "newObject2Key"}
# bucket,{"origKey": "object3key", "newKey": "newObject3Key"}
#
# We would simply URL-encode the JSON like this to create the real manifest to create a
 batch
# operations job with:
#
# my-bucket,%7B%22origKey%22%3A%20%22object1key%22%2C%20%22newKey%22%3A%20%22newObject1Key
%22%7D
# my-bucket,%7B%22origKey%22%3A%20%22object2key%22%2C%20%22newKey%22%3A%20%22newObject2Key
%22%7D
# my-bucket,%7B%22origKey%22%3A%20%22object3key%22%2C%20%22newKey%22%3A%20%22newObject3Key
%22%7D
#
```

```
def lambda_handler(event, context):
    # Parse job parameters from S3 batch operations
    jobId = event['job']['id']
    invocationId = event['invocationId']
    invocationSchemaVersion = event['invocationSchemaVersion']

    # Prepare results
    results = []

    # S3 batch operations currently only passes a single task at a time in the array of
 tasks.
    task = event['tasks'][0]

    # Extract the task values we might want to use
    taskId = task['taskId']
    s3Key = task['s3Key']
    s3VersionId = task['s3VersionId']
    s3BucketArn = task['s3BucketArn']
    s3BucketName = s3BucketArn.split(':::')[-1]

    try:
        # Assume it will succeed for now
        resultCode = 'Succeeded'
        resultString = ''

        # Decode the JSON string that was encoded into the S3 Key value and convert the
        # resulting string into a JSON structure.
        s3Key_decoded = unquote_plus(s3Key)
        keyJson = json.loads(s3Key_decoded)

        # Extract some values from the JSON that we might want to operate on.  In this
 example
        # we won't do anything except return the concatenated string as a fake result.
        newKey = keyJson['newKey']
        origKey = keyJson['origKey']
        resultString = origKey + " --> " + newKey

    except Exception as e:
        # If we run into any exceptions, fail this task so batch operations does not retry
 it and
        # return the exception string so we can see the failure message in the final report
        # created by batch operations.
        resultCode = 'PermanentFailure'
        resultString = 'Exception: {}'.format(e)
    finally:
        # Send back the results for this task.
        results.append({
            'taskId': taskId,
            'resultCode': resultCode,
            'resultString': resultString
        })

    return {
        'invocationSchemaVersion': invocationSchemaVersion,
        'treatMissingKeysAs': 'PermanentFailure',
        'invocationId': invocationId,
        'results': results
    }
```

Put Object ACL

Die Put Object (Objekt ablegen) ACL-Operation ersetzt die Amazon S3 Zugriffskontrolllisten (Access Control Lists, ACLs) für jedes im Manifest aufgeführte Objekt. Mithilfe von ACLs können Sie festlegen, wer auf ein Objekt zugreifen darf, und welche Aktionen die Person ausführen darf.

S3 Stapeloperationen unterstützt benutzerdefinierte ACLs, die von ihnen festgelegt werden, sowie vorgefertigte ACLs, die von Amazon S3 mit vordefinierten Gruppen von Zugriffsberechtigungen bereitgestellt werden.

Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, können Sie die ACLs auf bestimmten Versionen der einzelnen Objekte anwenden. Hierzu geben Sie für jedes Objekt im Manifest eine Versions-ID an. Wenn Sie für ein Objekt keine Versions-ID angeben, wendet S3 Stapeloperationen die ACL auf die aktuelle Version des Objekts an.

> Note
>
> Wenn Sie den öffentlichen Zugriff auf alle Objekte in einem Bucket einschränken möchten, sollten Sie die Amazon S3-Einstellungen zum Sperren des öffentlichen Zugriffs anstatt S3 Stapeloperationen verwenden. „Sperren des öffentlichen Zugriffs“ kann den öffentlichen Zugriff auf einen Bucket oder ein Konto mithilfe einer einzelnen, einfachen Operation einschränken, die sich zudem auch noch schnell umsetzen lässt. Daher ist dies die bessere Wahl, wenn Sie die Absicht haben, den öffentlichen Zugriff auf alle Objekte in einem Bucket oder einem Konto einzuschränken. Verwenden Sie S3 Stapeloperationen, wenn Sie eine benutzerdefinierte ACL auf jedes Objekt im Manifest anwenden möchten. Weitere Informationen zum Blockieren des öffentlichen Zugriffs durch Amazon S3 finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493).

Beschränkungen und Einschränkungen

- Die Rolle, die den Auftrag „Put Object ACL“ ausführen soll, benötigt ausreichende Berechtigungen, um die zugrundeliegende Amazon S3-ACL-Operation „PUT Object“ durchzuführen. Weitere Informationen über die erforderlichen Berechtigungen finden Sie unter PUT Object ACL in der Amazon Simple Storage Service API Reference.
- S3 Stapeloperationen verwendet die Amazon S3-ACL-Operation „PUT Object“, um die angegebene ACL auf jedes Objekt im Manifest anzuwenden. Daher gelten auch alle Einschränkungen und Begrenzungen, die auf die zugrundeliegende ACL-Operation „PUT Object“ zutreffen, auch auf alle S3 Stapeloperationen „PUT Object“-ACL-Aufträge. Weitere Informationen finden Sie im Abschnitt Zugehörige Ressourcen (p. 570) auf dieser Seite.

Zugehörige Ressourcen

- Zugriffsverwaltung mit ACLs (p. 479)
- GET Object ACL in der Amazon Simple Storage Service API Reference

Put Object Tagging

Die Operation Put Object Tagging (Objekt-Tagging setzen) ersetzt die Amazon S3-Objekt-Tags für jedes im Manifest aufgeführte Objekt. Ein Amazon S3-Objekt-Tag ist ein Schlüssel-Wert-Paar, mit dem Sie Metadaten über ein Objekt speichern können.

Um einen Auftrag zum Put Object Tagging zu erstellen, müssen Sie eine Reihe von Tags bereitstellen, die angewendet werden sollen. S3 Stapeloperationen wendet die dieselbe Gruppe von Tags auf jedes Objekt an. Die von Ihnen bereitgestellte Tag-Gruppe ersetzt die Tag-Gruppen, die den Objekten bereits im

Manifest zugewiesen sind. Das Hinzufügen von Tags zu Objekten, während gleichzeitig bereits vorhandene Tags erhalten bleiben, wird von S3 Stapeloperationen nicht unterstützt.

Wenn die Objekte in Ihrem Manifest zu einem versionierten Bucket gehören, können Sie die Tag-Gruppen auf bestimmten Versionen der einzelnen Objekte anwenden. Hierzu geben Sie für jedes Objekt im Manifest eine Versions-ID an. Wenn Sie für ein Objekt keine Versions-ID angeben, wendet S3 Stapeloperationen die Tag-Gruppe auf die aktuelle Version der einzelnen Objekte an.

Beschränkungen und Einschränkungen

- Die Rolle, die den Auftrag „Put Object Tagging“ ausführen soll, benötigt ausreichende Berechtigungen, um die zugrundeliegende Amazon S3 PUT Object Tagging-Operation durchzuführen. Weitere Informationen über die erforderlichen Berechtigungen finden Sie unter PUT Object Tagging in der Amazon Simple Storage Service API Reference.
- S3 Stapeloperationen verwendet die Amazon S3 PUT Object Tagging-Operation, um jedem Objekt im Manifest Tags zuzuweisen. Daher gelten auch alle Einschränkungen und Begrenzungen, die auf die zugrundeliegende PUT Object Tagging-Operation zutreffen, auch auf alle S3 Stapeloperationen PUT Object Tagging-Aufträge. Weitere Informationen finden Sie im Abschnitt Zugehörige Ressourcen (p. 571) auf dieser Seite.

Zugehörige Ressourcen

- Markieren von Objekten (p. 130)
- GET Object Tagging in der Amazon Simple Storage Service API Reference
- PUT Object Tagging in der Amazon Simple Storage Service API Reference

Verwalten von S3 Stapeloperationen-Aufträgen

Amazon S3 stellt robuste Tools bereit, die Sie bei der Verwaltung Ihrer Stapeloperationen-Aufträge unterstützen, nachdem Sie sie erstellt haben. In diesem Abschnitt werden die Operationen beschrieben, mit denen Sie Ihre Aufträge verwalten können. Sie können alle in diesem Abschnitt genannten Operationen mit der AWS Management Console, der AWS CLI, den AWS-SDKs oder den REST-APIs durchführen.

Themen

Im folgenden Video wird kurz beschrieben, wie Sie die Amazon S3-Konsole zum Verwalten Ihrer S3 Stapeloperationen-Aufträge verwenden können.

Auflisten von Aufträgen

Sie können eine Liste Ihrer S3 Stapeloperationen-Aufträge abrufen. Die Liste umfasst Aufträge, die noch nicht abgeschlossen sind, sowie Aufträge, die in den letzten 90 Tagen abgeschlossen wurden. Die

Auftragsliste enthält Informationen zu jedem Auftrag, z. B. die ID, die Beschreibung, die Priorität, den aktuellen Status und die Anzahl an Aufgaben, die erfolgreich durchgeführt wurden oder fehlgeschlagen sind. Sie können die Auftragsliste nach dem Status filtern. Wenn Sie eine Auftragsliste über die Konsole abrufen, können Sie Ihre Aufträge auch nach Beschreibung oder ID durchsuchen und nach AWS Region filtern.

Anzeigen von Auftragsdetails

Wenn Sie mehr Informationen zu einem Auftrag benötigen, als Sie beim Auflisten von Aufträgen erhalten, können Sie alle Details zu einem einzelnen Auftrag anzeigen lassen. Neben den Angaben, die in einer Auftragsliste enthalten sind, umfassen die Details zu einem einzelnen Auftrag auch noch andere Informationen. Dazu gehören die Parameter der Operation, detaillierte Angaben zum Manifest, Informationen zum Abschlussbericht (sofern Sie diesen Bericht bei der Auftragserstellung konfiguriert haben) und den Amazon Resource Name (ARN) der Benutzerrolle, dem Sie die Ausführung des Auftrags zugewiesen haben. Durch das Anzeigen der Details eines einzelnen Auftrags erhalten Sie Zugriff auf die gesamte Konfiguration eines Auftrags.

Steuern von Zugriffs- und Labeling-Aufträgen mithilfe von Tags

Sie können den Zugriff auf Ihre S3 Stapeloperationen-Aufträge bezeichnen und steuern, indem Sie Tags hinzufügen. Anhand von Tags kann identifiziert werden, wer für einen Stapeloperationen-Job verantwortlich ist. Das Vorhandensein von Aufgabe-Tags kann einem Benutzer die Möglichkeit gewähren oder einschränken, eine Aufgabe abzubrechen, eine Aufgabe im Bestätigungsstatus zu aktivieren oder die Prioritätsstufe einer Aufgabe zu ändern. Sie können Aufträge mit ihnen zugeordneten Tags erstellen oder Sie können Aufträgen nach ihrer Erstellung Tags hinzufügen. Jedes Tag ist ein Schlüssel-Wert-Paar, das beim Erstellen des Auftrags oder später aktualisiert werden kann.

> Warning
>
> Auftrags-Tags sollten keine vertraulichen Informationen oder persönlichen Daten enthalten.

Betrachten Sie das folgende Tagging-Beispiel: Angenommen, Ihre Finanzabteilung soll einen Stapeloperationen-Auftrag erstellen. Sie können eine AWS Identity and Access Management (IAM)-Richtlinie schreiben, die es einem Benutzer ermöglicht `CreateJob` aufzurufen, vorausgesetzt, der Auftrag wird mit dem `Department`-Tag erstellt, dem der Wert `Finance` zugewiesen ist. Außerdem können Sie diese Richtlinie an alle Benutzer anfügen, die Mitglieder der Finanzabteilung sind.

Wenn Sie mit diesem Beispiel fortfahren, können Sie eine Richtlinie schreiben, die es einem Benutzer ermöglicht, die Priorität aller Aufträge, die über die gewünschten Tags verfügen, zu aktualisieren oder alle Aufträge, die diese Tags besitzen, abzubrechen. Weitere Informationen finden Sie unter the section called “Job-Tags Beispiele” (p. 577).

Sie können Tags beim Erstellen neuer S3 Stapeloperationen-Aufträge oder zu bereits vorhandenen Aufträgen hinzufügen.

Beachten Sie die folgenden Tag-Einschränkungen:

- Sie können einem Auftrag bis zu 50 Tags zuordnen, solange diese über eindeutige Tag-Schlüssel verfügen.
- Ein Tag-Schlüssel kann maximal 128 Unicode-Zeichen lang sein, und die Tag-Werte können bis zu 256 Unicode-Zeichen lang sein.
- Bei Schlüsseln und Werten wird die Groß-/Kleinschreibung berücksichtigt.

Weitere Informationen zu Tag-Einschränkungen finden Sie unter Einschränkungen benutzerdefinierter Tags im AWS Billing and Cost Management Benutzerhandbuch.

API-Operationen im Zusammenhang mit S3 Stapeloperationen-Auftrags-Tagging

Amazon S3 unterstützt die folgenden API-Vorgänge, die spezifisch für S3 Stapeloperationen-Auftrags-Tagging sind:

- GetJobTagging – Gibt den Tag-Satz zurück, der einem Stapeloperationen-Auftrag zugeordnet ist.
- putJobTagging – Ersetzt den Tag-Satz, der einem Auftrag zugeordnet ist. Es gibt zwei unterschiedliche Szenarien der S3 Stapeloperationen-Auftrags-Tag-Verwaltung unter Verwendung dieser API-Aktion:
  - Auftrag ohne Tags – Sie können einen Satz von Tags zu einem Auftrag hinzufügen (der Auftag hat keine vorherigen Tags).
  - Auftrag verfügt über einen Satz vorhandener Tags – Um den vorhandenen Tag-Satz zu ändern, können Sie entweder den vorhandenen Tag-Satz vollständig ersetzen oder Änderungen innerhalb des vorhandenen Tag-Satzes vornehmen, indem Sie den vorhandenen Tag-Satz mit GetJobTagging abrufen, diesen Tag-Satz ändern und mit dieser API-Aktion diesen Tag-Satz gegen einen von Ihnen modifizierten auswechseln.

    Note

    Wenn Sie diese Anforderung mit einer leeren Tag-Menge senden, löscht S3 Stapeloperationen die vorhandene Tag-Menge für das Objekt. Wenn Sie diese Methode verwenden, wird Ihnen eine Tier 1-Anforderung (PUT) berechnet. Weitere Informationen finden Sie unter Amazon S3 – Preise.
    Wenn Sie vorhandene Tags für Ihren Stapeloperationen-Auftrag löschen möchten, wird die Aktion `DeleteJobTagging` bevorzugt, da sie dasselbe Ergebnis ohne anfallende Gebühren erzielt.
- DeleteJobTagging – Löscht den Tag-Satz, der einem Stapeloperationen-Auftrag zugeordnet ist.

Zuweisen der Auftragspriorität

Sie können jedem Auftrag einen numerischen Prioritätswert zuweisen. Dabei kann es sich um jede positive Ganzzahl handeln. S3 Stapeloperationen priorisiert die Aufträge gemäß den zugewiesenen Prioritäten. Aufträge mit höherer Priorität (oder einem höheren Wert für den Prioritätsparameter) werden zuerst bewertet. Die Priorität wird in absteigender Reihenfolge ermittelt. So erhält beispielsweise eine Auftragswarteschlange mit einem Prioritätswert von 10 Vorrang vor einer Auftragswarteschlange mit dem Wert 1.

Sie können die Priorität eines Auftrags auch während seiner Ausführung ändern. Wenn Sie einen neuen Auftrag übermitteln, während ein Auftrag ausgeführt wird, kann der Auftrag mit niedrigerer Priorität pausieren und der Auftrag mit höherer Priorität ausgeführt werden.

Note

S3 Stapeloperationen erkennt die Auftragsprioritäten auf einer Best-Effort-Basis an. Zwar werden Aufträge mit höheren Prioritäten im Allgemeinen vor Aufträgen mit niedrigeren Prioritäten ausgeführt. Amazon S3 garantiert jedoch keine strikte Reihenfolge der Aufträge.

Sobald Ihr Auftrag erstellt ist, wird das Auftrags-Dashboard geöffnet, wo Sie Ihre Aufträge anzeigen und verwalten können.

Nachdem Sie einen Auftrag erstellt haben, durchläuft dieser eine Reihe von Status. Das folgende Diagramm beschreibt die gängigsten Status eines Auftrags sowie die möglichen Übergänge zwischen den einzelnen Auftragsstatus.

Status	Beschreibung	Übergänge
`New`	Ein Auftrag beginnt nach der Erstellung mit dem Status `New`.	Ein Auftrag wechselt automatisch in den Status `Preparing`, nachdem Amazon S3 mit der Verarbeitung des Manifestobjekts beginnt.
`Preparing`	Amazon S3 verarbeitet das Manifestobjekt und andere Auftragsparameter, um den Auftrag einzurichten und auszuführen.	Ein Auftrag wechselt automatisch in den Status `Ready`, sobald Amazon S3 die Verarbeitung des Manifests und anderer Parameter abschließt. Er ist dann bereit, die angegebene Operation an den im Manifest aufgeführten Objekten auszuführen. Wenn der Auftrag vor der Ausführung bestätigt werden muss, wie dies bei der Erstellung eines Auftrags mit der Amazon S3-Konsole der Fall ist, wechselt der Auftrag von `Preparing` zu `Suspended`. Der Auftrag verbleibt im Status `Suspended`, bis Sie bestätigen, dass Sie ihn ausführen möchten.
`Suspended`	Der Auftrag erfordert eine Bestätigung, aber Sie haben noch nicht bestätigt, dass Sie ihn ausführen möchten. Nur Aufträge, die mit der Amazon S3-Konsole erstellt wurden, benötigen eine Bestätigung. Ein Auftrag, der mit der Konsole erstellt wurde, wechselt nach `Suspended` sofort zu `Preparing`. Nachdem Sie die Ausführung des Auftrags bestätigen und der Auftrag in den Status `Ready` wechselt, kehrt er nicht wieder in den Status `Suspended` zurück.	Nachdem Sie die Ausführung des Auftrags bestätigen, wechselt der Status zu `Ready`.
`Ready`	Amazon S3 ist bereit, die angeforderten Objektoperationen auszuführen.	Ein Auftrag wechselt automatisch in den Status `Active`, nachdem Amazon S3 mit dessen Ausführung beginnt. Der Zeitraum, den ein Auftrag im Status `Ready` verbringt, hängt davon ab, ob bereits Aufträge mit einer höheren Priorität ausgeführt werden und wie lange es dauert, diese Aufträge abzuschließen.

Status	Beschreibung	Übergänge
`Active`	Amazon S3 führt die angeforderte Operation an den im Manifest aufgeführten Objekten aus. Während ein Auftrag `Active` ist, können Sie den Fortschritt über die Amazon S3-Konsole oder anhand der `DescribeJob`-Operation über die REST API, AWS CLI oder AWS-SDKs überwachen.	Ein Auftrag verlässt den Status `Active`, wenn er nicht länger Operationen an Objekten ausführt. Dies kann automatisch erfolgen, wenn ein Auftrag erfolgreich beendet wird oder fehlschlägt. Oder es kann aufgrund von Benutzeraktionen erfolgen, wenn beispielsweise ein Auftrag abgebrochen wird. Der neue Status des Auftrags hängt vom Grund für den Übergang ab.
`Pausing`	Der Auftrag wechselt von `Paused` zu einem anderen Status über.	Ein Auftrag wechselt automatisch in den Status `Paused`, wenn die Stufe `Pausing` abgeschlossen ist.
`Paused`	Ein Auftrag wechselt in den Status `Paused`, wenn ein anderer Auftrag mit einer höheren Priorität abgesendet wird, während der aktuelle Auftrag ausgeführt wird.	Ein Auftrag mit dem Status `Paused` kehrt automatisch in den Status `Active` zurück, nachdem alle Aufträge mit einer höheren Priorität, die die Ausführung des Auftrag blockierten, abgeschlossen, fehlgeschlagen oder gesperrt sind.
`Complete`	Der Auftrag hat die Ausführung der angeforderten Operation an allen Objekten abgeschlossen. Die Operation kann bei den einzelnen Objekten entweder erfolgreich abgeschlossen oder fehlgeschlagen sein. Wenn Sie den Auftrag so konfiguriert haben, dass ein Abschlussbericht generiert werden soll, dann ist der Bericht sofort verfügbar, nachdem der Auftrag in den Status `Complete` gewechselt ist.	`Complete` ist ein Beendigungsstatus. Sobald ein Auftrag den Status `Complete` erreicht, geht er in keinen anderen Status über.
`Cancelling`	Der Auftrag wechselt in den Status `Cancelled`.	Ein Auftrag wechselt automatisch in den Status `Cancelled`, wenn die Stufe `Cancelling` abgeschlossen ist.
`Cancelled`	Sie haben angefordert, dass der Auftrag abgebrochen wird und S3 Stapeloperationen hat den Auftrag erfolgreich abgebrochen. Der Auftrag sendet keine neuen Anforderungen an Amazon S3.	`Cancelled` ist ein Beendigungsstatus. Nachdem ein Auftrag den Status `Cancelled` erreicht, geht er in keinen anderen Status über.

Status	Beschreibung	Übergänge
`Failing`	Der Auftrag wechselt in den Status `Failed`.	Ein Auftrag wechselt automatisch in den Status `Failed`, nachdem die Stufe `Failing` abgeschlossen ist.
`Failed`	Der Auftrag ist fehlgeschlagen und wird nicht länger ausgeführt. Weitere Informationen zu Auftragsfehlern finden Sie unter Nachverfolgen von Auftragsfehlern (p. 576).	`Failed` ist ein Beendigungsstatus. Nachdem ein Auftrag den Status `Failed` erreicht, geht er in keinen anderen Status über.

Nachverfolgen von Auftragsfehlern

Wenn bei einem S3 Stapeloperationen-Auftrag ein Problem auftritt, das die erfolgreiche Ausführung verhindert, schlägt der Auftrag fehl. Dies ist beispielsweise der Fall, wenn das angegebene Manifest nicht gelesen werden kann. Wenn ein Auftrag fehlschlägt, erzeugt er mindestens einen Fehlercode oder eine Fehlerursache. S3 Stapeloperationen speichert die Fehlercodes und -ursachen zusammen mit dem Auftrag, sodass Sie sie anzeigen lassen können, indem Sie die Auftragsdetails abrufen. Wenn Sie einen Abschlussbericht für den Auftrag angefordert haben, sind die Fehlercodes und -ursachen auch darin enthalten.

Um zu verhindern, dass zu viele Operationen fehlschlagen, wendet Amazon S3 einen Schwellenwert für fehlgeschlagene Aufträge auf jeden Stapeloperationen-Auftrag an. Wenn ein Auftrag mindestens 1.000 Aufgaben ausgeführt hat, überwacht Amazon S3 die Aufgabenfehlerrate. Wenn die Fehlerrate (also die Anzahl fehlgeschlagener Aufgaben im Verhältnis zur Gesamtzahl ausgeführter Aufgaben) zu einem beliebigen Zeitpunkt 50 Prozent überschreitet, schlägt der Auftrag fehl. Wenn Ihr Auftrag fehlschlägt, weil er den Schwellenwert für fehlgeschlagene Aufträge überschritten hat, können Sie die Ursache der Fehler identifizieren. Beispielsweise könnte der Fehler darin liegen, dass Sie versehentlich Objekte in das Manifest aufgenommen haben, die im angegebenen Bucket nicht vorhanden sind. Nach dem Beheben der Fehler können Sie den Auftrag erneut übermitteln.

> Note
>
> S3 Stapeloperationen wird asynchron betrieben und die Aufträge werden nicht zwangsläufig in der Reihenfolge ausführt, in der die Objekte im Manifest aufgeführt sind. Daher können Sie anhand der Reihenfolge im Manifest nicht feststellen, welche Objektaufgaben erfolgreich ausgeführt wurden und welche fehlgeschlagen sind. Um dies zu ermitteln, können Sie (sofern Sie ihn angefordert haben) den Abschlussbericht zu Ihrem Auftrag nutzen oder ihre AWS CloudTrail-Ereignisprotokolle anzeigen lassen, um so die Fehlerursache zu finden.

Benachrichtigungen und Protokollierung

Sie können nicht nur Abschlussberichte anfordern, sondern haben auch die Möglichkeit, die Stapeloperationen-Aktivitäten mithilfe der AWS CloudTrail zu erfassen, überprüfen und überwachen. Da Stapeloperationen vorhandene Amazon S3-APIs für die Durchführung von Aufgaben nutzen, geben diese Aufgaben dieselben Ereignisse aus, die sie ausgeben würden, wenn Sie sie direkt aufgerufen hätten. So können Sie den Fortschritt Ihres Auftrags und all seiner Aufgaben mit denselben Benachrichtigungs-, Protokollierungs- und Prüftools und -prozessen nachverfolgen und aufzeichnen, die Sie bereits mit Amazon S3 verwenden.

Weitere Informationen über Amazon S3-Ereignisse finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).

Abschlussberichte

Wenn Sie einen Auftrag erstellen, können Sie einen Abschlussbericht anfordern. Solange S3 Stapeloperationen mindestens eine Aufgabe erfolgreich aufruft, generiert Amazon S3 einen Abschlussbericht, wenn die Ausführung von Aufträgen abgeschlossen, fehlgeschlagen oder abgebrochen wurde. Sie können den Abschlussbericht so konfigurieren, dass er alle oder nur fehlgeschlagene Aufgaben enthält.

Der Abschlussbericht umfasst neben der Auftragskonfiguration und dem Auftragsstatus auch Informationen zu jeder Aufgabe, darunter zu Objektschlüssel und -version, Status, Fehlercodes und Beschreibungen von Fehlern. Wenn Sie keinen Abschlussbericht konfigurieren, können Sie Ihren Auftrag und dessen Aufgaben mit CloudTrail und Amazon CloudWatch dennoch überwachen und prüfen. Abschlussberichte bieten allerdings eine einfache Möglichkeit, die Ergebnisse von Aufgaben in einem konsolidierten Format ohne zusätzliche Einrichtung anzuzeigen. Ein Beispiel für einen Abschlussbericht finden Sie unter Beispiel: Anfordern von S3 Stapeloperationen-Abschlussberichten (p. 582).

Beispiele für S3 Stapeloperationen

Sie können S3 Stapeloperationen verwenden, um große S3 Stapeloperationen für Milliarden von S3-Objekten ausführen, die Exabyte von Daten enthalten. Sie können Ihre Ihre Stapeloperationen-Aufträge mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDKs oder REST API verwalten.

Dieser Abschnitt enthält die folgenden Beispiele zum Erstellen und Verwalten von Stapeloperationen-Aufträgen in Amazon S3.

Themen

Beispiel: Verwenden von Auftrags-Tags zum Steuern von Berechtigungen für S3 Stapeloperationen

Um Ihnen bei der Verwaltung Ihrer S3 Stapeloperationen-Aufträge zu helfen, können Sie Auftags-Tags hinzufügen. Mit Auftrags-Tags können Sie den Zugriff auf Ihre Stapeloperationen-Aufträge steuern und erzwingen, dass Tags angewendet werden, wenn ein Auftrag erstellt wird.

Sie können bis zu 50 Auftrag-Tags auf jeden Stapeloperationen-Auftrag anwenden. Auf diese Weise können Sie sehr detaillierte Richtlinien festlegen, die die Gruppe von Benutzern einschränken, die den Auftrag bearbeiten können. Aufgaben-Tags können einem Benutzer die Möglichkeit gewähren oder einschränken, eine Aufgabe abzubrechen, eine Aufgabe im Bestätigungsstatus zu aktivieren oder die Prioritätsstufe einer Aufgabe zu ändern. Darüber hinaus können Sie erzwingen, dass Tags auf alle neuen Aufträge angewendet werden, und die zulässigen Schlüssel-Wert-Paare für die Tags angeben. Sie können alle diese Bedingungen mit derselben IAM-Richtliniensprache ausdrücken. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon S3 (p. 394).

Das folgende Beispiel zeigt, wie Sie S3 Stapeloperationen-Auftrags-Tags verwenden können, um Benutzern die Berechtigung zu erteilen, nur die Jobs zu erstellen und zu bearbeiten, die in einer bestimmten Abteilung (z. B. der Abteilung Finanzen oder Compliance) ausgeführt werden. Sie können Aufträge auch basierend auf der Entwicklungsphase zuweisen, auf die sie sich beziehen, z. B QS oder Produktion.

In diesem Beispiel verwenden Sie S3 Stapeloperationen-Auftrags-Tags in AWS Identity and Access Management (IAM)-Richtlinien, um Benutzern die Berechtigung zu erteilen, nur die Auftrage zu erstellen und zu bearbeiten, die in ihrer Abteilung ausgeführt werden. Sie weisen Aufträge basierend auf der Entwicklungsphase zu, mit der sie verbunden sind, wie QS oder Produktion.

In diesem Beispiel werden die folgenden Abteilungen verwendet, die Stapeloperationen auf jeweils unterschiedliche Weise einsetzen:

- Finanzen
- Compliance
- Business Intelligence
- Entwicklung

Steuern des Zugriffs durch Zuweisen von Tags zu Benutzern und Ressourcen

In diesem Szenario verwenden die Administratoren attributbasierte Zugriffssteuerung (ABAC). ABAC ist eine IAM-Autorisierungsstrategie, die Berechtigungen definiert, indem Tags sowohl an IAM-Benutzer als auch an AWS-Ressourcen angehängt werden.

Benutzern und Jobs wird eines der folgenden Abteilungstags zugewiesen:

Schlüssel : Wert

- `department : Finance`
- `department : Compliance`
- `department : BusinessIntelligence`
- `department : Engineering`

> **Note**
>
> Bei Tag-Schlüsseln und -Werten muss die Groß-/Kleinschreibung beachtet werden.

Mit der ABAC-Zugriffssteuerungsstrategie erteilen Sie einem Benutzer in der Finanzabteilung die Berechtigung, S3 Stapeloperationen-Aufträge innerhalb seiner Abteilung zu erstellen und zu verwalten, indem Sie das Tag `department=Finance` mit dem IAM-Benutzer verknüpfen.

Darüber hinaus können Sie dem IAM-Benutzer eine verwaltete Richtlinie anfügen, die es jedem Benutzer in seinem Unternehmen ermöglicht, S3 Stapeloperationen-Aufträge innerhalb seiner jeweiligen Abteilungen zu erstellen oder zu ändern.

Die Richtlinie in diesem Beispiel enthält drei Richtlinienanweisungen:

- Die erste Anweisung in der Richtlinie ermöglicht es dem Benutzer, einen Stapeloperationen-Auftrag zu erstellen, vorausgesetzt, die Auftragserstellungsanforderung enthält ein Auftrags-Tag, das der jeweiligen Abteilung entspricht. Dies wird mithilfe der Syntax `"${aws:PrincipalTag/department}"` ausgedrückt, die zum Zeitpunkt der Richtlinienauswertung durch das Abteilungs-Tag des IAM-Benutzers ersetzt wird. Die Bedingung ist erfüllt, wenn der für das Abteilungs-Tag in der Anforderung (`"aws:RequestTag/department"`) angegebene Wert mit dem der Abteilung des Benutzers übereinstimmt.

- Die zweite Anweisung in der Richtlinie ermöglicht es Benutzern, die Priorität von Aufträgen zu ändern oder den Status eines Auftrags zu aktualisieren, sofern der Auftrag, den der Benutzer aktualisiert, mit der Abteilung des Benutzers übereinstimmt.
- Die dritte Anweisung ermöglicht es einem Benutzer, die Tags eines Stapeloperationen-Auftrags jederzeit über eine `PutJobTagging`-Anforderung zu aktualisieren, solange (1) ihr Abteilung-Tag erhalten bleibt und (2) sich der Auftrag, den sie aktualisieren, innerhalb ihrer Abteilung befindet.

```
{
      "Version": "2012-10-17",
      "Statement": [
            {
                  "Effect": "Allow",
                  "Action": "s3:CreateJob",
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:RequestTag/department": "${aws:PrincipalTag/department}"

                }
            }
        },
            {
                  "Effect": "Allow",
                  "Action": [
                        "s3:UpdateJobPriority",
                        "s3:UpdateJobStatus"
            ],
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        },
            {
                  "Effect": "Allow",
                  "Action": "s3:PutJobTagging",
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:RequestTag/department": "${aws:PrincipalTag/
department}",
                              "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        }
    ]
}
```

Markieren von Stapeloperationen-Auftragen nach Stufe und Durchsetzen von Limits für die Auftrags-Priorität

Alle S3 Stapeloperationen-Aufträge haben eine numerische Priorität, anhand der Amazon S3 entscheidet, in welcher Reihenfolge sie auszuführen sind. In diesem Beispiel beschränken Sie die maximale Priorität, die die meisten Benutzer Aufträgen zuweisen können, wobei höhere Prioritätsbereiche für eine begrenzte Gruppe von berechtigten Benutzern reserviert sind:

- Prioritätsbereich der QA-Stufe (niedrig): 1-100

- Prioritätsbereich der Produktionsstufe (hoch): 1-300

Dazu führen Sie einen neuen Tag-Satz ein, der die Phase des Auftrags repräsentiert:

Schlüssel : Wert

- `stage : QA`
- `stage : Production`

Erstellen und Aktualisieren von Aufträgen mit geringer Priorität in einer Abteilung

Mit dieser Richtlinie werden neben der abteilungsbasierten Einschränkung zwei neue Beschränkungen für die Erstellung und Aktualisierung von S3 Stapeloperationen-Arbeitsplätzen eingeführt:

- Es ermöglicht Benutzern, Aufträge in ihrer Abteilung mit einer neuen Bedingung zu erstellen oder zu aktualisieren, die erfordert, dass der Auftrag das Tag `stage=QA` enthält.
- Es ermöglicht Benutzern, Aufträge mit einer maximalen Priorität von bis 100 zu erstellen oder zu aktualisieren.

```
{
        "Version": "2012-10-17",
        "Statement": [
        {
        "Effect": "Allow",
        "Action": "s3:CreateJob",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/department": "${aws:PrincipalTag/department}",
                "aws:RequestTag/stage": "QA"
            },
            "NumericLessThanEquals": {
                "s3:RequestJobPriority": 100
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:UpdateJobStatus"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:UpdateJobPriority",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}",
                "aws:ResourceTag/stage": "QA"
            },
            "NumericLessThanEquals": {
                "s3:RequestJobPriority": 100
            }
```

```
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:PutJobTagging",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/department" : "${aws:PrincipalTag/department}",
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}",
                "aws:RequestTag/stage": "QA",
                "aws:ResourceTag/stage": "QA"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:GetJobTagging",
        "Resource": "*"
    }
    ]
}
```

Erstellen und Aktualisieren von Aufträgen mit hoher Priorität in einer Abteilung

Einer kleinen Anzahl von Benutzern muss u. U. möglich sein, Aufträge hoher Priorität in QS oder Produktion zu erstellen. Um diesen Bedarf zu unterstützen, erstellen Sie eine verwaltete Richtlinie durch Abwandlung einer Richtlinie mit niedriger Priorität im vorherigen Abschnitt.

Diese Richtlinie gewährt die folgenden Aktionen:

- Ermöglicht Benutzern, Aufträge in ihrer Abteilung entweder mit dem Tag `stage=QA` oder `stage=Production` zu erstellen oder zu aktualisieren.
- Ermöglicht Benutzern, bei der Erstellung oder Aktualisierung eines Auftrags als Priorität bis zu 300 anzugeben.

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
                "Effect": "Allow",
                "Action": "s3:CreateJob",
                "Resource": "*",
                "Condition": {
                      "ForAnyValue:StringEquals": {
                            "aws:RequestTag/stage": [
                                  "QA",
                                  "Production"
                    ]
                },
                      "StringEquals": {
                            "aws:RequestTag/department": "${aws:PrincipalTag/department}"

                },
                      "NumericLessThanEquals": {
                            "s3:RequestJobPriority": 300
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": [
```

```
                    "s3:UpdateJobStatus"
            ],
                "Resource": "*",
                "Condition": {
                      "StringEquals": {
                            "aws:ResourceTag/department": "${aws:PrincipalTag/department}"

                }
            }
        },
          {
                "Effect": "Allow",
                "Action": "s3:UpdateJobPriority",
                "Resource": "*",
                "Condition": {
                      "ForAnyValue:StringEquals": {
                            "aws:ResourceTag/stage": [
                                  "QA",
                                  "Production"
                        ]
                },
                      "StringEquals": {
                            "aws:ResourceTag/department": "${aws:PrincipalTag/department}"

                },
                      "NumericLessThanEquals": {
                            "s3:RequestJobPriority": 300
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": "s3:PutJobTagging",
                "Resource": "*",
                "Condition": {
                      "StringEquals": {
                            "aws:RequestTag/department": "${aws:PrincipalTag/department}",
                            "aws:ResourceTag/department": "${aws:PrincipalTag/department}"

                },
                      "ForAnyValue:StringEquals": {
                            "aws:RequestTag/stage": [
                                  "QA",
                                  "Production"
                        ],
                            "aws:ResourceTag/stage": [
                                  "QA",
                                  "Production"
                        ]
                }
            }
        }
    ]
}
```

Beispiel: Anfordern von S3 Stapeloperationen-Abschlussberichten

Wenn Sie einen S3 Stapeloperationen-Auftrag erstellen, können Sie einen Abschlussbericht für alle oder nur für die fehlgeschlagenen Aufgaben anfordern. Solange mindestens eine Aufgabe erfolgreich aufgerufen wurde, generiert S3 Stapeloperationen einen Bericht für abgeschlossene, fehlgeschlagene oder stornierte Aufträge.

Der Abschlussbericht enthält zusätzliche Informationen zu jeder Aufgabe, darunter den Namen und die Version des Objektschlüssels, Status, Fehlercodes sowie Beschreibungen zu etwaigen Fehlern. Die Beschreibung der Fehler für jede fehlgeschlagene Aufgabe kann herangezogen werden, um Probleme im Rahmen der Auftragserstellung zu diagnostizieren, beispielsweise mit Berechtigungen.

Example – Manifest-Ergebnisdatei der höchsten Ebene

Die `manifest.json`-Datei der höchsten Ebene enthält die Position jedes Erfolgsberichts und (sofern Fehler in der Auftragsabwicklung aufgetreten sind) jedes Fehlerberichts (siehe folgendes Beispiel).

```
{
    "Format": "Report_CSV_20180820",
    "ReportCreationDate": "2019-04-05T17:48:39.725Z",
    "Results": [
        {
            "TaskExecutionStatus": "succeeded",
            "Bucket": "my-job-reports",
            "MD5Checksum": "83b1c4cbe93fc893f54053697e10fd6e",
            "Key": "job-f8fb9d89-a3aa-461d-bddc-ea6a1b131955/
results/6217b0fab0de85c408b4be96aeaca9b195a7daa5.csv"
        },
        {
            "TaskExecutionStatus": "failed",
            "Bucket": "my-job-reports",
            "MD5Checksum": "22ee037f3515975f7719699e5c416eaa",
            "Key": "job-f8fb9d89-a3aa-461d-bddc-ea6a1b131955/results/
b2ddad417e94331e9f37b44f1faf8c7ed5873f2e.csv"
        }
    ],
    "ReportSchema": "Bucket, Key, VersionId, TaskStatus, ErrorCode, HTTPStatusCode,
 ResultMessage"
}
```

Example – Berichte zu fehlgeschlagenen Aufgaben

Berichte zu fehlgeschlagenen Aufgaben enthalten die folgenden Informationen für alle fehlgeschlagenen Aufgaben:

- `Bucket`
- `Key`
- `VersionId`
- `TaskStatus`
- `ErrorCode`
- `HTTPStatusCode`
- `ResultMessage`

Der folgende Beispielbericht zeigt eine Zeitüberschreitung der AWS Lambda-Funktion, die dazu führt, dass die Anzahl der Fehler den Fehlergrenzwert überschreitet. Deshalb wurde die Kennzeichnung `PermanentFailure` zugewiesen.

```
awsexamplebucket1,image_14975,,failed,200,PermanentFailure,"Lambda returned function error:
 {""errorMessage"":""2019-04-05T17:35:21.155Z 2845ca0d-38d9-4c4b-abcf-379dc749c452 Task
 timed out after 3.00 seconds""}"
awsexamplebucket1,image_15897,,failed,200,PermanentFailure,"Lambda returned function error:
 {""errorMessage"":""2019-04-05T17:35:29.610Z 2d0a330b-de9b-425f-b511-29232fde5fe4 Task
 timed out after 3.00 seconds""}"
awsexamplebucket1,image_14819,,failed,200,PermanentFailure,"Lambda returned function error:
 {""errorMessage"":""2019-04-05T17:35:22.362Z fcf5efde-74d4-4e6d-b37a-c7f18827f551 Task
 timed out after 3.00 seconds""}"
```

```
awsexamplebucket1,image_15930,,failed,200,PermanentFailure,"Lambda returned function error:
 {""errorMessage"":""2019-04-05T17:35:29.809Z 3dd5b57c-4a4a-48aa-8a35-cbf027b7957e Task
 timed out after 3.00 seconds""}"
awsexamplebucket1,image_17644,,failed,200,PermanentFailure,"Lambda returned function error:
 {""errorMessage"":""2019-04-05T17:35:46.025Z 10a764e4-2b26-4d8c-9056-1e1072b4723f Task
 timed out after 3.00 seconds""}"
awsexamplebucket1,image_17398,,failed,200,PermanentFailure,"Lambda returned function error:
 {""errorMessage"":""2019-04-05T17:35:44.661Z 1e306352-4c54-4eba-aee8-4d02f8c0235c Task
 timed out after 3.00 seconds""}"
```

Example – Bericht zu erfolgreichen Aufgaben

Berichte zu erfolgreichen Aufgaben enthalten für alle abgeschlossenen Aufgaben Folgendes:

- `Bucket`
- `Key`
- `VersionId`
- `TaskStatus`
- `ErrorCode`
- `HTTPStatusCode`
- `ResultMessage`

Im folgenden Beispiel hat die Lambda-Funktion das Amazon S3-Objekt in einen anderen Bucket kopiert. Die zurückgegebene Amazon S3-Antwort wird an S3 Stapeloperationen zurückgegeben und dann in den endgültigen Abschlussbericht geschrieben.

```
awsexamplebucket1,image_17775,,succeeded,200,,"{u'CopySourceVersionId':
 'xVR78haVKlRnurYofbTfYr3ufYbktF8h', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 39, tzinfo=tzlocal()), u'ETag':
 '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata': {'HTTPStatusCode':
 200, 'RetryAttempts': 0, 'HostId': 'nXNaClIMxEJzWNmeMNQV2KpjbaCJLn0OGoXWZpuVOFS/
iQYWxb3QtTvzX9SVfx2lA3oTKLwImKw=', 'RequestId': '3ED5852152014362', 'HTTPHeaders':
 {'content-length': '234', 'x-amz-id-2': 'nXNaClIMxEJzWNmeMNQV2KpjbaCJLn0OGoXWZpuVOFS/
iQYWxb3QtTvzX9SVfx2lA3oTKLwImKw=', 'x-amz-copy-source-version-id':
 'xVR78haVKlRnurYofbTfYr3ufYbktF8h', 'server': 'AmazonS3', 'x-amz-request-id':
 '3ED5852152014362', 'date': 'Fri, 05 Apr 2019 17:35:39 GMT', 'content-type': 'application/
xml'}}}"
awsexamplebucket1,image_17763,,succeeded,200,,"{u'CopySourceVersionId':
 '6HjOUSim4Wj6BTcbxToXW44pSZ.40pwq', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 39, tzinfo=tzlocal()),
 u'ETag': '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata':
 {'HTTPStatusCode': 200, 'RetryAttempts': 0, 'HostId': 'GiCZNYr8LHd/
Thyk6beTRP96IGZk2sYxujLe13TuuLpq6U2RD3we0YoluuIdm1PRvkMwnEW1aFc=', 'RequestId':
 '1BC9F5B1B95D7000', 'HTTPHeaders': {'content-length': '234', 'x-amz-id-2': 'GiCZNYr8LHd/
Thyk6beTRP96IGZk2sYxujLe13TuuLpq6U2RD3we0YoluuIdm1PRvkMwnEW1aFc=', 'x-amz-copy-source-
version-id': '6HjOUSim4Wj6BTcbxToXW44pSZ.40pwq', 'server': 'AmazonS3', 'x-amz-request-id':
 '1BC9F5B1B95D7000', 'date': 'Fri, 05 Apr 2019 17:35:39 GMT', 'content-type': 'application/
xml'}}}"
awsexamplebucket1,image_17860,,succeeded,200,,"{u'CopySourceVersionId':
 'm.MDD0g_QsUnYZ8TBzVFrp.TmjN8PJyX', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 40, tzinfo=tzlocal()), u'ETag':
 '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata': {'HTTPStatusCode':
 200, 'RetryAttempts': 0, 'HostId': 'F9ooZOgpE5g9sNgBZxjdiPHqB4+0DNWgj3qbsir
+sKai4fv7rQEcF2fBN1VeeFc2WH45a9ygb2g=', 'RequestId': '8D9CA56A56813DF3', 'HTTPHeaders':
 {'content-length': '234', 'x-amz-id-2': 'F9ooZOgpE5g9sNgBZxjdiPHqB4+0DNWgj3qbsir
+sKai4fv7rQEcF2fBN1VeeFc2WH45a9ygb2g=', 'x-amz-copy-source-version-id':
 'm.MDD0g_QsUnYZ8TBzVFrp.TmjN8PJyX', 'server': 'AmazonS3', 'x-amz-request-id':
 '8D9CA56A56813DF3', 'date': 'Fri, 05 Apr 2019 17:35:40 GMT', 'content-type': 'application/
xml'}}}"
```

Beispiel: Kopieren von Objekten über AWS-Konten mit S3 Stapeloperationen

Sie können S3 Stapeloperationen verwenden, um einen PUT-Kopierauftrag zu erstellen, mit dem Objekte in ein anderes AWS-Konto (Zielkonto) kopiert werden sollen. Hierzu können Sie mit Amazon S3 Inventory den Bestandsbericht dem Zielkonto zur Verwendung bei der Auftragserstellung bereitstellen. Sie können auch ein CSV-Manifest (kommagetrennte Werte) im Quell- oder Zielkonto verwenden.

Die folgenden Abschnitte erläutern, wie ein Manifest gespeichert und genutzt werden kann, das sich in einem anderen AWS-Konto befindet.

Themen

Verwenden eines im Zielkonto bereitgestellten Bestandsberichts

Der Amazon S3-Bestand generiert Bestandslisten der Objekte in einem Bucket. Die resultierende Liste wird in einer Ausgabedatei veröffentlicht. Der Bucket, für den die Bestandsliste erstellt wird, wird als Quell-Bucket bezeichnet, der Bucket, in dem die Bestandsberichtsdatei gespeichert wird, dagegen als Ziel-Bucket.

Der Amazon S3-Bestandsbericht kann so konfiguriert werden, dass er an ein anderes AWS-Konto übermittelt wird. S3 Stapeloperationen kann den Bestandsbericht dann lesen, wenn der Auftrag im AWS-Zielkonto erstellt wird.

Weitere Informationen zu Quell- und Ziel-Buckets für Amazon S3-Bestände finden Sie unter Wie richte ich Amazon S3 Inventory ein? (p. 504).

Die einfachste Möglichkeit, einen Bestand einzurichten, ist die Verwendung der AWS Management Console, Sie können aber auch REST-API, AWS Command Line Interface (AWS CLI) oder AWS SDKs verwenden.

Die folgende Konsolenprozedur enthält die High-Level-Schritte zum Einrichten von Berechtigungen für einen S3 Stapeloperationen-Auftrag. Bei diesem Verfahren kopieren Sie Objekte aus einem Quellkonto in ein Zielkonto, wobei der Bestandsbericht im AWS-Zielkonto gespeichert wird.

So richten Sie einen Amazon S3-Bestand für Quell- und Ziel-Buckets ein, die zu unterschiedlichen Konten gehören

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie einen Ziel-Bucket, in dem der Bestandsbericht gespeichert werden soll.

   Legen Sie einen Ziel-Manifest-Bucket fest, in dem der Bestandsbericht gespeichert werden soll. In diesem Verfahren ist das Zielkonto das Konto, zu dem der Ziel-Manifest-Bucket sowie der Bucket gehören, in den die Objekte kopiert werden.
3. Konfigurieren Sie eine Bestandsliste, um die Objekte in einem Quell-Bucket aufzulisten und die Liste in einem Ziel-Bucket zu veröffentlichen.

   Konfigurieren Sie eine Bestandsliste für einen Quell-Bucket. Wenn Sie dies tun, geben Sie den Ziel-Bucket an, in dem die Liste gespeichert werden soll. Der Bestandsbericht für den Quell-Bucket wird im Ziel-Bucket veröffentlicht. In diesem Verfahren ist das Quellkonto das Konto, zu dem der Quell-Bucket gehört.

Informationen zur Verwendung der Konsole zum Konfigurieren eines Bestands finden Sie unter Wie konfiguriere ich Amazon S3 Inventory? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Wählen Sie CSV als Ausgabeformat.

Wenn Sie Informationen für den Ziel-Bucket eingeben, wählen Sie Buckets in another account (Buckets in einem anderen Konto). Geben Sie dann den Namen des Ziel-Manifest-Buckets ein. Optional können Sie die Konto-ID des Zielkontos eingeben.

Nachdem die Bestandskonfiguration gespeichert wurde, zeigt die Konsole eine Meldung wie die folgende an:

Amazon S3 could not create a bucket policy on the destination bucket. Ask the destination bucket owner to add the following bucket policy to allow Amazon S3 to place data in that bucket.

Die Konsole zeigt dann eine Bucket-Richtlinie an, die Sie für den Ziel-Bucket verwenden können.

4. Kopieren Sie die in der Konsole angezeigte Ziel-Bucket-Richtlinie.
5. Fügen Sie die kopierte Bucket-Richtlinie im Zielkonto dem Ziel-Manifest-Bucket hinzu, in dem der Bestandsbericht gespeichert wird.
6. Erstellen Sie eine Rolle im Zielkonto, die auf der S3 Stapeloperationen-Vertrauensrichtlinie basiert. Weitere Informationen zur Vertrauensrichtlinie siehe Vertrauensrichtlinie (p. 557).

   Weitere Informationen zum Erstellen einer Rolle finden Sie im Abschnitt zum Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

   Geben Sie einen Namen für die Rolle ein (die Beispielrolle hat den Namen `BatchOperationsDestinationRoleCOPY`). Wählen Sie den Service S3 und dann den Anwendungsfall S3 Bucket Stapeloperationen aus, mit dem die Vertrauensrichtlinie der Rolle zugewiesen wird.

   Wählen Sie dann Create policy (Richtlinie erstellen) aus, um der Rolle die folgende Richtlinie anzufügen.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "AllowBatchOperationsDestinationObjectCOPY",
                "Effect": "Allow",
                "Action": [
                    "s3:PutObject",
                    "s3:PutObjectVersionAcl",
                    "s3:PutObjectAcl",
                    "s3:PutObjectVersionTagging",
                    "s3:PutObjectTagging",
                    "s3:GetObject",
                    "s3:GetObjectVersion",
                    "s3:GetObjectAcl",
                    "s3:GetObjectTagging",
                    "s3:GetObjectVersionAcl",
                    "s3:GetObjectVersionTagging"
                ],
                "Resource": [
                    "arn:aws:s3:::ObjectDestinationBucket/*",
                    "arn:aws:s3:::ObjectSourceBucket/*",
                    "arn:aws:s3:::ObjectDestinationManifestBucket/*"
                ]
            }
        ]
```

```
}
```

Die Rolle verwendet die Richtlinie, um die Berechtigung `batchoperations.s3.amazonaws.com` zum Lesen des Manifests im Ziel-Bucket zu gewähren. Zudem werden Berechtigungen für GET-Objekte, Zugriffskontrolllisten (ACLs), Tags und Versionen im Quell-Bucket der Objekte gewährt. Und es werden Berechtigungen für PUT-Objekte, ACLs, Tags sowie Versionen im Ziel-Bucket der Objekte gewährt.

7. Erstellen Sie im Quellkonto eine Bucket-Richtlinie für den Quell-Bucket, die der im vorherigen Schritt erstellten Rolle Berechtigungen für GET-Objekte, ACLs, Tags und Versionen im Quell-Bucket gewährt. Dieser Schritt ermöglicht es S3 Stapeloperationen, Objekte über die vertrauenswürdige Rolle aus dem Quell-Bucket abzurufen.

   Das folgende Beispiel zeigt eine Bucket-Richtlinie für das Quellkonto.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowBatchOperationsSourceObjectCOPY",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging"
            ],
            "Resource": "arn:aws:s3:::ObjectSourceBucket/*"
        }
    ]
}
```

8. Sobald der Bestandsbericht verfügbar ist, erstellen Sie einen S3 Stapeloperationen-PUT-Auftrag zum Kopieren von Objekten im Zielkonto und wählen den Bestandsbericht im Ziel-Manifest-Bucket aus. Sie benötigen den ARN der im Zielkonto erstellten Rolle.

   Allgemeine Informationen zum Erstellen eines Auftrags siehe Erstellen eines S3 Stapeloperationen-Auftrags (p. 554).

   Informationen zum Erstellen eines Auftrags mit der Konsole finden Sie unter Erstellen eines S3 Stapeloperationen-Auftrags im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Verwenden eines im Quellkonto gespeicherten CSV-Manifests

Sie können eine in einem anderen AWS-Konto gespeicherte CSV-Datei als Manifest für einen S3 Stapeloperationen-Auftrag verwenden.

Das folgende Verfahren zeigt, wie Berechtigungen bei Verwendung eines S3 Stapeloperationen-Auftrags zum Kopieren von Objekten aus einem Quellkonto in ein Zielkonto unter Verwendung der CSV-Manifestdatei, die im Quellkonto gespeichert ist, eingerichtet werden.

So richten Sie ein in einem anderen AWS-Konto gespeichertes CSV-Manifest ein

1. Erstellen Sie eine Rolle im Zielkonto, die auf der S3 Stapeloperationen-Vertrauensrichtlinie basiert. In diesem Verfahren ist das Zielkonto das Konto, in das die Objekte kopiert werden.

   Weitere Informationen zur Vertrauensrichtlinie siehe Vertrauensrichtlinie (p. 557).

   Weitere Informationen zum Erstellen einer Rolle finden Sie im Abschnitt Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

   Wenn Sie die Rolle mit der Konsole erstellen, geben Sie einen Namen für die Rolle ein (der Name der Beispielrolle lautet `BatchOperationsDestinationRoleCOPY`). Wählen Sie den Service S3 und dann den Anwendungsfall S3 Bucket Batch Operations (S3-Bucket-Stapeloperationen) aus, mit dem die Vertrauensrichtlinie der Rolle zugewiesen wird.

   Wählen Sie dann Create policy (Richtlinie erstellen) aus, um der Rolle die folgende Richtlinie anzufügen.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "AllowBatchOperationsDestinationObjectCOPY",
                "Effect": "Allow",
                "Action": [
                    "s3:PutObject",
                    "s3:PutObjectVersionAcl",
                    "s3:PutObjectAcl",
                    "s3:PutObjectVersionTagging",
                    "s3:PutObjectTagging",
                    "s3:GetObject",
                    "s3:GetObjectVersion",
                    "s3:GetObjectAcl",
                    "s3:GetObjectTagging",
                    "s3:GetObjectVersionAcl",
                    "s3:GetObjectVersionTagging"
                ],
                "Resource": [
                    "arn:aws:s3:::ObjectDestinationBucket/*",
                    "arn:aws:s3:::ObjectSourceBucket/*",
                    "arn:aws:s3:::ObjectSourceManifestBucket/*"
                ]
            }
        ]
}
```

   Mit der Richtlinie gewährt die Rolle die Berechtigung `batchoperations.s3.amazonaws.com` zum Lesen des Manifests im Quell-Manifest-Bucket. Sie gewährt Berechtigungen für GET-Objekte, Zugriffskontrolllisten (ACLs), Tags und Versionen im Quell-Bucket der Objekte. Zudem gewährt sie Berechtigungen für PUT-Objekte, ACLs, Tags und Versionen im Ziel-Bucket der Objekte.

2. Erstellen Sie im Quellkonto eine Bucket-Richtlinie für den Bucket, die der im vorherigen Schritt erstellten Rolle Berechtigungen für GET-Objekte, ACLs, Tags und Versionen in Quell-Manifest-Bucket gewährt.

   Dieser Schritt ermöglicht S3 Stapeloperationen das Lesen des Manifests unter Verwendung der vertrauenswürdigen Rolle. Weisen Sie die Bucket-Richtlinie dem Bucket zu, in dem sich das Manifest befindet.

   Das folgende Beispiel zeigt eine Bucket-Richtlinie, die dem Quell-Manifest-Bucket zugewiesen werden soll.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowBatchOperationsSourceManfiestRead",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::DestinationAccountNumber:user/
ConsoleUserCreatingJob",
                    "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
                ]
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::ObjectSourceManifestBucket/*"
        }
    ]
}
```

Diese Richtlinie gewährt außerdem die erforderlichen Berechtigungen, damit ein Konsolenbenutzer, der einen Auftrag im Zielkonto erstellt, diese Berechtigungen über dieselbe Bucket-Richtlinie auch für den Ziel-Manifest-Bucket erhält.

3. Erstellen Sie im Quellkonto eine Bucket-Richtlinie für den Quell-Bucket, die der erstellten Rolle Berechtigungen für GET-Objekte, ACLs, Tags und Versionen im Quell-Bucket mit den Objekten gewährt. S3 Stapeloperationen kann dann über die die vertrauenswürdige Richtlinie Objekte aus dem Quell-Bucket abrufen.

   Es folgt ein Beispiel für die Bucket-Richtlinie für den Bucket, der die Quellobjekte enthält.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowBatchOperationsSourceObjectCOPY",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging"
            ],
            "Resource": "arn:aws:s3:::ObjectSourceBucket/*"
        }
    ]
}
```

4. Erstellen Sie einen S3 Stapeloperationen-Auftrag im Zielkonto. Sie benötigen den Amazon-Ressourcennamen (ARN) der im Zielkonto erstellten Rolle.

   Allgemeine Informationen zum Erstellen eines Auftrags siehe Erstellen eines S3 Stapeloperationen-Auftrags (p. 554).

Informationen zum Erstellen eines Auftrags mit der Konsole finden Sie unter Erstellen eines S3 Stapeloperationen-Auftrags im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

S3 Stapeloperationen-Beispiele unter Verwenden der AWS CLI

S3 Stapeloperationen-Stapeloperationen verfolgen den Fortschritt nach, senden Benachrichtigungen und speichern einen detaillierten Abschlussbericht zu allen Aktionen. So erhalten Sie eine vollständig verwaltete, prüfbare und serverlose Umgebung. Sie können S3 Stapeloperationen über die AWS Management Console, die AWS CLI, die AWS-SDKs oder die REST-API verwenden. Weitere Informationen finden Sie unter the section called “Die Grundlagen: Aufträge” (p. 553).

Die folgenden Beispiele zeigen, wie Sie S3 Stapeloperationen mit der AWS Command Line Interface (AWS CLI) verwenden können.

Themen

Erstellen eines S3 Stapeloperationen-Auftrags mit der AWS CLI

Im folgenden Beispiel wird ein S3 Stapeloperationen `S3PutObjectTagging`-Auftrag mit AWS CLI erstellt.

So erstellen Sie einen Stapeloperationen-**`S3PutObjectTagging`**-Auftrag

1. Erstellen Sie eine AWS Identity and Access Management (IAM)-Rolle und weisen Sie Berechtigungen zu. Diese Rolle gewährt Amazon S3 die Berechtigung Objekt-Tags hinzuzufügen. Dafür erstellen Sie im nächsten Schritt einen Auftrag.

   a. Erstellen Sie folgendermaßen eine IAM-Rolle.

```
aws iam create-role \
 --role-name S3BatchJobRole \
 --assume-role-policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}'
```

   Notieren Sie sich den Amazon-Ressourcenname (ARN) der Rolle. Sie benötigen den ARN zum Erstellen eines Auftrags.

   b. Erstellen Sie eine IAM-Richtlinie mit Berechtigungen und fügen Sie sie der im vorherigen Schritt erstellten IAM-Rolle an. Weitere Informationen zu Berechtigungen finden Sie unter Gewähren von Berechtigungen für Amazon S3-Stapeloperationen (p. 556).

```
aws iam put-role-policy \
```

```
  --role-name S3BatchJobRole \
  --policy-name PutObjectTaggingBatchJobPolicy \
  --policy-document '{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectTagging",
        "s3:PutObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::{{TargetResource}}/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::{{ManifestBucket}}/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:GetBucketLocation"
      ],
      "Resource":[
        "arn:aws:s3:::{{ReportBucket}}/*"
      ]
    }
  ]
}'
```

2. Erstellen Sie einen `S3PutObjectTagging`-Auftrag.

   Die `manifest.csv`-Datei stellt eine Liste mit Bucket- und Objektschlüsselwerten bereit. Der Auftrag wendet die angegebenen Tags auf Objekte an, die im Manifest aufgelistet wurden. `ETag` ist das ETag des `manifest.csv`-Objekts, das Sie mit der Amazon S3-Konsole abrufen können. Die Anforderung enthält den `no-confirmation-required`-Parameter. Aus diesem Grund erteilt Amazon S3 die Berechtigung zur Ausführung des Auftrags, ohne dass Sie den `udpate-job-status`-Befehl ausführen müssen.

```
aws s3control create-job \
    --region us-west-2 \
    --account-id acct-id \
    --operation '{"S3PutObjectTagging": { "TagSet": [{"Key":"keyOne",
 "Value":"ValueOne"}] }}' \
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":{"ObjectArn":"arn:aws:s3:::my_manifests/
manifest.csv","ETag":"60e460c9d1046e73f7dde5043ac3ae85"}}' \
    --report '{"Bucket":"arn:aws:s3:::bucket-where-
completion-report-goes","Prefix":"final-reports",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}' \
    --priority 42 \
    --role-arn IAM-role \
    --client-request-token $(uuidgen) \
    --description "job Description" \
    --no-confirmation-required
```

Als Antwort gibt Amazon S3 eine Auftrags-ID zurück (beispielsweise `00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c`). Sie benötigen die ID in den nächsten Befehlen.

3. Rufen Sie die Beschreibung des Auftrags ab.

```
aws s3control describe-job \
    --region us-west-2 \
    --account-id acct-id \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c
```

4. Rufen Sie eine Liste mit `Active`- und `Complete`-Aufträgen ab.

```
aws s3control list-jobs \
    --region us-west-2 \
    --account-id acct-id \
    --job-statuses '["Active","Complete"]' \
    --max-results 20
```

5. Aktualisieren Sie die Priorität des Auftrags (eine höhere Zahl bedeutet eine höhere Priorität bei der Ausführung).

```
aws s3control update-job-priority \
    --region us-west-2 \
    --account-id acct-id \
    --priority 98 \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c
```

6. Wenn Sie den Parameter `--no-confirmation-required` in `create-job` nicht angegeben haben, bleibt der Auftrag so lange ausgesetzt, bis Sie den Auftrag bestätigen, indem Sie dessen Status auf `Ready` setzen. Amazon S3 erlaubt dann die Ausführung des Auftrags.

```
aws s3control update-job-status \
    --region us-west-2 \
    --account-id 181572960644 \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c \
    --requested-job-status 'Ready'
```

7. Brechen Sie den Auftrag ab, indem Sie den Auftragsstatus auf `Cancelled` einstellen.

```
aws s3control update-job-status \
     --region us-west-2 \
     --account-id 181572960644 \
     --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c \
     --status-update-reason "No longer needed" \
     --requested-job-status Cancelled
```

Verwenden von S3 Stapeloperationen-Auftrags-Tagging mithilfe der AWS CLI

Sie können den Zugriff auf Ihre S3 Stapeloperationen-Aufträge bezeichnen und steuern, indem Sie Tags hinzufügen. Anhand von Tags kann identifiziert werden, wer für einen Stapeloperationen-Job verantwortlich ist. Sie können Aufträge mit ihnen zugeordneten Tags erstellen oder Sie können Aufträgen nach ihrer Erstellung Tags hinzufügen. Weitere Informationen finden Sie unter the section called “Steuern von Zugriffs- und Labeling-Aufträgen mithilfe von Tags” (p. 572).

Themen

- Löschen der Tags eines S3 Stapeloperationen-Auftrags (p. 594)
- Anfordern der Auftrags-Tags eines S3 Stapeloperationen-Auftrags (p. 594)
- Einfügen von Auftrag-Tags in einen vorhandenen S3 Stapeloperationen-Auftrag (p. 594)

Erstellen eines S3 Stapeloperationen-Auftrags mit Tags

Im folgenden Beispiel wird ein S3 Stapeloperationen `S3PutObjectCopy`-Auftrag mit Auftrags-Tags als Bezeichnungen für den Auftrag mithilfe der AWS CLI erstellt.

1. Wählen Sie die Aktion oder `OPERATION` aus, die der Stapeloperationen-Auftrag ausführen soll, und wählen Sie Ihre `TargetResource` aus.

```
read -d '' OPERATION <<EOF
{
  "S3PutObjectCopy": {
    "TargetResource": "arn:aws:s3:::destination-bucket"
  }
}
EOF
```

2. Identifizieren Sie die Auftrags-`TAGS`, die Sie für den Auftrag wünschen. In diesem Fall wenden Sie beiden Tags `department` und `FiscalYear` mit den Werten `Marketing` bzw. `2020` an.

```
read -d '' TAGS <<EOF
[
  {
    "Key": "department",
    "Value": "Marketing"
  },
  {
    "Key": "FiscalYear",
    "Value": "2020"
  }
]
EOF
```

3. Geben Sie das `MANIFEST` für den Stapeloperationen-Auftrag an.

```
read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "EXAMPLE_S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::example-bucket/example_manifest.csv",
    "ETag": "example-5dc7a8bfb90808fc5d546218"
  }
}
EOF
```

4. Konfigurieren Sie den `REPORT` für den Stapeloperationen-Auftrag.

```
read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::example-report-bucket",
  "Format": "Example_Report_CSV_20180820",
```

```
  "Enabled": true,
  "Prefix": "reports/copy-with-replace-metadata",
  "ReportScope": "AllTasks"
}
EOF
```

5. Führen Sie die`create-job`-Aktion aus, um Ihren Stapeloperationen-Auftrag mit Eingaben zu erstellen, die in den vorherigen Schritten festgelegt wurden.

```
aws \
    s3control create-job \
    --account-id 123456789012 \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn arn:aws:iam::123456789012:role/batch-operations-role \
    --tags "${TAGS//$'\n'/}" \
    --client-request-token "$(uuidgen)" \
    --region us-west-2 \
    --description "Copy with Replace Metadata";
```

Löschen der Tags eines S3 Stapeloperationen-Auftrags

Im folgenden Beispiel werden die Tags aus einem Stapeloperationen-Auftrag mithilfe der AWS CLI gelöscht.

```
aws \
    s3control delete-job-tagging \
    --account-id 123456789012 \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Anfordern der Auftrags-Tags eines S3 Stapeloperationen-Auftrags

Im folgenden Beispiel werden die Tags eines Stapeloperationen-Auftrags mithilfe der AWS CLI abgerufen.

```
aws \
    s3control get-job-tagging \
    --account-id 123456789012 \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Einfügen von Auftrag-Tags in einen vorhandenen S3 Stapeloperationen-Auftrag

Im Folgenden finden Sie ein Beispiel für das Hinzufügen von Auftrags-Tags zu Ihrem S3 Stapeloperationen-Auftrag mit `s3control put-job-tagging` mithilfe der AWS CLI.

> Note
>
> Wenn Sie diese Anforderung mit einer leeren Tag-Menge senden, löscht S3 Stapeloperationen die vorhandene Tag-Menge für das Objekt. Wenn Sie diese Methode verwenden, wird Ihnen eine Tier 1-Anforderung (PUT) berechnet. Weitere Informationen finden Sie unter Amazon S3 – Preise. Wenn Sie vorhandene Tags für Ihren Stapeloperationen-Auftrag löschen möchten, wird die Aktion `DeleteJobTagging` bevorzugt, da sie dasselbe Ergebnis ohne anfallende Gebühren erzielt.

1. Identifizieren Sie die Auftrags-`TAGS`, die Sie für den Auftrag wünschen. In diesem Fall wenden Sie beiden Tags `department` und `FiscalYear` mit den Werten `Marketing` bzw. `2020` an.

```
read -d '' TAGS <<EOF
[
  {
    "Key": "department",
    "Value": "Marketing"
  },
  {
    "Key": "FiscalYear",
    "Value": "2020"
  }
]
EOF
```

2. Führen Sie die `put-job-tagging`-Aktion mit den erforderlichen Parametern aus.

```
aws \
    s3control put-job-tagging \
    --account-id 123456789012 \
    --tags "${TAGS//$'\n'/}" \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Beispiel: Erstellen und Verwalten von S3 Stapeloperationen-Aufträgen mithilfe des AWS SDK for Java

Dieser Abschnitt bietet Beispiele für das Erstellen und Verwalten von S3 Stapeloperationen-Aufträgen mit dem AWS SDK for Java. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

Themen

Erstellen eines Stapeloperationen-Auftrags

Im folgenden Beispiel wird ein S3 Stapeloperationen-Auftrag mit dem AWS SDK for Java erstellt.

Weitere Informationen zum Erstellen einer Aufgabe finden Sie unter Erstellen eines S3 Stapeloperationen-Auftrags (p. 554).

Weitere Informationen zum Einrichten der zum Erstellen eines Auftrags erforderlichen Berechtigungen finden Sie unter Gewähren von Berechtigungen für Amazon S3-Stapeloperationen (p. 556).

Example

```
package aws.example.s3control;
```

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.*;

import java.util.UUID;
import java.util.ArrayList;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateJob {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String iamRoleArn = "IAM Role ARN";
        String reportBucketName = "arn:aws:s3:::bucket-where-completion-report-goes";
        String uuid = UUID.randomUUID().toString();

        ArrayList tagSet = new ArrayList<S3Tag>();
        tagSet.add(new S3Tag().withKey("keyOne").withValue("ValueOne"));


        try {
            JobOperation jobOperation = new JobOperation()
                    .withS3PutObjectTagging(new S3SetObjectTaggingOperation()
                            .withTagSet(tagSet)
                    );

            JobManifest manifest = new JobManifest()
                    .withSpec(new JobManifestSpec()
                            .withFormat("S3BatchOperations_CSV_20180820")
                            .withFields(new String[]{
                                    "Bucket", "Key"
                            }))
                    .withLocation(new JobManifestLocation()
                            .withObjectArn("arn:aws:s3:::my_manifests/manifest.csv")
                            .withETag("60e460c9d1046e73f7dde5043ac3ae85"));
            JobReport jobReport = new JobReport()
                    .withBucket(reportBucketName)
                    .withPrefix("reports")
                    .withFormat("Report_CSV_20180820")
                    .withEnabled(true)
                    .withReportScope("AllTasks");

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.createJob(new CreateJobRequest()
                    .withAccountId(accountId)
                    .withOperation(jobOperation)
                    .withManifest(manifest)
                    .withReport(jobReport)
                    .withPriority(42)
                    .withRoleArn(iamRoleArn)
                    .withClientRequestToken(uuid)
                    .withDescription("job description")
                    .withConfirmationRequired(false)
            );

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
```

```
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Stornieren eines Stapeloperationen-Auftrags

Im folgenden Beispiel wird ein S3 Stapeloperationen-Auftrag mit dem AWS SDK for Java abgebrochen.

Example

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobStatusRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CancelJob {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobStatus(new UpdateJobStatusRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withStatusUpdateReason("No longer needed")
                    .withRequestedJobStatus("Cancelled"));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Aktualisieren des Status eines Stapeloperationen-Auftrags

Im folgenden Beispiel wird der Status eines S3 Stapeloperationen-Auftrags mithilfe des AWS SDK for Java aktualisiert.

Weitere Informationen zum Auftragsstatus finden Sie unter Sobald Ihr Auftrag erstellt ist, wird das Auftrags-Dashboard geöffnet, wo Sie Ihre Aufträge anzeigen und verwalten können. (p. 573).

Example

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobStatusRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateJobStatus {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobStatus(new UpdateJobStatusRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withRequestedJobStatus("Ready"));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Aktualisieren der Priorität eines Stapeloperationen-Auftrags

Im folgenden Beispiel wird die Priorität eines S3 Stapeloperationen-Auftrags mithilfe des AWS SDK for Java aktualisiert.

Weitere Informationen zur Auftragspriorität finden Sie unter Zuweisen der Auftragspriorität (p. 573).

Example

```
package aws.example.s3control;



import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
```

```
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobPriorityRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateJobPriority {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobPriority(new UpdateJobPriorityRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withPriority(98));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Verwenden von Stapeloperationen mit Tags

Sie können den Zugriff auf Ihre S3 Stapeloperationen-Aufträge bezeichnen und steuern, indem Sie Tags hinzufügen. Anhand von Tags kann identifiziert werden, wer für einen Stapeloperationen-Job verantwortlich ist. Sie können Aufträge mit ihnen zugeordneten Tags erstellen oder Sie können Aufträgen nach ihrer Erstellung Tags hinzufügen. Weitere Informationen finden Sie unter the section called “Steuern von Zugriffs- und Labeling-Aufträgen mithilfe von Tags” (p. 572).

Themen

Erstellen eines Stapeloperationen-Auftrags mit Auftrags-Tags, die für Labeling verwendet werden

Example

Im folgenden Beispiel wird mit dem AWS SDK for Java ein S3 Stapeloperationen-Auftrag mit Tags erstellt.

```
public String createJob(final AWSS3ControlClient awss3ControlClient) {
```

```
    final String manifestObjectArn = "arn:aws:s3:::example-manifest-bucket/
manifests/10_manifest.csv";
    final String manifestObjectVersionId = "example-5dc7a8bfb90808fc5d546218";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new
 JobManifestSpec().withFormat(JobManifestFormat.S3InventoryReport_CSV_20161130);

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::example-report-bucket";
    final String jobReportPrefix = "example-job-reports";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final String lambdaFunctionArn = "arn:aws:lambda:us-
west-2:123456789012:function:example-function";

    final JobOperation jobOperation = new JobOperation()
            .withLambdaInvoke(new
 LambdaInvokeOperation().withFunctionArn(lambdaFunctionArn));

    final S3Tag departmentTag = new S3Tag().withKey("department").withValue("Marketing");
    final S3Tag fiscalYearTag = new S3Tag().withKey("FiscalYear").withValue("2020");

    final String roleArn = "arn:aws:iam::123456789012:role/example-batch-operations-role";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Test lambda job")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withTags(departmentTag, fiscalYearTag)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Löschen der Auftrags-Tags eines Stapeloperationen-Auftrags

Example

Im folgenden Beispiel werden die Tags eines S3 Stapeloperationen-Auftrags mit der AWS SDK for Java gelöscht.

```
public void deleteJobTagging(final AWSS3ControlClient awss3ControlClient,
```

```
                              final String jobId) {
    final DeleteJobTaggingRequest deleteJobTaggingRequest = new DeleteJobTaggingRequest()
            .withJobId(jobId);

    final DeleteJobTaggingResult deleteJobTaggingResult =
                awss3ControlClient.deleteJobTagging(deleteJobTaggingRequest);
}
```

Anfordern der Auftrags-Tags eines Stapeloperationen-Auftrags

Example

Im folgenden Beispiel werden die Tags eines S3 Stapeloperationen-Auftrags mithilfe von AWS SDK for Java angefordert.

```
public List<S3Tag> getJobTagging(final AWSS3ControlClient awss3ControlClient,
                                  final String jobId) {
    final GetJobTaggingRequest getJobTaggingRequest = new GetJobTaggingRequest()
            .withJobId(jobId);

    final GetJobTaggingResult getJobTaggingResult =
            awss3ControlClient.getJobTagging(getJobTaggingRequest);

    final List<S3Tag> tags = getJobTaggingResult.getTags();

    return tags;
}
```

Einfügen von Auftrags-Tags in einen Stapeloperationen-Auftrag

Example

Im folgenden Beispiel werden die Tags eines S3 Stapeloperationen-Auftrags mithilfe von AWS SDK for Java abgelegt.

```
public void putJobTagging(final AWSS3ControlClient awss3ControlClient,
                          final String jobId) {
    final S3Tag departmentTag = new S3Tag().withKey("department").withValue("Marketing");
    final S3Tag fiscalYearTag = new S3Tag().withKey("FiscalYear").withValue("2020");

    final PutJobTaggingRequest putJobTaggingRequest = new PutJobTaggingRequest()
            .withJobId(jobId)
            .withTags(departmentTag, fiscalYearTag);

    final PutJobTaggingResult putJobTaggingResult =
 awss3ControlClient.putJobTagging(putJobTaggingRequest);
}
```

Hosten einer statischen Website auf Amazon S3

Sie können mit Amazon S3 eine statische Website hosten. Auf einer statischen Website enthalten einzelne Webseiten statischen Inhalt. Sie könnten auch clientseitige Skripts enthalten.

Im Gegensatz dazu basiert eine dynamische Website auf der serverseitigen Verarbeitung, einschließlich serverseitiger Skripts wie PHP, JSP oder ASP.NET. Amazon S3 unterstützt kein serverseitiges Skripting, aber AWS bietet andere Ressourcen für das Hosting dynamischer Websites. Um mehr über das Website-Hosting auf AWS zu erfahren, lesen Sie Web-Hosting.

Mithilfe der AWS Management Console können Sie Ihren Bucket für statisches Website-Hosting konfigurieren, ohne Code zu schreiben. Sie können die Website-Konfiguration unter Verwendung der AWS SDKs programmgesteuert erstellen, aktualisieren und löschen. Die SDKs stellen Wrapper-Klassen für die Amazon S3-REST-APIs bereit. Falls in Ihrer Anwendung erforderlich, können Sie auch direkt von Ihrer Anwendung aus REST API-Anfragen senden.

<title></title>
Um eine statische Website auf Amazon S3 zu hosten, konfigurieren Sie einen Amazon S3-Bucket für ein Website-Hosting und laden dann Ihren Website-Inhalt in den Bucket hoch. Wenn Sie einen Bucket als statische Website konfigurieren, müssen Sie statische Website-Hosting aktivieren (p. 605), Berechtigungen festlegen (p. 608) und ein Indexdokument erstellen und hinzufügen (p. 606). Abhängig von den Anforderungen Ihrer Website können Sie auch Weiterleitungen (p. 614), Webdatenverkehrs-Protokollierung (p. 611) und ein benutzerdefiniertes Fehlerdokument (p. 612) konfigurieren.

Nachdem Sie Ihren Bucket als statische Website konfiguriert haben, können Sie über die AWS Region-spezifischen Amazon S3-Website-Endpunkte für Ihren Bucket auf den Bucket zugreifen. Website-Endpunkte unterscheiden sich von den Endpunkten, von denen aus Sie REST API-Anfragen senden. Weitere Informationen finden Sie unter Website-Endpunkte (p. 602).

Weitere Informationen, einschließlich Anweisungen und Schritt-für-Schritt-Walkthroughs, finden Sie in den folgenden Themen:

Themen

Website-Endpunkte

Wenn Sie Ihren Bucket als statische Website konfigurieren, steht die Website an dem für die AWS-Region spezifischen Website-Endpunkt des Buckets zur Verfügung. Website-Endpunkte unterscheiden sich von den Endpunkten, von denen aus Sie REST API-Anfragen senden. Weitere Informationen zu den Unterschieden zwischen den Endpunkten finden Sie unter Wichtige Unterschiede zwischen einem Website-Endpunkt und einem REST API-Endpunkt (p. 604).

Je nach Region weisen Ihre Amazon S3-Website-Endpunkte eins der beiden folgenden Formate auf.

- s3-website dash (-) Region - `http://bucket-name.s3-website.Region.amazonaws.com`

- s3-website dot (.) Region - `http://bucket-name.s3-website-Region.amazonaws.com`

Diese URLs geben ein Standard-Indexdokument zurück, das Sie für die Website konfiguriert haben: Eine vollständige Liste der Amazon S3-Website-Endpunkte finden Sie unter Amazon S3-Website-Endpunkte.

Damit Ihre Kunden auf Inhalt am Website-Endpunkt zugreifen können, müssen Sie Ihren gesamten Inhalt öffentlich lesbar machen. Dazu können Sie die Einstellungen zum Blockieren des öffentlichen Zugriffs von S3 für das Konto bearbeiten. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493). Verwenden Sie anschließend eine Bucket-Richtlinie oder eine Zugriffskontrollliste (ACL) für ein Objekt, um die erforderlichen Berechtigungen zu erteilen. Weitere Informationen finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff (p. 608).

> **Important**
>
> Die Amazon S3-Website-Endpunkte unterstützen kein HTTPS. Informationen zur Verwendung von HTTPS mit einem Amazon S3-Bucket finden Sie im Folgenden:
>
> - Wie verwende ich CloudFront, um HTTPS-Anfragen für meinen Amazon S3-Bucket zu bedienen?
> - Anfordern von HTTPS für die Kommunikation zwischen CloudFront und Ihrem Amazon S3-Ursprung
>
> Buckets mit Zahlung durch den Anforderer erlauben keinen Zugriff über den Website-Endpunkt. Jede Anforderung an einen solchen Bucket erhält die Antwort 403 Access Denied (403 Zugriff verweigert). Weitere Informationen finden Sie unter Buckets mit Zahlung durch Auftraggeber (p. 82).

Themen

Website-Endpunkt-Beispiele

Die folgenden Beispiele zeigen, wie Sie auf einen Amazon S3-Bucket zugreifen können, der als statische Website konfiguriert ist.

Example – Anfordern eines Objekts auf Stammebene

Um ein bestimmtes Objekt anzufordern, das auf der Stammebene im Bucket gespeichert ist, verwenden Sie die folgende URL-Struktur.

```
http://bucket-name.s3-website.Region.amazonaws.com/object-name
```

Die folgende URL fordert beispielsweise das Objekt `photo.jpg` an, das auf der Stammebene im Bucket gespeichert ist.

```
http://example-bucket.s3-website.us-west-2.amazonaws.com/photo.jpg
```

Example – Anfordern eines Objekts in einem Präfix

Um ein Objekt anzufordern, das in einem Ordner in Ihrem Bucket gespeichert ist, verwenden Sie diese URL-Struktur.

```
http://bucket-name.s3-website.Region.amazonaws.com/folder-name/object-name
```

Die folgende URL fordert das `docs/doc1.html`-Objekt in Ihrem Bucket an.

```
http://example-bucket.s3-website.us-west-2.amazonaws.com/docs/doc1.html
```

Hinzufügen eines DNS-CNAME

Wenn Sie eine registrierte Domäne haben, können Sie einen DNS CNAME-Eintrag hinzufügen, der auf den Amazon S3 Website-Endpunkt verweist. Wenn Sie beispielsweise die Domäne `www.example-bucket.com` registriert haben, könnten Sie den Bucket `www.example-bucket.com` erstellen und einen DNS CNAME-Datensatz hinzufügen, der auf `www.example-bucket.com.s3-website.Region.amazonaws.com` verweist. Alle Anfragen an `http://www.example-bucket.com` werden an `www.example-bucket.com.s3-website.Region.amazonaws.com` weitergeleitet.

Weitere Informationen finden Sie unter Anpassen von Amazon S3-URLs mit CNAMEs (p. 51).

Verwenden einer benutzerdefinierten Domäne mit Route 53

Anstatt über einen Amazon S3-Website-Endpunkt auf die Website zuzugreifen, können Sie beispielsweise Ihre eigene Domain verwenden, die bei Amazon Route 53 registriert ist, um Ihre Inhalte bereitzustellen, z. B. `example.com`. Sie können Amazon S3 mit Route 53 verwenden, um eine Website in der Stammdomäne zu hosten. Wenn Sie beispielsweise die Root-Domäne `example.com` haben und Ihre Website auf Amazon S3 hosten, können Ihre Website-Besucher von ihrem Browser aus auf die Seite zugreifen, indem sie `http://www.example.com` oder `http://example.com` eingeben.

Einen Beispielanleitung finden Sie unter Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten, bei Route 53 registrierten Domäne (p. 631).

Wichtige Unterschiede zwischen einem Website-Endpunkt und einem REST API-Endpunkt

Der Amazon S3-Website-Endpunkt ist auf den Zugriff über einen Webbrowser optimiert. In der folgenden Tabelle werden die wichtigsten Unterschiede zwischen einem REST API-Endpunkt und einem Website-Endpunkt zusammengefasst.

Wichtiger Unterschied	REST API-Endpunkt	Website-Endpunkt
Zugriffskontrolle	Unterstützt öffentlichen und privaten Inhalt	Unterstützt nur öffentlich lesbaren Inhalt
Verarbeiten von Fehlermeldungen	Gibt eine XML-formatierte Fehlermeldung zurück	Gibt ein HTML-Dokument zurück
Unterstützung einer Umleitung	Nicht zutreffend	Unterstützt Umleitungen auf Objekt- und auf Bucket-Ebene
Unterstützte Anfragen	Unterstützt alle Bucket- und Objekt-Operationen	Unterstützt nur GET- und HEAD-Anforderungen für Objekte

Wichtiger Unterschied	REST API-Endpunkt	Website-Endpunkt
Reagiert auf GET- und HEAD-Anfragen an der Root eines Buckets	Gibt eine Liste der Objektschlüssel im Bucket zurück.	Gibt das in der Website-Konfiguration angegebene Indexdokument zurück.
Unterstützung von SSL (Secure Sockets Layer)	Unterstützt SSL-Verbindungen	Unterstützt keine SSL-Verbindungen.

Eine vollständige Liste der Amazon S3-Endpunkte finden Sie unter Amazon S3 Service Endpunkte und Kontingente im AWS General Reference.

Konfigurieren eines Buckets als statische Website mithilfe der AWS Management Console

Mit der AWS Management Console können Sie Ihren Amazon S3-Bucket als statische Website konfigurieren, ohne Code zu schreiben. Abhängig von Ihren Websiteanforderungen können Sie auch einige optionale Konfigurationen verwenden, einschließlich Weiterleitungen, Webdatenverkehrs-Protokollierung und benutzerdefinierte Fehlerdokumente.

Erforderliche Konfigurationen:

- Aktivieren des Website-Hostings (p. 605)
- Konfigurieren eines Indexdokuments (p. 606)
- Festlegen von Berechtigungen für den Website-Zugriff (p. 608)

Optionale Konfigurationen:

- (Optional) Konfigurierung eines benutzerdefinierten Fehlerdokuments (p. 612)
- (Optional) Konfigurieren einer Webseiten-Umleitung (p. 614)
- (Optional) Protokollieren des Webdatenverkehrs (p. 611)

Aktivieren des Website-Hostings

Wenn Sie einen Bucket als statische Website konfigurieren, müssen Sie das statische Website-Hosting aktivieren, ein Indexdokument konfigurieren und Berechtigungen festlegen.

Gehen Sie wie folgt vor, um das Website-Hosting für Ihre Amazon S3-Buckets mit der Amazon S3-Konsole zu aktivieren. Weitere Informationen zu den nächsten Schritten finden Sie unter Konfigurieren eines Indexdokuments (p. 606) und Festlegen von Berechtigungen für den Website-Zugriff (p. 608). Informationen zum Konfigurieren Ihrer Website mit einer benutzerdefinierten Domäne finden Sie unter Beispielhafte Walkthroughs – Hosting von Websites auf Amazon S3 (p. 625).

So aktivieren Sie das statische Website-Hosting

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Wählen Sie in der Liste Bucket name (Bucket-Name) den Bucket aus, den Sie für Ihre statische Website verwenden möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Wählen Sie Use this bucket to host a website (Diesen Bucket zum Hosten einer Website verwenden).
6. Geben Sie den Namen Ihres Index-Dokuments ein.

   Der Name des Indexdokuments lautet in der Regel `index.html`. Der Name des Indexdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Indexdokuments übereinstimmen, das Sie in den S3-Bucket hochladen möchten. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments (p. 606).
7. (Optional) Wenn Sie ein benutzerdefiniertes Fehlerdokument hinzufügen möchten, geben Sie im Feld Error document (Fehlerdokument) den Schlüsselnamen des Fehlerdokuments ein (z. B. **`error.html`**).

   Der Name des Fehlerdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Fehlerdokuments übereinstimmen, das Sie in Ihren S3-Bucket hochladen möchten. Weitere Informationen finden Sie unter (Optional) Konfigurierung eines benutzerdefinierten Fehlerdokuments (p. 612).
8. (Optional) Wenn Sie erweiterte Umleitungsregeln angeben möchten, verwenden Sie unter Edit redirection rules (Umleitungsregeln bearbeiten) XML zur Beschreibung der Regeln.

   Weitere Informationen finden Sie unter Konfigurieren erweiterter bedingter Umleitungen (p. 617).
9. Notieren Sie unter Static website hosting (Statisches Website-Hosting) den Wert für Endpoint (Endpunkt).

   Der Endpoint (Endpunkt) ist der Amazon S3-Website-Endpunkt für Ihren Bucket. Nachdem Sie den Bucket als statische Website konfiguriert haben, können Sie diesen Endpunkt verwenden, um Ihre Website zu testen.
10. Wählen Sie Save (Speichern) aus.

Als Nächstes müssen Sie das Indexdokument konfigurieren und Berechtigungen festlegen. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments (p. 606) und Festlegen von Berechtigungen für den Website-Zugriff (p. 608). Optional können Sie auch ein Fehlerdokument (p. 612), eineWebdatenverkehrs-Protokollierung (p. 611) oder eine Weiterleitung (p. 614) konfigurieren.

Konfigurieren eines Indexdokuments

Wenn Sie Website-Hosting aktivieren, müssen Sie auch ein Indexdokument konfigurieren und hochladen. Ein Indexdokument ist eine Webseite, die Amazon S3 zurückgibt, wenn eine Anfrage für die Root einer Website oder einen Unterordner erfolgt. Wenn ein Benutzer beispielsweise `http://www.example.com` in den Browser eingibt, fordert er keine spezifische Seite an. In diesem Fall stellt Amazon S3 das Indexdokument bereit, das manchmal auch als Standardseite bezeichnet wird.

Wenn Sie das statische Website-Hosting für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B. `index.html`). Nachdem Sie das statische Website-Hosting für Ihren Bucket aktiviert haben, laden Sie eine HTML-Datei mit dem Indexdokumentnamen in Ihren Bucket hoch.

Der abschließend Schrägstrich der Root-Level-URL ist optional. Wenn Sie beispielsweise Ihre Website mit `index.html` als Indexdokument konfigurieren, gibt jede der folgenden URLs `index.html` zurück.

```
http://example-bucket.s3-website.Region.amazonaws.com/
http://example-bucket.s3-website.Region.amazonaws.com
```

Weitere Informationen zu Amazon S3-Website-Endpunkten finden Sie unter Website-Endpunkte (p. 602).

Indexdokument und Ordner

In Amazon S3 ist ein Bucket ein eindimensionaler Container mit Objekten. Er bietet keine hierarchische Organisation wie das Dateisystem auf Ihrem Computer. Sie können jedoch mit Objektschlüsselnamen, die eine Ordnerstruktur implizieren, eine logische Hierarchie erstellen.

Betrachten wir beispielsweise einen Bucket mit drei Objekten, die die folgenden Schlüsselnamen besitzen. Obwohl diese nicht in einer physischen Hierarchie gespeichert sind, können Sie aus den Schlüsselnamen die folgende logische Ordnerstruktur ableiten:

- `sample1.jpg`: Objekt befindet sich im Stammverzeichnis des Buckets.
- `photos/2006/Jan/sample2.jpg`: Das Objekt befindet sich im Unterordner `photos/2006/Jan`.
- `photos/2006/Feb/sample3.jpg`: Das Objekt befindet sich im Unterordner `photos/2006/Feb`.

In der Amazon S3-Konsole können Sie auch einen Ordner in einem Bucket erstellen. Sie können beispielsweise einen Ordner mit dem Namen `photos` erstellen. Sie können Objekte in den Bucket hochladen, oder in den `photos`-Ordner im Bucket. Wenn Sie dem Bucket das Objekt `sample.jpg` hinzufügen, ist der Schlüsselname `sample.jpg`. Wenn Sie das Objekt in den Ordner `photos` hochladen, ist der Objektschlüsselname `photos/sample.jpg`.

Wenn Sie eine Ordnerstruktur in Ihrem Bucket erstellen, muss es auf jeder Ebene ein Indexdokument geben. In jedem Ordner muss das Indexdokument den gleichen Namen haben, `index.html`. Wenn ein Benutzer eine URL angibt, die an eine Ordnersuche erinnert, bestimmt das Vorhandensein oder Fehlen eines abschließenden Schrägstrichs das Verhalten der Website. Die folgende URL beispielsweise mit einem abschließenden Schrägstrich gibt das Indexdokument `photos/index.html` zurück.

```
http://bucket-name.s3-website.Region.amazonaws.com/photos/
```

Wenn Sie jedoch den abschließenden Schrägstrich aus der obigen URL weglassen, sucht Amazon S3 zuerst im Bucket nach einem Objekt `photos`. Wird das Objekt `photos` nicht gefunden, sucht es nach einem Indexdokument, `photos/index.html`. Wird dieses Dokument gefunden, gibt Amazon S3 die Meldung `302 Found` zurück und verweist auf den `photos/`-Schlüssel. Für nachfolgende Anfragen für `photos/` gibt Amazon S3 `photos/index.html` zurück. Wenn das Index-Dokument nicht gefunden wird, gibt Amazon S3 einen Fehler zurück.

Konfigurieren eines Indexdokuments

Wenn Sie das Hosting statischer Websites für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B. **`index.html`**). Nachdem Sie das Hosting statischer Websites für den Bucket aktiviert haben, laden Sie eine HTML-Datei mit diesem Indexdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie das Indexdokument

1. Erstellen Sie eine Datei `index.html`.

   Wenn Sie nicht über eine Datei `index.html` verfügen, können Sie mit dem folgenden HTML-Code eine Datei erstellen:

   ```
   <html xmlns="http://www.w3.org/1999/xhtml" >
   <head>
       <title>My Website Home Page</title>
   </head>
   <body>
     <h1>Welcome to my website</h1>
     <p>Now hosted on Amazon S3!</p>
   </body>
   </html>
   ```

2. Speichern Sie die Indexdatei lokal und notieren Sie sich den Dateinamen (z. B. `index.html`).

   Wenn Sie das Hosting statischer Websites aktivieren und Ihren Indexdokumentnamen eingeben, geben Sie diesen exakten Dateinamen ein (z. B. `index.html`).
3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Indexdokuments ein (z. B. `index.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings (p. 605).

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.
6. Führen Sie einen der folgenden Schritte aus, um das Indexdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Indexdatei per Drag & Drop in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   Schrittweise Anleitungen finden Sie unter Wie lade ich Dateien und Ordner in einen Amazon S3-Bucket hoch? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
7. (Optional) Laden Sie andere Website-Inhalte in Ihren Bucket hoch.

Als Nächstes müssen Sie Berechtigungen für den Websitezugriff festlegen. Weitere Informationen finden Sie unter Festlegen von Berechtigungen für den Website-Zugriff (p. 608). Optional können Sie auch ein Fehlerdokument (p. 612), eineWebdatenverkehrs-Protokollierung (p. 611) oder eine Weiterleitung (p. 614) konfigurieren.

Festlegen von Berechtigungen für den Website-Zugriff

Wenn Sie einen Bucket als statische Website konfigurieren und Ihre Website öffentlich sein soll, können Sie öffentlichen Lesezugriff gewähren. Um Ihren Bucket öffentlich lesbar zu machen, müssen Sie die Einstellungen für den öffentlichen Zugriff für den Bucket deaktivieren und eine Bucket-Richtlinie schreiben, die öffentlichen Lesezugriff gewährt. Wenn Ihr Bucket Objekte enthält, die nicht im Besitz des Bucket-Eigentümers sind, müssen Sie möglicherweise auch eine Objekt-Zugriffskontrollliste (Access Control List, ACL) hinzufügen, die jedem Benutzer Lesezugriff erteilt.

Note

Wenn ein Benutzer auf dem Endpunkt der Website ein Objekt anfordert, das nicht existiert, gibt Amazon S3 den HTTP-Antwortcode `404 (Not Found)` zurück. Wenn das Objekt existiert, aber Sie keine Leseberechtigungen dafür erteilt haben, gibt der Website-Endpunkt den HTTP-Antwortcode `403 (Access Denied)` zurück. Der Benutzer kann aus dem Antwortcode ableiten, ob ein bestimmtes Objekt existiert. Wenn Sie dieses Verhalten nicht wünschen, sollten Sie den Website-Support für Ihren Bucket nicht aktivieren.

Schritt 1: Bearbeiten der Block Public Access-Einstellungen

Wenn Sie einen bestehenden Bucket als statische Website konfigurieren möchten, die einen öffentlichen Zugriff bietet, müssen Sie die Einstellungen für den öffentlichen Zugriff für diesen Bucket bearbeiten. Sie müssen möglicherweise auch Ihre Einstellungen zur Blockierung des öffentlichen Zugriffs auf Kontoebene bearbeiten. Amazon S3 wendet die restriktivste Kombination der Einstellungen zur Blockierung des öffentlichen Zugriffs auf Bucket- und Kontoebene an.

Wenn Sie beispielsweise den öffentlichen Zugriff für einen Bucket erlauben, aber den gesamten öffentlichen Zugriff auf Kontoebene blockieren, blockiert Amazon S3 den öffentlichen Zugriff auf den Bucket

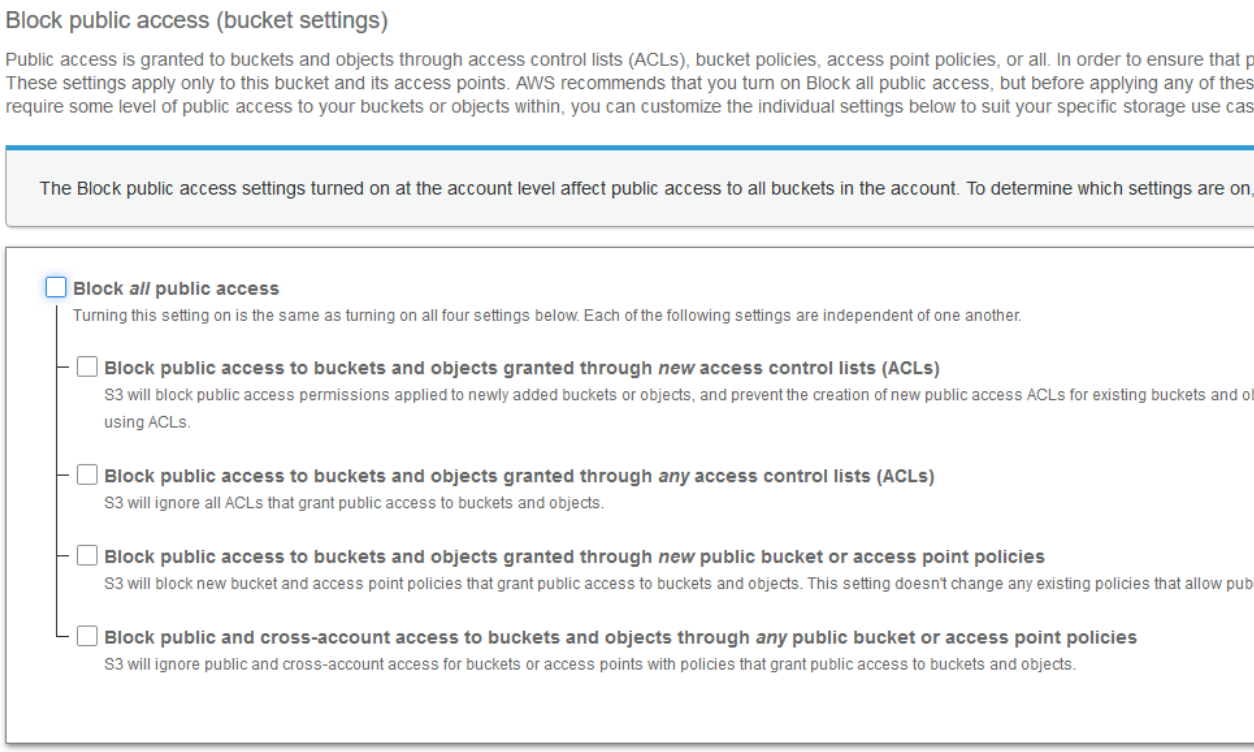
weiter. In diesem Szenario müssten Sie Ihre Einstellungen zum Blockieren des öffentlichen Zugriffs auf Bucket- und Kontoebene bearbeiten. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Block Public Access (p. 493).

Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

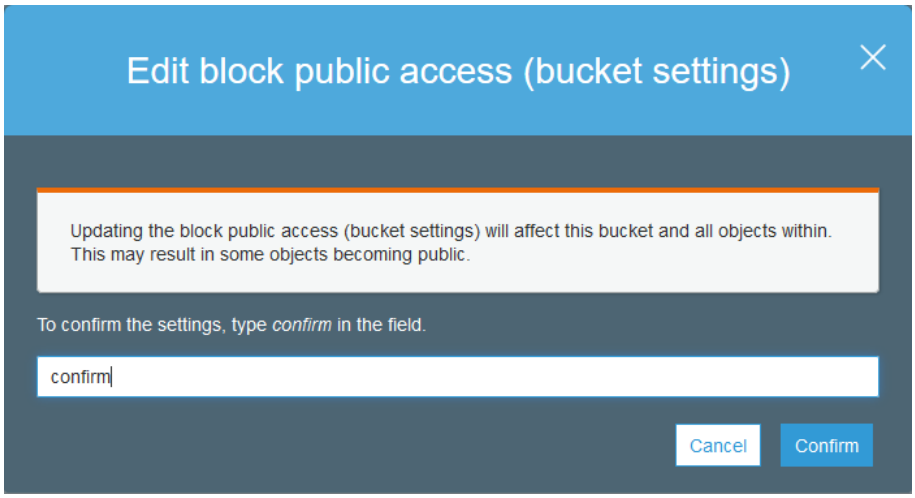
> Warning
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Verwenden von Amazon S3 Block Public, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren), und wählen Sie Save (Speichern).

> Warning
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Verwenden von Amazon S3Block Public, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



6. Geben Sie im Bestätigungsfeld **`confirm`** ein, und wählen Sie dann Confirm (Bestätigen).



   Unter S3-Buckets wird der Access (Zugriff) auf Ihren Bucket auf Objects can be public (Objekte können öffentlich sein) aktualisiert. Sie können nun eine Bucket-Richtlinie hinzufügen, um die Objekte im Bucket öffentlich lesbar zu machen. Wenn der Access (Zugriff) immer noch als Bucket and objects not public (Bucket und Objekte nicht öffentlich) angezeigt wird, müssen Sie möglicherweise die Einstellungen für die Blockierung des öffentlichen Zugriff für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen.

Schritt 2: Hinzufügen einer Bucket-Richtlinie

Um die Objekte in Ihrem Bucket öffentlich lesbar zu machen, müssen Sie eine Bucket-Richtlinie schreiben, die jedem die `s3:GetObject`-Berechtigung erteilt.

Nachdem Sie die Einstellungen für S3 Block Public Access bearbeitet haben, können Sie eine Bucket-Richtlinie hinzufügen, um öffentlichen Lesezugriff auf den Bucket zu gewähren. Wenn Sie öffentlichen Lesezugriff gewähren, kann jeder im Internet auf Ihren Bucket zugreifen.

> **Important**
>
> Die zuvor genannte Richtlinie ist nur ein Beispiel und erlaubt Vollzugriff auf die Inhalte Ihres Buckets. Bevor Sie mit diesem Schritt fortfahren, lesen Sie den Abschnitt Wie kann ich die Dateien in meinem Amazon S3-Bucket sichern? , um sicherzustellen, dass Sie die bewährten Methoden zum Sichern der Dateien in Ihrem S3-Bucket und die Risiken in Zusammenhang mit der Gewährung von öffentlichem Zugriff kennen.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Permissions.
3. Wählen Sie Bucket Policy aus.
4. Um öffentlichen Lesezugriff auf Ihre Website zu gewähren, kopieren Sie die folgende Bucket-Richtlinie und fügen Sie sie in den Bucket policy editor (Bucket-Richtlinieneditor) ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
```

```
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::example.com/*"
            ]
        }
    ]
}
```

5. Aktualisieren Sie den `Resource`, um Ihren Bucket-Namen anzugeben.

   In der vorangegangenen Beispiel-Bucket-Richtlinie ist *example.com* der Bucket-Name. Um diese Bucket-Richtlinie mit Ihrem eigenen Bucket zu verwenden, müssen Sie diesen Namen so aktualisieren, dass er mit Ihrem Bucket übereinstimmt.

6. Wählen Sie Save (Speichern) aus.

   Es wird eine Warnung angezeigt, die darauf hinweist, dass der Bucket öffentlich zugänglich ist. Unter Bucket Policy (Bucket-Richtlinie) wird die Bezeichnung Public (Öffentlich) angezeigt.

   Wenn die Fehlermeldung `Policy has invalid resource` angezeigt wird, bestätigen Sie, dass der Bucket-Name in der Bucket-Richtlinie mit Ihrem Bucket-Namen übereinstimmt. Informationen zum Hinzufügen einer Bucket-Richtlinie finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu?

   Wenn Sie die Warnung Error - Access denied (Fehler - Zugriff verweigert) erhalten und der Bucket policy editor (Bucket-Richtlinieneditor) das Speichern der Bucket-Richtlinie nicht zulässt, überprüfen Sie die Block Public Access-Einstellungen auf Kontoebene und Bucket-Ebene, um zu bestätigen, dass Sie öffentlichen Zugriff auf den Bucket zulassen. Weitere Informationen zu Website-Berechtigungen finden Sie unter Erforderliche Berechtigungen für den Website-Zugriff.

Objektzugriffskontrolllisten

Sie können eine Bucket-Richtlinie verwenden, um Ihren Objekten öffentliche Leserechte zu erteilen. Die Bucket-Richtlinie gilt jedoch nur für Objekte, die sich im Besitz des Bucket-Eigentümers befinden. Wenn Ihr Bucket Objekte enthält, die nicht dem Bucket-Eigentümer gehören, sollte der Bucket-Eigentümer die Objektzugriffskontrollliste (ACL) verwenden, um öffentlichen Lesezugriff auf diese Objekte zu gewähren.

Um ein Objekt über eine ACL öffentlich lesbar zu machen, erteilen Sie der Gruppe `AllUsers` die READ-Berechtigung, wie im folgenden Berechtigungselement dargestellt. Fügen Sie dieses Rechteelement der Objekt-ACL hinzu. Weitere Informationen zur Verwaltung von ACLs finden Sie unter Zugriffsverwaltung mit ACLs (p. 479).

```
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:type="Group">
    <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
  </Grantee>
  <Permission>READ</Permission>
</Grant>
```

(Optional) Protokollieren des Webdatenverkehrs

Sie können optional die Amazon S3-Serverzugriffsprotokollierung für einen Bucket aktivieren, der als statische Website konfiguriert ist. Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anfragen, die an Ihren Bucket gestellt werden. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777). Wenn Sie jedoch planen, zur Beschleunigung Ihrer Website (p. 641) Amazon CloudFront zu verwenden, können Sie auch die CloudFront-Protokollierung

verwenden. Weitere Informationen finden Sie unter Konfigurieren und Verwenden von Zugriffsprotokollen im Entwicklerhandbuch für Amazon CloudFront.

So aktivieren Sie die Serverzugriffsprotokollierung für Ihren statischen Website-Bucket

1. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.
2. Erstellen Sie in derselben Region, in der Sie den Bucket erstellt haben, der als statische Website konfiguriert ist, einen Bucket für die Protokollierung, z. B. `logs.example.com`.
3. Erstellen Sie einen Ordner für die Protokolldateien der Serverzugriffsprotokollierung (z. B. `logs`).
4. (Optional) Wenn Sie die Leistung Ihrer Website mit CloudFront verbessern möchten, erstellen Sie einen Ordner für die CloudFront-Protokolldateien (z. B. `cdn`).
5. Wählen Sie in der Liste Bucket den Bucket aus.
6. Wählen Sie Properties (Eigenschaften) aus.
7. Wählen Sie Server access logging (Serverzugriffsprotokollierung) aus.
8. Wählen Sie Enable logging (Protokollierung aktivieren) aus.
9. Wählen Sie unter Target Bucket (Ziel-Bucket) den Bucket aus, den Sie für die Protokolldateien erstellt haben, z. B. `logs.example.com`.
10. Geben Sie unter Target prefix (Zielpräfix) den Namen des Ordners ein, den Sie für die Protokolldateien erstellt haben, gefolgt von dem Trennzeichen (/), z. B. **`logs/`**.

    Wenn Sie das Target prefix (Zielpräfix) festlegen, gruppieren Sie die Protokolldatendateien in einem Ordner, sodass sie leicht zu finden sind.
11. Wählen Sie Save (Speichern) aus.

    In Ihrem Protokoll-Bucket können Sie nun auf Ihre Protokolle zugreifen. Amazon S3 schreibt alle 2 Stunden Website-Zugriffsprotokolle in Ihren Protokoll-Bucket.
12. Um die Protokolle anzuzeigen, wählen Sie Overview (Überblick) und den Ordner aus.

(Optional) Konfigurierung eines benutzerdefinierten Fehlerdokuments

Nachdem Sie Ihren Bucket als statische Website konfiguriert haben, gibt ein HTML-Fehlerdokument Amazon S3 zurück, wenn ein Fehler auftritt. Sie können Ihren Bucket optional mit einem benutzerdefinierten Fehlerdokument konfigurieren, sodass Amazon S3 bei Auftreten eines Fehlers dieses Dokument zurückgibt.

> Note
>
> Tritt jedoch ein Fehler auf, zeigen einige Browser beim Auftreten eines Fehlers ihre eigene Fehlermeldung an und ignorieren das Fehlerdokument, das Amazon S3 zurückgibt. Wenn beispielsweise ein Fehler HTTP 404 Not Found auftritt, ignoriert Google Chrome möglicherweise das Fehler-Dokument, das Amazon S3 zurückgibt, und zeigt seinen eigenen Fehler an.

Themen

Amazon S3-HTTP-Antwortcodes

Die folgende Tabelle listet die Untermenge der HTTP-Antwortcodes auf, die Amazon S3 zurückgibt, wenn ein Fehler auftritt.

HTTP-Fehlercode	Beschreibung
301 Moved Permanently (301 Dauerhaft verschoben)	Wenn ein Benutzer eine Anforderung direkt an die Amazon S3-Website-Endpunkte(`http://s3-website.Region.amazonaws.com/`) sendet, gibt Amazon S3 als Antwort 301 Moved Permanently (301 dauerhaft verschoben) zurück und leitet diese Anforderungen an `https://aws.amazon.com/s3/` weiter.
302 Found (302 Gefunden)	Wenn Amazon S3 eine Anfrage für den Schlüssel `x`, `http://bucket-name.s3-website.Region.amazonaws.com/x`, ohne nachfolgenden Schrägstrich erhält, sucht es zuerst nach dem Objekt mit dem Schlüsselnamen `x`. Wenn das Objekt nicht gefunden wird, stellt Amazon S3 fest, dass die Anfrage für den Unterordner `x` vorgesehen ist, und leitet die Anfrage um, indem es einen Schrägstrich am Ende einfügt. Es gibt 302 Found (302 Gefunden) zurück.
304 Not Modified (304 Nicht verändert)	Amazon S3-Benutzer fragen die Header `If-Modified-Since`, `If-Unmodified-Since`, `If-Match` und/oder `If-None-Match` an, um festzustellen, ob das angefragte Objekt mit der zwischengespeicherten Kopie des Clients identisch ist. Ist das Objekt dasselbe, gibt der Website-Endpunkt die Antwort 304 Not Modified (304 Nicht verändert) zurück.
400 Malformed Request (400 Falsch formatierte Anfrage)	Der Website-Endpunkt antwortet mit 400 Malformed Request (400 Falsch formatierte Anfrage), wenn ein Benutzer versucht, über einen inkorrekten regionalen Endpunkt auf einen Bucket zuzugreifen.
403 Forbidden	Der Website-Endpunkt antwortet mit 403 Forbidden (403 Verboten), wenn ein Benutzer auf ein Objekt zugreifen will, das nicht öffentlich lesbar ist. Der Objekteigentümer muss zuerst das Objekt über eine Bucket-Richtlinie oder eine ACL öffentlich lesbar machen.
404 Not Found (404 Nicht gefunden)	Der Website-Endpunkt gibt aus den folgenden Gründen 404 Not Found (404 Nicht gefunden) zurück: • Amazon S3 stellt fest, dass die URL der Website auf einen nicht existierenden Objektschlüssel verweist. • Amazon S3 bedeutet, dass es sich bei der Anforderung um ein Index-Dokument handelt, das nicht existiert. • Ein in der URL angegebener Bucket ist nicht vorhanden. • Ein in der URL angegebener Bucket ist vorhanden, aber nicht als Website konfiguriert. Sie können ein benutzerdefiniertes Objekt erstellen, das für 404 Not Found (404 Nicht gefunden) zurückgegeben wird. Stellen Sie sicher, dass das Dokument in den als Website konfigurierten Bucket hochgeladen ist, und dass die Konfiguration für das Website-Hosting darauf ausgelegt ist, das Dokument zu verwenden. Weitere Informationen darüber, wie Amazon S3 die URL als Anfrage für ein Objekt oder ein Indexdokument interpretiert, finden Sie unter Konfigurieren eines Indexdokuments (p. 606).
500 Service Error (500 Servicefehler)	Der Website-Endpunkt reagiert mit 500 Service Error (500 Servicefehler), wenn ein interner Serverfehler auftritt.
503 Service Unavailable (503	Der Website-Endpunkt antwortet mit 503 Service Unavailable (503 Service nicht verfügbar), wenn Amazon S3 feststellt, dass Sie Ihre Anfragerate reduzieren müssen.

HTTP-Fehlercode	Beschreibung
Service nicht verfügbar)	

Für jeden dieser Fehler gibt Amazon S3 eine vordefinierte HTML-Meldung zurück. Nachfolgend sehen Sie ein Beispiel für eine HTML-Meldung, die für eine 403 Forbidden (403 Verboten)-Nachricht zurückgegeben wird.

Konfigurieren eines benutzerdefinierten Fehlerdokuments

Wenn Sie Ihren Bucket als statische Website konfigurieren, können Sie optional ein benutzerdefiniertes Fehlerdokument bereitstellen, das eine benutzerfreundliche Fehlermeldung und zusätzliche Hilfe enthält. Amazon S3 gibt Ihr benutzerdefiniertes Fehlerdokument nur für die HTTP-4XX-Klasse von Fehlercodes zurück.

So konfigurieren Sie ein benutzerdefiniertes Fehlerdokument

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets (Buckets) den Namen Ihres Buckets.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).

   Wenn Ihr Bucket bereits als statische Website konfiguriert ist, können Sie den nächsten Schritt ausführen, um Fehlerdokumentinformationen zu aktualisieren oder hinzuzufügen. Wenn Sie Ihren Bucket nicht als statische Website konfiguriert haben, müssen Sie zunächst die erforderliche Konfiguration einrichten. Weitere Informationen finden Sie unter Aktivieren des Website-Hostings (p. 605).
5. Geben Sie im Feld Error document (Fehlerdokument) den Namen des Fehlerdokuments ein.
6. Wählen Sie Save (Speichern) aus.

Weitere Informationen zur Verwendung der REST-API zum Konfigurieren des Buckets als statische Website mit einem benutzerdefinierten Fehlerdokument finden Sie unter PutBucketWebsite im Amazon Simple Storage Service API Reference.

(Optional) Konfigurieren einer Webseiten-Umleitung

Wenn Ihr Amazon S3-Bucket für das Website-Hosting konfiguriert ist, können Sie eine Webseitenumleitung konfigurieren. Sie haben die folgenden Optionen für die Konfigurierung einer Umleitung.

Themen

Einrichten einer Seitenumleitung in der Amazon S3-Konsole

Sie können Anforderungen für ein Objekt an ein anderes Objekt oder eine andere URL umleiten, indem Sie in den Metadaten des Objekts den Speicherort für die Websiteumleitung festlegen. Sie richten die Umleitung ein, indem Sie den Objekt-Metadaten die `x-amz-website-redirect-location`-Eigenschaft hinzufügen. In der Amazon S3-Konsole legen Sie den Wert für Website Redirect Location (Websiteumleitungsort) in den Metadaten des Objekts fest. Wenn Sie die Amazon S3-API (p. 616) verwenden, legen Sie `x-amz-website-redirect-location` fest. Die Website interpretiert anschließend das Objekt als eine 301-Umleitung.

Um eine Anfrage an ein anderes Objekt umzuleiten, richten Sie den Umleitungsstandort auf den Schlüssel für das Zielobjekt ein. Um eine Anfrage an eine externe URL umzuleiten, richten Sie den Umleitungsstandort auf die gewünschte URL ein. Weitere Informationen zu Objekt-Metadaten erhalten Sie unter Systemdefinierte Objektmetadaten (p. 120).

Wenn Sie eine Seitenumleitung einrichten, können Sie den Inhalt des Quellobjekts beibehalten oder löschen. Wenn es in Ihrem Bucket beispielsweise das Objekt `page1.html` gibt, können Sie alle Anforderungen für diese Seite an ein anderes Objekt umleiten, `page2.html`. Sie haben hierfür zwei Möglichkeiten:

- Behalten Sie den Inhalt des Objekts `page1.html` und leiten Sie Seitenanforderungen um.
- Löschen Sie den Inhalt von `page1.html` und laden Sie ein Null-Byte-Objekt mit dem Namen `page1.html` hoch, um das vorhandene Objekt zu ersetzen und Seitenanforderungen umzuleiten.

So leiten Sie Anforderungen für ein Objekt um

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben (z. B. `example.com`).
3. Wählen Sie auf der Registerkarte Overview (Übersicht) für Ihren Bucket das Objekt aus, für das Sie eine Umleitung erstellen möchten.
4. Wählen Sie Properties (Eigenschaften) aus.
5. Wählen Sie Metadata (Metadaten) aus.
6. Wählen Sie + Add Metadata (+ Metadaten hinzufügen) aus.
7. Wählen Sie unter Key (Schlüssel) die Option Website-Redirect-Location (Websiteumleitungsort) aus.
8. Geben Sie in Value (Wert) den Schlüsselnamen des Objekts ein, zu dem Sie umleiten möchten, z. B. `/page2.html`.

   Für ein anderes Objekt im selben Bucket ist das Präfix `/` im Wert erforderlich. Sie können den Wert auch auf eine externe URL festlegen, z. B. `http://www.example.com`.
9. Wählen Sie Save aus.

Einrichten einer Seitenumleitung von REST API aus

Die folgenden Amazon S3-API-Aktionen unterstützen den Header `x-amz-website-redirect-location` in der Anfrage. Amazon S3 speichert den Header-Wert in den Objekt-Metadaten als `x-amz-website-redirect-location`.

- PUT Object
- Initiieren eines mehrteiligen Uploads
- POST Object
- PUT Object – Kopieren

Ein Bucket, der für das Website-Hosting konfiguriert ist, hat sowohl den Website-Endpunkt als auch den REST-Endpunkt. Eine Anfrage für eine Seite, die als 301-Umleitung konfiguriert ist, erzeugt die folgenden möglichen Ergebnisse, abhängig vom Endpunkt der Anfrage:

- Regionsspezifischer Website-Endpunkt – Amazon S3; leitet die Seitenanfrage in Übereinstimmung mit dem Wert der `x-amz-website-redirect-location`-Eigenschaft um.
- REST-Endpunkt – Amazon S3 leitet die Seitenanfrage nicht um. Es gibt das angefragte Objekt zurück.

Weitere Informationen zu den Endpunkten finden Sie unter Wichtige Unterschiede zwischen einem Website-Endpunkt und einem REST API-Endpunkt (p. 604).

Wenn Sie eine Seitenumleitung einrichten, können Sie den Inhalt des Objekts beibehalten oder löschen. Nehmen wir zum Beispiel an, dass Sie ein `page1.html`-Objekt in Ihrem Bucket haben.

- Um den Inhalt von `page1.html` beizubehalten und nur Seitenanfragen umzuleiten, können Sie eine PUT Object - Copy-Anfrage stellen, um ein neues `page1.html`-Objekt zu erstellen, das das vorhandene `page1.html`-Objekt als Quelle verwendet. In Ihrer Anfrage richten Sie den `x-amz-website-redirect-location`-Header ein. Nachdem die Anfrage abgeschlossen ist, ist die Originalseite mit ihrem Inhalt unverändert, aber Amazon S3 leitet alle Anfragen für die Seite an den von Ihnen angegebenen Umleitungsstandort um.
- Um den Inhalt des `page1.html`-Objekts zu löschen und Anforderungen für die Seite umzuleiten, können Sie eine PUT-Objektanforderung senden, um ein Nullbyte-Objekt hochzuladen, das den gleichen Objektschlüssel hat: `page1.html`. In der PUT-Anfrage setzen Sie `x-amz-website-redirect-location` für `page1.html` auf das neue Objekt. Nachdem die Anfrage abgeschlossen ist, hat `page1.html` keinen Inhalt mehr, und Anfragen werden an den Standort umgeleitet, der in `x-amz-website-redirect-location` angegeben ist.

Wenn Sie das Objekt mit der Aktion GET Object abrufen, gibt Amazon S3 zusammen mit anderen Objekt-Metadaten den `x-amz-website-redirect-location`-Header in der Antwort zurück.

Umleiten von Anforderungen für den Website-Endpunkt eines Buckets an einen anderen Host

Sie können alle Anforderungen für einen Website-Endpunkt eines Buckets an einen anderen Host umleiten. Wenn Sie alle Anforderungen umleiten, werden alle Anforderungen an den Website-Endpunkt an den angegebenen Hostnamen umgeleitet.

Beispiel: Wenn Ihre Root-Domain `example.com` ist, und Sie Anfragen sowohl für `http://example.com`, als auch für `http://www.example.com` bedienen möchten, erstellen Sie zwei Buckets namens `example.com` und `www.example.com`. Pflegen Sie dann den Inhalt im Bucket `example.com` und konfigurieren Sie den anderen Bucket `www.example.com` so, dass alle Anforderungen an den Bucket `example.com` umgeleitet werden. Weitere Informationen finden Sie unter Konfigurieren einer statischen Website mit einem benutzerdefinierten Domänennamen.

So leiten Sie Anforderungen für einen Bucket-Website-Endpunkt um

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben (z. B. `example.com`).
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Wählen Sie Redirect requests (Anforderungen umleiten).
6. Geben Sie im Feld Target bucket or domain (Ziel-Bucket oder -Domäne) den Bucket oder die Domäne ein, zu der Sie umleiten möchten.

   Wenn Sie Anforderungen beispielsweise zu einer Root-Domänenadresse umleiten, geben Sie **`example.com`** ein.
7. Geben Sie im Feld Protocol (Protokoll) das Protokoll für die umgeleiteten Anforderungen (**`http`** oder **`https`**) ein.

   Wenn Sie kein Protokoll angeben, wird das Protokoll der ursprünglichen Anforderung verwendet.
8. Wählen Sie Save (Speichern) aus.

Konfigurieren erweiterter bedingter Umleitungen

Unter Verwendung erweiterter Umleitungsregeln können Sie bedingt Anfragen abhängig von bestimmten Objektschlüsselnamen oder Präfixen in der Anfrage weiterleiten, oder abhängig von Antwortcodes. Angenommen, Sie löschen ein Objekt in Ihrem Bucket oder benennen es um. Sie können eine Weiterleitungsregel hinzufügen, die die Anfrage an ein anderes Objekt weiterleitet. Wenn Sie einen Ordner nicht zur Verfügung stellen wollen, können Sie eine Umleitungsregel hinzufügen, um die Anfrage an eine andere Webseite umzuleiten. Sie können auch eine Weiterleitungsregel hinzufügen, um Fehlerbedingungen zu verarbeiten, indem Sie Anfragen, die den Fehler zurückgeben, an eine andere Domäne weiterleiten, wenn der Fehler verarbeitet wird.

Wenn Sie einen Bucket für das Website-Hosting konfigurieren, haben Sie die Möglichkeit, erweiterte Umleitungsregeln anzugeben. Amazon S3 hat eine Begrenzung von 50 Routingregeln pro Websitekonfiguration. Wenn Sie mehr als 50 Routingregeln benötigen, können Sie die Objektumleitung verwenden. Weitere Informationen finden Sie unter (Optional) Konfigurieren einer Webseiten-Umleitung (p. 614).

So konfigurieren Sie Umleitungsregeln für eine statische Website

Gehen Sie folgendermaßen vor, um Umleitungsregeln für einen Bucket hinzuzufügen, für den das statische Website-Hosting bereits aktiviert ist.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets (Buckets) den Namen eines Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Geben Sie in Redirection rules (Umleitungsregeln) Ihre Umleitungsregeln ein.

   Sie schreiben die Regeln mit XML. Die allgemeine Syntax und Beispiele für die Angabe von Umleitungsregeln finden Sie unter Syntax für die Angabe von Weiterleitungsregeln (p. 618). In Amazon S3 gilt eine Begrenzung von 50 Routing-Regeln pro Website-Konfiguration. Wenn Sie mehr als 50 Routingregeln benötigen, können Sie die Objektumleitung verwenden. Weitere Informationen finden Sie unter Einrichten einer Seitenumleitung in der Amazon S3-Konsole (p. 615).
6. Wählen Sie Save aus.

Syntax für die Angabe von Weiterleitungsregeln

Im Folgenden finden Sie eine allgemeine Syntax zur Definition der Routingregeln in einer Website-Konfiguration.

```
<RoutingRules> =
    <RoutingRules>
         <RoutingRule>...</RoutingRule>
         [<RoutingRule>...</RoutingRule>
         ...]
    </RoutingRules>

<RoutingRule> =
   <RoutingRule>
      [ <Condition>...</Condition> ]
      <Redirect>...</Redirect>
   </RoutingRule>

<Condition> =
   <Condition>
      [ <KeyPrefixEquals>...</KeyPrefixEquals> ]
      [ <HttpErrorCodeReturnedEquals>...</HttpErrorCodeReturnedEquals> ]
   </Condition>
    Note: <Condition> must have at least one child element.

<Redirect> =
   <Redirect>
      [ <HostName>...</HostName> ]
      [ <Protocol>...</Protocol> ]
      [ <ReplaceKeyPrefixWith>...</ReplaceKeyPrefixWith>  ]
      [ <ReplaceKeyWith>...</ReplaceKeyWith> ]
      [ <HttpRedirectCode>...</HttpRedirectCode> ]
   </Redirect>
    Note: <Redirect> must have at least one child element.
           Also, you can have either ReplaceKeyPrefix with or ReplaceKeyWith,
           but not both.
```

In der folgenden Tabelle werden die Elemente in der Weiterleitungsregel beschrieben.

Name	Beschreibung
`RoutingRules`	Container für eine Sammlung von `RoutingRule`-Elementen.
`RoutingRule`	Eine Regel, die eine Bedingung identifiziert, sowie die Weiterleitung, die angewendet wird, wenn die Bedingung erfüllt ist. Bedingung: • Ein `RoutingRules`-Container muss mindestens eine Weiterleitungsregel enthalten.
`Condition`	Container für die Beschreibung einer Bedingung, die für die angegebene Weiterleitung erfüllt sein muss, damit sie angewendet wird. Wenn die Weiterleitungsregel keine Bedingung enthält, wird die Regel auf alle Anfragen angewendet.
`KeyPrefixEquals`	Das Präfix des Objektschlüsselnamens, von dem die Anfragen weitergeleitet werden. `KeyPrefixEquals` ist erforderliche, wenn `HttpErrorCodeReturnedEquals` nicht angegeben ist. Wenn

Name	Beschreibung
	`KeyPrefixEquals` und `HttpErrorCodeReturnedEquals` angegeben sind, müssen beide true sein, damit die Bedingung erfüllt ist.
`HttpErrorCodeReturnedEquals`	Der HTTP-Fehlercode, mit dem eine Übereinstimmung vorliegen muss, damit die Umleitung angewendet wird. Wenn ein Fehler auftritt und der Fehlercode mit diesem Wert übereinstimmt, gilt die angegebene Weiterleitung. `HttpErrorCodeReturnedEquals` ist erforderliche, wenn `KeyPrefixEquals` nicht angegeben ist. Wenn `KeyPrefixEquals` und `HttpErrorCodeReturnedEquals` angegeben sind, müssen beide true sein, damit die Bedingung erfüllt ist.
`Redirect`	Container-Element, das Anweisungen für die Weiterleitung der Anfrage enthält. Sie können Anfragen an einen anderen Host oder eine andere Seite umleiten, oder ein anderes Protokoll vorgeben. Eine `RoutingRule` muss ein `Redirect`-Element besitzen. Ein `Redirect`-Element muss mindestens eines der folgenden zugeordneten Elemente enthalten: `Protocol`, `HostName`, `ReplaceKeyPrefixWith`, `ReplaceKeyWith` oder `HttpRedirectCode`.
`Protocol`	Das Protokoll, `http` oder `https`, das im `Location`-Header verwendet werden soll, der in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `Protocol` nicht erforderlich.
`HostName`	Der Hostname, der im `Location`-Header verwendet werden soll, der in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `HostName` nicht erforderlich.
`ReplaceKeyPrefixWith`	Das Präfix des Objektschlüsselnamens, der den Wert von `KeyPrefixEquals` in der Umleitungsanforderung ersetzt. Wenn eines der zugeordneten Elemente vorhanden ist, ist `ReplaceKeyPrefixWith` nicht erforderlich. Es kann nur bereitgestellt werden, wenn `ReplaceKeyWith` nicht bereitgestellt wird.
`ReplaceKeyWith`	Der Objektschlüssel, der im `Location`-Header verwendet werden soll und in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `ReplaceKeyWith` nicht erforderlich. Es kann nur bereitgestellt werden, wenn `ReplaceKeyPrefixWith` nicht bereitgestellt wird.
`HttpRedirectCode`	Der HTTP-Umleitungscode, der im `Location`-Header verwendet werden soll, der in der Antwort zurückgegeben wird. Wenn eines der zugeordneten Elemente vorhanden ist, ist `HttpRedirectCode` nicht erforderlich.

Beispiele

Die folgenden Beispiele erklären häufige Weiterleitungsfälle:

Example 1: Weiterleitung nach der Umbenennung eines Schlüsselpräfix

Angenommen, Ihr Bucket enthält die folgenden Objekte:

- index.html
- docs/article1.html
- docs/article2.html

Jetzt wollen Sie den Ordner von `docs/` in `documents/` umbenennen. Nach dieser Änderung müssen Sie Anfragen für das Präfix `docs/` in `documents/` weiterleiten. Beispielsweise muss die Anfragen für `docs/article1.html` an `documents/article1.html` weitergeleitet werden.

In diesem Fall fügen Sie der Konfiguration der Website die folgende Routingregel hinzu.

```
<RoutingRules>
  <RoutingRule>
  <Condition>
    <KeyPrefixEquals>docs/</KeyPrefixEquals>
  </Condition>
  <Redirect>
    <ReplaceKeyPrefixWith>documents/</ReplaceKeyPrefixWith>
  </Redirect>
  </RoutingRule>
</RoutingRules>
```

Example 2: Weiterleitung von Anfragen für einen gelöschten Ordner auf eine Seite

Angenommen, Sie löschen den Ordner `images/` (d. h. Sie löschen alle Objekte mit dem Schlüsselpräfix `images/`). Sie können eine Weiterleitungsregel einrichten, die Anfrage für jedes Objekt mit dem Schlüsselpräfix `images/` auf eine Seite namens `folderdeleted.html` weiterleitet.

```
<RoutingRules>
  <RoutingRule>
  <Condition>
     <KeyPrefixEquals>images/</KeyPrefixEquals>
  </Condition>
  <Redirect>
    <ReplaceKeyWith>folderdeleted.html</ReplaceKeyWith>
  </Redirect>
  </RoutingRule>
</RoutingRules>
```

Example 3: Weiterleitung für einen HTTP-Fehler.

Angenommen, Sie möchten Anfragen an eine Amazon Elastic Compute Cloud (Amazon EC2)-Instance weiterleiten, wenn ein angefordertes Objekt nicht gefunden wurde. Fügen Sie eine Umleitungsregel hinzu, sodass, wenn ein HTTP-Statuscode 404 (Not Found) zurückgegeben wird, der Besucher der Website zu einer Amazon EC2-Instance umgeleitet wird, die die Anfrage bearbeitet.

Im folgenden Beispiel wird auch das Objektschlüsselpräfix `report-404/` in die Weiterleitung eingefügt. Wenn Sie beispielsweise eine Seite mit der Bezeichnung `ExamplePage.html` anfordern und ein HTTP-404-Fehler ausgegeben wird, wird die Anfrage an eine Seite mit dem Namen `report-404/ExamplePage.html` auf der angegebenen Amazon EC2-Instance weitergeleitet. Wenn es keine

Weiterleitungsregel gibt und der HTTP-Fehler 404 auftritt, wird das in der Konfiguration festgelegte Fehlerdokument zurückgegeben.

```
<RoutingRules>
  <RoutingRule>
  <Condition>
    <HttpErrorCodeReturnedEquals>404</HttpErrorCodeReturnedEquals >
  </Condition>
  <Redirect>
    <HostName>ec2-11-22-333-44.compute-1.amazonaws.com</HostName>
    <ReplaceKeyPrefixWith>report-404/</ReplaceKeyPrefixWith>
  </Redirect>
  </RoutingRule>
</RoutingRules>
```

Programmgesteuertes Konfigurieren eines Buckets als statische Website

Um eine statische Website auf Amazon S3 zu hosten, konfigurieren Sie einen Amazon S3-Bucket für ein Website-Hosting und laden dann Ihren Website-Inhalt in den Bucket hoch. Sie können die AWS SDKs auch verwenden, um die Websitekonfiguration programmgesteuert zu erstellen, zu aktualisieren und zu löschen. Die SDKs stellen Wrapper-Klassen für die Amazon S3-REST-APIs bereit. Falls in Ihrer Anwendung erforderlich, können Sie auch direkt von Ihrer Anwendung aus REST API-Anfragen senden.

Weitere Informationen zum Konfigurieren des Buckets für statisches Website-Hosting mithilfe der AWS Management Console finden Sie unter Konfigurieren eines Buckets als statische Website mithilfe der AWS Management Console (p. 605).

Weitere Informationen zum Verwenden von AWS CLI zum Konfigurieren eines S3-Buckets als statische Website finden Sie unter Website im AWS CLI Command Reference. Weitere Informationen zum programmgesteuerten Konfigurieren eines S3-Buckets als statische Website finden Sie in den folgenden Themen.

Themen

Verwalten von Websites mit dem AWS SDK for Java

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK for Java die Website-Konfiguration für einen Bucket verwalten. Um einem Bucket eine Website-Konfiguration hinzuzufügen, geben Sie einen Bucket-Namen und eine Website-Konfiguration an. Die Website-Konfiguration muss ein Indexdokument enthalten und kann optional ein Fehlerdokument enthalten. Diese Dokumente müssen bereits im Bucket vorhanden sein. Weitere Informationen finden Sie unter PUT Bucket-Website. Weitere Informationen zur Amazon S3 Website-Funktion finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).

Example

Das folgende Beispiel fügt mit AWS SDK for Java einem Bucket eine Website-Konfiguration hinzu, ruft die Konfiguration ab und gibt die Antwort aus, löscht dann die Konfiguration und überprüft den Löschvorgang.

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketWebsiteConfiguration;

import java.io.IOException;

public class WebsiteConfiguration {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String indexDocName = "*** Index document name ***";
        String errorDocName = "*** Error document name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Print the existing website configuration, if it exists.
            printWebsiteConfig(s3Client, bucketName);

            // Set the new website configuration.
            s3Client.setBucketWebsiteConfiguration(bucketName, new
 BucketWebsiteConfiguration(indexDocName, errorDocName));

            // Verify that the configuration was set properly by printing it.
            printWebsiteConfig(s3Client, bucketName);

            // Delete the website configuration.
            s3Client.deleteBucketWebsiteConfiguration(bucketName);

            // Verify that the website configuration was deleted by printing it.
            printWebsiteConfig(s3Client, bucketName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void printWebsiteConfig(AmazonS3 s3Client, String bucketName) {
        System.out.println("Website configuration: ");
        BucketWebsiteConfiguration bucketWebsiteConfig =
 s3Client.getBucketWebsiteConfiguration(bucketName);
        if (bucketWebsiteConfig == null) {
            System.out.println("No website config.");
        } else {
            System.out.println("Index doc: " +
 bucketWebsiteConfig.getIndexDocumentSuffix());
            System.out.println("Error doc: " + bucketWebsiteConfig.getErrorDocument());
        }
```

```
    }
}
```

Verwalten von Websites mit dem AWS SDK für .NET

Das folgende Beispiel veranschaulicht, wie Sie mit AWS SDK für .NET die Website-Konfiguration für einen Bucket verwalten. Um einem Bucket eine Website-Konfiguration hinzuzufügen, geben Sie einen Bucket-Namen und eine Website-Konfiguration an. Die Website-Konfiguration muss ein Indexdokument enthalten und kann optional ein Fehlerdokument enthalten. Diese Dokumente müssen im Bucket gespeichert sein. Weitere Informationen finden Sie unter PUT Bucket-Website. Weitere Informationen zur Amazon S3 Website-Funktion finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).

Example

Das folgende C#-Code-Beispiel fügt dem angegebenen Bucket eine Website-Konfiguration hinzu. Die Konfiguration gibt die Namen für das Indexdokument und das Fehlerdokument an. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class WebsiteConfigTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string indexDocumentSuffix = "*** index object key ***"; // For
 example, index.html.
        private const string errorDocument = "*** error object key ***"; // For example,
 error.html.
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            AddWebsiteConfigurationAsync(bucketName, indexDocumentSuffix,
 errorDocument).Wait();
        }

        static async Task AddWebsiteConfigurationAsync(string bucketName,
                                            string indexDocumentSuffix,
                                            string errorDocument)
        {
            try
            {
                // 1. Put the website configuration.
                PutBucketWebsiteRequest putRequest = new PutBucketWebsiteRequest()
                {
                    BucketName = bucketName,
                    WebsiteConfiguration = new WebsiteConfiguration()
                    {
                        IndexDocumentSuffix = indexDocumentSuffix,
                        ErrorDocument = errorDocument
                    }
                };
```

```
                PutBucketWebsiteResponse response = await
 client.PutBucketWebsiteAsync(putRequest);

                // 2. Get the website configuration.
                GetBucketWebsiteRequest getRequest = new GetBucketWebsiteRequest()
                {
                    BucketName = bucketName
                };
                GetBucketWebsiteResponse getResponse = await
 client.GetBucketWebsiteAsync(getRequest);
                Console.WriteLine("Index document: {0}",
 getResponse.WebsiteConfiguration.IndexDocumentSuffix);
                Console.WriteLine("Error document: {0}",
 getResponse.WebsiteConfiguration.ErrorDocument);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Verwalten von Websites mit dem AWS SDK für PHP

Dieses Thema beschreibt die Verwendung von AWS SDK für PHP-Klassen für das Konfigurieren und Verwalten eines Amazon S3-Buckets für das Website-Hosting. Es wird vorausgesetzt, dass Sie den Anleitungen für Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812) folgen und der AWS SDK für PHP ordnungsgemäß installiert ist. Weitere Informationen zur Amazon S3-Website-Funktion finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).

Das folgende PHP-Beispiel fügt dem angegebenen Bucket eine Website-Konfiguration hinzu. Die Methode `create_website_config` stellt explizit die Namen des Indexdokuments und des Fehlerdokuments bereit. Das Beispiel ruft auch die Website-Konfiguration ab und gibt die Antwort aus. Weitere Informationen zur Amazon S3-Website-Funktion finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).

Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen (p. 812).

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);


// Add the website configuration.
$s3->putBucketWebsite([
    'Bucket'                => $bucket,
    'WebsiteConfiguration'  => [
```

```
        'IndexDocument' => ['Suffix' => 'index.html'],
        'ErrorDocument' => ['Key' => 'error.html']
    ]
]);

// Retrieve the website configuration.
$result = $s3->getBucketWebsite([
    'Bucket' => $bucket
]);
echo $result->getPath('IndexDocument/Suffix');

// Delete the website configuration.
$s3->deleteBucketWebsite([
    'Bucket' => $bucket
]);
```

Zugehörige Ressourcen

- AWS SDK für PHP für Amazon S3 Aws\S3\S3Client Class
- AWS SDK für PHP-Dokumentation

Verwalten von Websites mit der REST API

Sie können die AWS Management Console oder das AWS-SDK für die Konfiguration eines Buckets als Website verwenden. Falls in Ihrer Anwendung jedoch erforderlich, können Sie REST-Anforderungen auch direkt senden. Weitere Informationen finden Sie in der folgenden Abschnitten im Amazon Simple Storage Service API Reference.

- PUT Bucket-Website
- GET Bucket-Website
- DELETE Bucket-Website

Beispielhafte Walkthroughs – Hosting von Websites auf Amazon S3

In diesem Abschnitt finden Sie zwei Beispiele. Im ersten Beispiel konfigurieren Sie einen Bucket für das Website-Hosting, laden ein Beispiel-Indexdokument hoch und testen den Amazon S3-Website-Endpunkt für den Bucket. Das zweite Beispiel zeigt, wie Sie Ihre eigene Domäne, wie z. B. `example.com`, anstelle des Amazon S3-Endpunkts der Bucket-Website verwenden und Inhalte aus einem als Website konfigurierten Amazon S3-Bucket bereitstellen können. Das Beispiel zeigt auch, wie Amazon S3 die Unterstützung von Root-Domänen anbietet.

Themen

- Konfigurieren einer statischen Website (p. 625)
- Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten, bei Route 53 registrierten Domäne (p. 631)

Konfigurieren einer statischen Website

Sie können einen Amazon S3-Bucket so konfigurieren, dass er sich wie eine Website verhält. In diesem Beispiel sehen Sie die Schritte für das Hosten einer Website auf Amazon S3.

Themen

Schritt 1: Erstellen eines Buckets

Die folgenden Anweisungen geben einen Überblick darüber, wie Sie Ihre Buckets für das Website-Hosting erstellen. Ausführliche Schritt-für-Schritt-Anleitungen zum Erstellen eines Buckets finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

So erstellen Sie einen Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie Create Bucket (Bucket erstellen) aus.
3. Geben Sie den Bucket name (Bucket-Namen) ein (z. B. `example.com`).
4. Wählen Sie die Region aus, in der Sie Ihren Bucket erstellen möchten.

   Wählen Sie eine Region in der Nähe aus, um Latenz und Kosten auf einem Minimum zu halten oder behördliche Vorschriften zu erfüllen. Die von Ihnen ausgewählte Region bestimmt Ihren Amazon S3-Website-Endpunkt. Weitere Informationen finden Sie unter Website-Endpunkte (p. 602).
5. Um die Standardeinstellungen zu übernehmen und den Bucket zu erstellen, wählen Sie Create (Erstellen).

Schritt 2: Aktivieren des statischen Website-Hostings

Nach der Erstellung eines Buckets können Sie das statische Website-Hosting für Ihren Bucket aktivieren. Sie können einen neuen Bucket erstellen oder einen vorhandenen Bucket verwenden.

So aktivieren Sie das statische Website-Hosting

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket name (Bucket-Name) den Bucket aus, den Sie für Ihre statische Website verwenden möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Wählen Sie Use this bucket to host a website (Diesen Bucket zum Hosten einer Website verwenden).
6. Geben Sie den Namen Ihres Index-Dokuments ein.

   Der Name des Indexdokuments lautet in der Regel `index.html`. Der Name des Indexdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Indexdokuments übereinstimmen, das Sie in den S3-Bucket hochladen möchten. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments (p. 606).

7. (Optional) Wenn Sie ein benutzerdefiniertes Fehlerdokument hinzufügen möchten, geben Sie im Feld Error document (Fehlerdokument) den Schlüsselnamen des Fehlerdokuments ein (z. B. **`error.html`**).

   Der Name des Fehlerdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Fehlerdokuments übereinstimmen, das Sie in Ihren S3-Bucket hochladen möchten. Weitere Informationen finden Sie unter (Optional) Konfigurierung eines benutzerdefinierten Fehlerdokuments (p. 612).
8. (Optional) Wenn Sie erweiterte Umleitungsregeln angeben möchten, verwenden Sie unter Edit redirection rules (Umleitungsregeln bearbeiten) XML zur Beschreibung der Regeln.

   Weitere Informationen finden Sie unter Konfigurieren erweiterter bedingter Umleitungen (p. 617).
9. Notieren Sie unter Static website hosting (Statisches Website-Hosting) den Wert für Endpoint (Endpunkt).

   Der Endpoint (Endpunkt) ist der Amazon S3-Website-Endpunkt für Ihren Bucket. Nachdem Sie den Bucket als statische Website konfiguriert haben, können Sie diesen Endpunkt verwenden, um Ihre Website zu testen.
10. Wählen Sie Save (Speichern) aus.

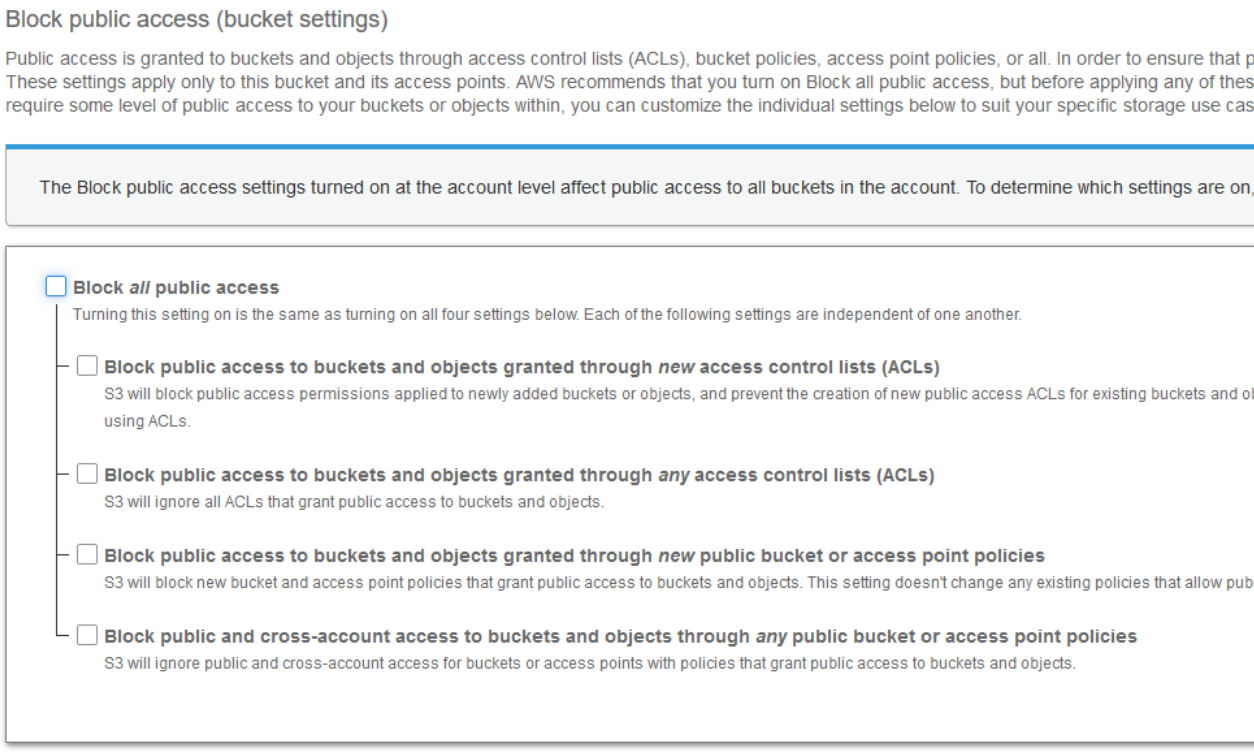
Schritt 3: Bearbeiten der Block Public Access-Einstellungen

Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

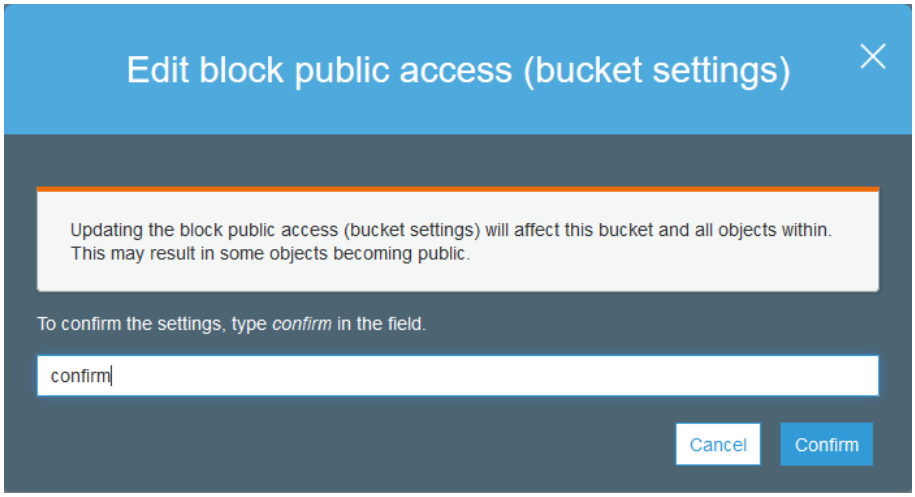
> Warning
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Verwenden von Amazon S3 Block Public, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren), und wählen Sie Save (Speichern).

   > Warning
   >
   > Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Verwenden von Amazon S3Block Public, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



6. Geben Sie im Bestätigungsfeld **`confirm`** ein, und wählen Sie dann Confirm (Bestätigen).



Unter S3-Buckets wird der Access (Zugriff) auf Ihren Bucket auf Objects can be public (Objekte können öffentlich sein) aktualisiert. Sie können nun eine Bucket-Richtlinie hinzufügen, um die Objekte im Bucket öffentlich lesbar zu machen. Wenn der Access (Zugriff) immer noch als Bucket and objects not public (Bucket und Objekte nicht öffentlich) angezeigt wird, müssen Sie möglicherweise die Einstellungen für die Blockierung des öffentlichen Zugriff für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen.

Schritt 4: Hinzufügen einer Bucket-Richtlinie, die den Inhalt Ihres Buckets öffentlich verfügbar macht

Nachdem Sie die Einstellungen für S3 Block Public Access bearbeitet haben, können Sie eine Bucket-Richtlinie hinzufügen, um öffentlichen Lesezugriff auf den Bucket zu gewähren. Wenn Sie öffentlichen Lesezugriff gewähren, kann jeder im Internet auf Ihren Bucket zugreifen.

> **Important**
>
> Die zuvor genannte Richtlinie ist nur ein Beispiel und erlaubt Vollzugriff auf die Inhalte Ihres Buckets. Bevor Sie mit diesem Schritt fortfahren, lesen Sie den Abschnitt Wie kann ich die Dateien in meinem Amazon S3-Bucket sichern? , um sicherzustellen, dass Sie die bewährten Methoden zum Sichern der Dateien in Ihrem S3-Bucket und die Risiken in Zusammenhang mit der Gewährung von öffentlichem Zugriff kennen.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Permissions.
3. Wählen Sie Bucket Policy aus.
4. Um öffentlichen Lesezugriff auf Ihre Website zu gewähren, kopieren Sie die folgende Bucket-Richtlinie und fügen Sie sie in den Bucket policy editor (Bucket-Richtlinieneditor) ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::example.com/*"
            ]
        }
    ]
}
```

5. Aktualisieren Sie den `Resource`, um Ihren Bucket-Namen anzugeben.

   In der vorangegangenen Beispiel-Bucket-Richtlinie ist *`example.com`* der Bucket-Name. Um diese Bucket-Richtlinie mit Ihrem eigenen Bucket zu verwenden, müssen Sie diesen Namen so aktualisieren, dass er mit Ihrem Bucket übereinstimmt.
6. Wählen Sie Save (Speichern) aus.

   Es wird eine Warnung angezeigt, die darauf hinweist, dass der Bucket öffentlich zugänglich ist. Unter Bucket Policy (Bucket-Richtlinie) wird die Bezeichnung Public (Öffentlich) angezeigt.

   Wenn die Fehlermeldung `Policy has invalid resource` angezeigt wird, bestätigen Sie, dass der Bucket-Name in der Bucket-Richtlinie mit Ihrem Bucket-Namen übereinstimmt. Informationen zum Hinzufügen einer Bucket-Richtlinie finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu?

   Wenn Sie die Warnung Error - Access denied (Fehler - Zugriff verweigert) erhalten und der Bucket policy editor (Bucket-Richtlinieneditor) das Speichern der Bucket-Richtlinie nicht zulässt, überprüfen Sie die Block Public Access-Einstellungen auf Kontoebene und Bucket-Ebene, um zu bestätigen, dass Sie öffentlichen Zugriff auf den Bucket zulassen. Weitere Informationen zu Website-Berechtigungen finden Sie unter Erforderliche Berechtigungen für den Website-Zugriff.

Schritt 5: Konfigurieren eines Indexdokuments

Wenn Sie das Hosting statischer Websites für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B. **`index.html`**). Nachdem Sie das Hosting statischer Websites für den Bucket aktiviert haben, laden Sie eine HTML-Datei mit diesem Indexdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie das Indexdokument

1. Erstellen Sie eine Datei `index.html`.

   Wenn Sie nicht über eine Datei `index.html` verfügen, können Sie mit dem folgenden HTML-Code eine Datei erstellen:

   ```
   <html xmlns="http://www.w3.org/1999/xhtml" >
   <head>
       <title>My Website Home Page</title>
   </head>
   <body>
     <h1>Welcome to my website</h1>
     <p>Now hosted on Amazon S3!</p>
   </body>
   </html>
   ```

2. Speichern Sie die Indexdatei lokal und notieren Sie sich den Dateinamen (z. B. `index.html`).

   Wenn Sie das Hosting statischer Websites aktivieren und Ihren Indexdokumentnamen eingeben, geben Sie diesen exakten Dateinamen ein (z. B. `index.html`).
3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Indexdokuments ein (z. B. `index.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings (p. 605).

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.
6. Führen Sie einen der folgenden Schritte aus, um das Indexdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Indexdatei per Drag & Drop in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   Schrittweise Anleitungen finden Sie unter Wie lade ich Dateien und Ordner in einen Amazon S3-Bucket hoch? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
7. (Optional) Laden Sie andere Website-Inhalte in Ihren Bucket hoch.

Schritt 6: Testen des Website-Endpunkts

Nach der Konfigurierung des statischen Website-Hostings für Ihren Bucket können Sie den Website-Endpunkt testen.

> Note
>
> Amazon S3 unterstützt keinen HTTPS-Zugriff auf die Website. Wenn Sie HTTPS verwenden möchten, können Sie mit Amazon CloudFront eine statische Website bereitstellen, die auf Amazon S3 gehostet wird.

Weitere Informationen finden Sie unter How do I use CloudFront to serve a static website hosted on Amazon S3? und Requiring HTTPS for Communication Between CloudFront and Your Amazon S3 Origin.

So testen Sie Ihren Website-Endpunkt

Wenn Sie beim Aktivieren des statischen Website-Hosting Ihren Website-Endpunkt notiert haben, geben Sie den Website-Endpunkt zum Testen Ihrer Website in Ihrem Browser ein. Wenn Ihr Browser Ihre `index.html`-Seite anzeigt, wurde die Website erfolgreich bereitgestellt. Weitere Informationen finden Sie unter Amazon S3-Website-Endpunkte.

Wenn Sie Ihren Website-Endpunkt vor dem Testen abrufen müssen, gehen Sie folgendermaßen vor:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Um Ihren Website-Endpunkt zu testen, wählen Sie neben Endpoint (Endpunkt) Ihren Website-Endpunkt aus.

   Wenn Ihr Browser Ihre `index.html`-Seite anzeigt, wurde die Website erfolgreich bereitgestellt.

Sie haben jetzt eine in Amazon S3 gehostete Website. Diese Website steht am Amazon S3-Website-Endpunkt zur Verfügung. Möglicherweise haben Sie jedoch eine Domäne wie `example.com`, die den Inhalt von der von Ihnen erstellten Website bereitstellen soll. Sie könnten auch die Amazon S3-Unterstützung der Root-Domäne nutzen, um Anfragen für `http://www.example.com` und `http://example.com` zu bedienen. Dafür sind zusätzliche Schritte erforderlich. Ein Beispiel finden Sie unter Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten, bei Route 53 registrierten Domäne (p. 631).

Schritt 7: Bereinigen

Wenn Sie die statische Website nur zur Übung erstellt haben, löschen Sie die AWS-Ressourcen, die Sie zugewiesen haben, damit keine weiteren Kosten für Sie anfallen. Nach der Löschung der AWS-Ressourcen ist die Website nicht mehr verfügbar. Weitere Informationen finden Sie unter Wie lösche ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Konfigurieren einer statischen Website mithilfe einer benutzerdefinierten, bei Route 53 registrierten Domäne

Angenommen, Sie wollen Ihre statische Website auf Amazon S3 hosten. Sie haben eine Domäne bei Amazon Route 53 registriert (z. B. `example.com`) und wollen, dass Anfragen für `http://www.example.com` und `http://example.com` von Ihrem Amazon S3-Inhalt abgearbeitet werden. Sie können diese schrittweise Anleitung verwenden, um zu erfahren, wie Sie eine statische Website hosten und Umleitungen auf Amazon S3 für eine Website mit einem benutzerdefinierten Domänennamen, der bei Route 53 registriert ist, anlegen können. Sie können mit einer vorhandenen Website arbeiten, die Sie auf Amazon S3 hosten möchten, oder Sie verwenden diese schrittweise Anleitung, um bei Null anzufangen.

Nach Abschluss dieses Walkthrough können Sie optional Amazon CloudFront verwenden, um die Leistung Ihrer Website zu verbessern. Weitere Informationen finden Sie unter Beschleunigung Ihrer Website mit Amazon CloudFront (p. 641).

Themen

Bevor Sie beginnen

Während Sie den Schritten in diesem Beispiel folgen, arbeiten Sie mit den folgenden Services:

Amazon Route 53 – Sie verwenden Route 53, um Domänen zu registrieren und definieren, wohin der Internetdatenverkehr für Ihre Domäne geleitet werden soll. Das Beispiel zeigt, wie Sie Route 53-Alias-Datensätze erstellen, mit denen Sie Datenverkehr für Ihre Domäne (`example.com`) und Subdomäne (`www.example.com`) in einen Amazon S3-Bucket umleiten, der eine HTML-Datei enthält.

Amazon S3 – Mit Amazon S3 erstellen Sie Buckets, laden eine Beispielseite für eine Website hoch, konfigurieren Berechtigungen, so dass jeder den Inhalt sehen kann, und konfigurieren dann die Buckets für das Website-Hosting.

Schritt 1: Registrieren einer benutzerdefinierten Domäne bei Route 53

Wenn Sie noch keinen registrierten Domänennamen haben, wie z. B. `example.com`, registrieren Sie einen mit Route 53. Weitere Informationen finden Sie unter Registrierung einer neuen Domäne im Entwicklerhandbuch für Amazon Route 53. Nachdem Sie Ihren Domänennamen registriert haben, können Sie Ihre Amazon S3-Buckets für das Website-Hosting erstellen und konfigurieren.

Schritt 2: Erstellen von zwei Buckets

Um Anforderungen von sowohl der Root-Domäne als auch von der Subdomäne zu unterstützen, erstellen Sie zwei Buckets:

- Domänen-Bucket - `example.com`
- Subdomänen-Bucket - `www.example.com`

Sie hosten Ihre Inhalte aus dem Stammdomänen-Bucket (`example.com`). Sie erstellen eine Umleitungsanforderung für den Subdomänen-Bucket (`www.example.com`). Falls ein Benutzer `www.example.com` in seinen Browser eingibt, wird er zu `example.com` umgeleitet und sieht den Inhalt, der im Amazon S3-Bucket mit diesem Namen gehostet wird.

So erstellen Sie Ihre Buckets für Website-Hosting

Die folgenden Anweisungen geben einen Überblick darüber, wie Sie Ihre Buckets für das Website-Hosting erstellen. Ausführliche Schritt-für-Schritt-Anleitungen zum Erstellen eines Buckets finden Sie unter Wie erstelle ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Erstellen Sie den Stammdomänen-Bucket:
   a. Wählen Sie Create Bucket (Bucket erstellen) aus.
   b. Geben Sie den Bucket name (Bucket-Namen) ein (z. B. `example.com`).
   c. Wählen Sie die Region aus, in der Sie Ihren Bucket erstellen möchten.

      Wählen Sie eine Region in der Nähe aus, um Latenz und Kosten auf einem Minimum zu halten oder behördliche Vorschriften zu erfüllen. Die von Ihnen ausgewählte Region bestimmt Ihren Amazon S3-Website-Endpunkt. Weitere Informationen finden Sie unter Website-Endpunkte (p. 602).
   d. Um die Standardeinstellungen zu übernehmen und den Bucket zu erstellen, wählen Sie Create (Erstellen).
3. Erstellen Sie Ihren Subdomänen-Bucket:
   a. Wählen Sie Create Bucket (Bucket erstellen) aus.
   b. Geben Sie den Bucket name (Bucket-Namen) ein (z. B. `www.example.com`).
   c. Wählen Sie die Region aus, in der Sie Ihren Bucket erstellen möchten.

      Wählen Sie eine Region in der Nähe aus, um Latenz und Kosten auf einem Minimum zu halten oder behördliche Vorschriften zu erfüllen. Die von Ihnen ausgewählte Region bestimmt Ihren Amazon S3-Website-Endpunkt. Weitere Informationen finden Sie unter Website-Endpunkte (p. 602).
   d. Um die Standardeinstellungen zu übernehmen und den Bucket zu erstellen, wählen Sie Create (Erstellen).

Im nächsten Schritt konfigurieren Sie `example.com` für das Website-Hosting.

Schritt 3: Konfigurieren Ihres Root-Domänen-Buckets für Website-Hosting

In diesem Schritt konfigurieren Sie Ihren Stammdomänen-Bucket (`example.com`) als Website. Dieser Bucket enthält Ihren Website-Inhalt. Wenn Sie einen Bucket für das Website-Hosting konfigurieren, können Sie über Website-Endpunkte (p. 602) auf die Website zugreifen.

So aktivieren Sie das statische Website-Hosting

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Bucket name (Bucket-Name) den Bucket aus, den Sie für Ihre statische Website verwenden möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Wählen Sie Use this bucket to host a website (Diesen Bucket zum Hosten einer Website verwenden).
6. Geben Sie den Namen Ihres Index-Dokuments ein.

Der Name des Indexdokuments lautet in der Regel `index.html`. Der Name des Indexdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Indexdokuments übereinstimmen, das Sie in den S3-Bucket hochladen möchten. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments (p. 606).

7. (Optional) Wenn Sie ein benutzerdefiniertes Fehlerdokument hinzufügen möchten, geben Sie im Feld Error document (Fehlerdokument) den Schlüsselnamen des Fehlerdokuments ein (z. B. **`error.html`**).

   Der Name des Fehlerdokuments unterscheidet Groß- und Kleinschreibung und muss genau mit dem Dateinamen des HTML-Fehlerdokuments übereinstimmen, das Sie in Ihren S3-Bucket hochladen möchten. Weitere Informationen finden Sie unter (Optional) Konfigurierung eines benutzerdefinierten Fehlerdokuments (p. 612).
8. (Optional) Wenn Sie erweiterte Umleitungsregeln angeben möchten, verwenden Sie unter Edit redirection rules (Umleitungsregeln bearbeiten) XML zur Beschreibung der Regeln.

   Weitere Informationen finden Sie unter Konfigurieren erweiterter bedingter Umleitungen (p. 617).
9. Notieren Sie unter Static website hosting (Statisches Website-Hosting) den Wert für Endpoint (Endpunkt).

   Der Endpoint (Endpunkt) ist der Amazon S3-Website-Endpunkt für Ihren Bucket. Nachdem Sie den Bucket als statische Website konfiguriert haben, können Sie diesen Endpunkt verwenden, um Ihre Website zu testen.
10. Wählen Sie Save (Speichern) aus.

Im nächsten Schritt konfigurieren Sie Ihre Subdomäne (`www.example.com`) zur Weiterleitung von Anforderungen an Ihre Domäne (`example.com`).

Schritt 4: Konfigurieren Ihres Subdomänen-Buckets für die Website-Umleitung

Nachdem Sie Ihren Stammdomänen-Bucket für das Website-Hosting konfiguriert haben, können Sie Ihren Subdomänen-Bucket so konfigurieren, dass alle Anforderungen zu der Domäne umgeleitet werden. In diesem Beispiel werden alle Anforderungen für `www.example.com` an `example.com` umgeleitet.

So konfigurieren Sie eine Umleitungsanforderung

1. Wählen Sie in der Amazon S3-Konsole in der Liste Buckets (Buckets) Ihren Subdomänen-Bucket aus (in diesem Beispiel `www.example.com`).
2. Wählen Sie Properties (Eigenschaften).
3. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
4. Wählen Sie Redirect requests (Anforderungen umleiten).
5. Geben Sie Ihre Domäne (beispielweise **`example.com`**) in das Feld Target bucket or domain (Ziel-Bucket oder -Domäne) ein.
6. In das Feld Protocol (Protokoll) geben Sie **`http`** ein.
7. Wählen Sie Save (Speichern) aus.

Schritt 5: Konfigurieren der Protokollierung für Website-Datenverkehr

Wenn Sie die Anzahl der Besucher nachverfolgen möchten, die auf Ihre Website zugreifen, können Sie optional die Protokollierung für Ihren Stammdomänen-Bucket aktivieren. Weitere Informationen finden

Sie unter Amazon S3-Serverzugriffsprotokollierung. Wenn Sie jedoch planen, zur Beschleunigung Ihrer Website Amazon CloudFront zu verwenden, können Sie auch die CloudFront-Protokollierung verwenden.

So aktivieren Sie die Serverzugriffsprotokollierung für Ihren Stammdomänen-Bucket

1. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.
2. Erstellen Sie in derselben Region, in der Sie den Bucket erstellt haben, der als statische Website konfiguriert ist, einen Bucket für die Protokollierung, z. B. `logs.example.com`.
3. Erstellen Sie einen Ordner für die Protokolldateien der Serverzugriffsprotokollierung (z. B. `logs`).
4. (Optional) Wenn Sie die Leistung Ihrer Website mit CloudFront verbessern möchten, erstellen Sie einen Ordner für die CloudFront-Protokolldateien (z. B. `cdn`).
5. Wählen Sie in der Liste Bucket Ihren Stammdomänen-Bucket aus.
6. Wählen Sie Properties (Eigenschaften) aus.
7. Wählen Sie Server access logging (Serverzugriffsprotokollierung) aus.
8. Wählen Sie Enable logging (Protokollierung aktivieren) aus.
9. Wählen Sie unter Target Bucket (Ziel-Bucket) den Bucket aus, den Sie für die Protokolldateien erstellt haben, z. B. `logs.example.com`.
10. Geben Sie unter Target prefix (Zielpräfix) den Namen des Ordners ein, den Sie für die Protokolldateien erstellt haben, gefolgt von dem Trennzeichen (/), z. B. **`logs/`**.

    Wenn Sie das Target prefix (Zielpräfix) festlegen, gruppieren Sie die Protokolldatendateien in einem Ordner, sodass sie leicht zu finden sind.
11. Wählen Sie Save (Speichern) aus.

    In Ihrem Protokoll-Bucket können Sie nun auf Ihre Protokolle zugreifen. Amazon S3 schreibt alle 2 Stunden Website-Zugriffsprotokolle in Ihren Protokoll-Bucket.
12. Um die Protokolle anzuzeigen, wählen Sie Overview (Überblick) und den Ordner aus.

Schritt 6: Hochladen des Index und des Website-Inhalts

Als Nächstes laden Sie das Indexdokument und den optionalen Website-Inhalt zum Stammdomänen-Bucket hoch.

Wenn Sie das Hosting statischer Websites für Ihren Bucket aktivieren, geben Sie den Namen des Indexdokuments ein (z. B. **`index.html`**). Nachdem Sie das Hosting statischer Websites für den Bucket aktiviert haben, laden Sie eine HTML-Datei mit diesem Indexdokumentnamen in Ihren Bucket hoch.

So konfigurieren Sie das Indexdokument

1. Erstellen Sie eine Datei `index.html`.

   Wenn Sie nicht über eine Datei `index.html` verfügen, können Sie mit dem folgenden HTML-Code eine Datei erstellen:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>My Website Home Page</title>
</head>
<body>
  <h1>Welcome to my website</h1>
  <p>Now hosted on Amazon S3!</p>
</body>
</html>
```

2. Speichern Sie die Indexdatei lokal und notieren Sie sich den Dateinamen (z. B. `index.html`).

Wenn Sie das Hosting statischer Websites aktivieren und Ihren Indexdokumentnamen eingeben, geben Sie diesen exakten Dateinamen ein (z. B. `index.html`).

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
4. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
5. Aktivieren Sie das Hosting statischer Websites für Ihren Bucket und geben Sie den exakten Namen Ihres Indexdokuments ein (z. B. `index.html`). Weitere Informationen finden Sie unter Aktivieren des Website-Hostings (p. 605).

   Fahren Sie mit Schritt 6 fort, nachdem Sie das Hosting statischer Websites aktiviert haben.
6. Führen Sie einen der folgenden Schritte aus, um das Indexdokument in Ihren Bucket hochzuladen:

   - Ziehen Sie die Indexdatei per Drag & Drop in das Konsolen-Bucket-Verzeichnis.
   - Wählen Sie Upload (Hochladen) und folgen Sie den Anweisungen zur Auswahl und zum Hochladen der Indexdatei.

   Schrittweise Anleitungen finden Sie unter Wie lade ich Dateien und Ordner in einen Amazon S3-Bucket hoch? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
7. (Optional) Laden Sie andere Website-Inhalte in Ihren Bucket hoch.

Schritt 7: Bearbeiten der Block Public Access-Einstellungen

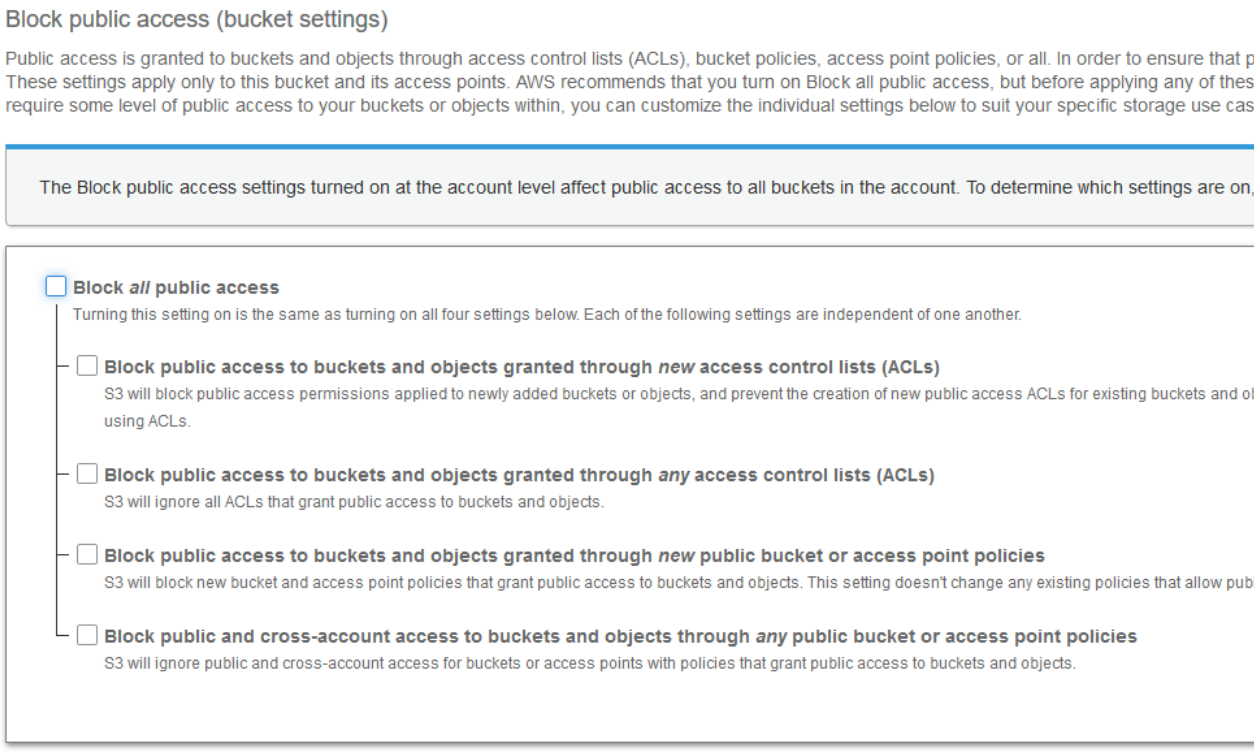
In diesem Beispiel bearbeiten Sie die Einstellungen für den öffentlichen Zugriff für den Domänen-Bucket (`example.com`), um den öffentlichen Zugriff zu ermöglichen.

Standardmäßig blockiert Amazon S3 den öffentlichen Zugriff auf Ihr Konto und Ihre Buckets. Wenn Sie einen Bucket verwenden möchten, um eine statische Website zu hosten, können Sie diese Schritte verwenden, um Ihre Einstellungen für Block Public Access zu bearbeiten:

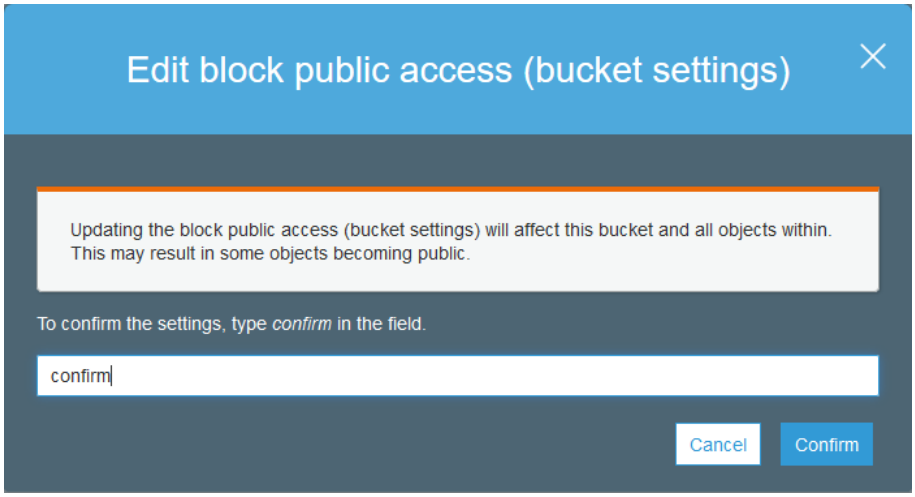
> Warning
>
> Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Verwenden von Amazon S3 Block Public, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie den Namen des Buckets aus, den Sie als statische Website konfiguriert haben.
3. Wählen Sie Permissions.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Löschen Sie Block all public access (Gesamten öffentlichen Zugriff blockieren), und wählen Sie Save (Speichern).

   > Warning
   >
   > Bevor Sie diesen Schritt ausführen, lesen Sie den Abschnitt Verwenden von Amazon S3Block Public, um sicherzustellen, dass Sie die mit dem Zulassen eines öffentlichen Zugriffs verbundenen Risiken kennen und akzeptieren. Wenn Sie die Einstellungen für Block Public Access deaktivieren, um Ihren Bucket öffentlich zu machen, kann jeder im Internet auf Ihren Bucket zugreifen. Wir empfehlen Ihnen, den gesamten öffentlichen Zugriff auf Ihre Buckets zu blockieren.



6. Geben Sie im Bestätigungsfeld **`confirm`** ein, und wählen Sie dann Confirm (Bestätigen).



Unter S3-Buckets wird der Access (Zugriff) auf Ihren Bucket auf Objects can be public (Objekte können öffentlich sein) aktualisiert. Sie können nun eine Bucket-Richtlinie hinzufügen, um die Objekte im Bucket öffentlich lesbar zu machen. Wenn der Access (Zugriff) immer noch als Bucket and objects not public (Bucket und Objekte nicht öffentlich) angezeigt wird, müssen Sie möglicherweise die Einstellungen für die Blockierung des öffentlichen Zugriff für Ihr Konto bearbeiten, bevor Sie eine Bucket-Richtlinie hinzufügen.

Schritt 8: Anfügen einer Bucket-Richtlinie

Nachdem Sie die Einstellungen für S3 Block Public Access bearbeitet haben, können Sie eine Bucket-Richtlinie hinzufügen, um öffentlichen Lesezugriff auf den Bucket zu gewähren. Wenn Sie öffentlichen Lesezugriff gewähren, kann jeder im Internet auf Ihren Bucket zugreifen.

> **Important**
>
> Die zuvor genannte Richtlinie ist nur ein Beispiel und erlaubt Vollzugriff auf die Inhalte Ihres Buckets. Bevor Sie mit diesem Schritt fortfahren, lesen Sie den Abschnitt Wie kann ich die Dateien in meinem Amazon S3-Bucket sichern? , um sicherzustellen, dass Sie die bewährten Methoden zum Sichern der Dateien in Ihrem S3-Bucket und die Risiken in Zusammenhang mit der Gewährung von öffentlichem Zugriff kennen.

1. Wählen Sie unter Buckets den Namen Ihres Buckets aus.
2. Wählen Sie Permissions.
3. Wählen Sie Bucket Policy aus.
4. Um öffentlichen Lesezugriff auf Ihre Website zu gewähren, kopieren Sie die folgende Bucket-Richtlinie und fügen Sie sie in den Bucket policy editor (Bucket-Richtlinieneditor) ein.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::example.com/*"
            ]
        }
    ]
}
```

5. Aktualisieren Sie den `Resource`, um Ihren Bucket-Namen anzugeben.

   In der vorangegangenen Beispiel-Bucket-Richtlinie ist *`example.com`* der Bucket-Name. Um diese Bucket-Richtlinie mit Ihrem eigenen Bucket zu verwenden, müssen Sie diesen Namen so aktualisieren, dass er mit Ihrem Bucket übereinstimmt.
6. Wählen Sie Save (Speichern) aus.

   Es wird eine Warnung angezeigt, die darauf hinweist, dass der Bucket öffentlich zugänglich ist. Unter Bucket Policy (Bucket-Richtlinie) wird die Bezeichnung Public (Öffentlich) angezeigt.

   Wenn die Fehlermeldung `Policy has invalid resource` angezeigt wird, bestätigen Sie, dass der Bucket-Name in der Bucket-Richtlinie mit Ihrem Bucket-Namen übereinstimmt. Informationen zum Hinzufügen einer Bucket-Richtlinie finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu?

   Wenn Sie die Warnung Error - Access denied (Fehler - Zugriff verweigert) erhalten und der Bucket policy editor (Bucket-Richtlinieneditor) das Speichern der Bucket-Richtlinie nicht zulässt, überprüfen Sie die Block Public Access-Einstellungen auf Kontoebene und Bucket-Ebene, um zu bestätigen, dass Sie öffentlichen Zugriff auf den Bucket zulassen. Weitere Informationen zu Website-Berechtigungen finden Sie unter Erforderliche Berechtigungen für den Website-Zugriff.

Im nächsten Schritt können Sie die Endpunkte Ihrer Website herausfinden und Ihren Domänen-Endpunkt testen.

Schritt 9: Testen Ihres Domänen-Endpunkts

Nachdem Sie den Stammdomänen-Bucket zum Hosten einer öffentlichen Website konfiguriert haben, können Sie Ihren Endpunkt testen. Weitere Informationen finden Sie unter Website-Endpunkte (p. 602). Sie können nur den Endpunkt für Ihren Domänen-Bucket testen, da Ihr Subdomänen-Bucket für die Website-Umleitung und nicht für das statische Website-Hosting eingerichtet ist.

> Note
>
> Amazon S3 unterstützt keinen HTTPS-Zugriff auf die Website. Wenn Sie HTTPS verwenden möchten, können Sie mit Amazon CloudFront eine statische Website bereitstellen, die auf Amazon S3 gehostet wird.
> Weitere Informationen finden Sie unter How do I use CloudFront to serve a static website hosted on Amazon S3? und Requiring HTTPS for Communication Between CloudFront and Your Amazon S3 Origin.

So testen Sie Ihren Website-Endpunkt

Wenn Sie beim Aktivieren des statischen Website-Hosting Ihren Website-Endpunkt notiert haben, geben Sie den Website-Endpunkt zum Testen Ihrer Website in Ihrem Browser ein. Wenn Ihr Browser Ihre `index.html`-Seite anzeigt, wurde die Website erfolgreich bereitgestellt. Weitere Informationen finden Sie unter Amazon S3-Website-Endpunkte.

Wenn Sie Ihren Website-Endpunkt vor dem Testen abrufen müssen, gehen Sie folgendermaßen vor:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie in der Liste Buckets den Namen des Buckets aus, den Sie zum Hosten einer statischen Website verwenden möchten.
3. Wählen Sie Properties (Eigenschaften).
4. Wählen Sie Static Website Hosting (Statisches Website-Hosting).
5. Um Ihren Website-Endpunkt zu testen, wählen Sie neben Endpoint (Endpunkt) Ihren Website-Endpunkt aus.

   Wenn Ihr Browser Ihre `index.html`-Seite anzeigt, wurde die Website erfolgreich bereitgestellt.

Im nächsten Schritt verwenden Sie Amazon Route 53, um den Kunden zu ermöglichen, Ihre Site über Ihre beiden benutzerdefinierten URLs aufzurufen.

Schritt 10: Hinzufügen von Aliasdatensätzen für Ihre Domäne und Subdomäne

In diesem Schritt erstellen Sie die Aliasdatensätze, die Sie der gehosteten Zone für Ihre Domänenzuordnungen `example.com` und `www.example.com` hinzufügen. Anstatt IP-Adressen zu verwenden, verwenden die Alias-Datensätze die Amazon S3-Website-Endpunkte. Amazon Route 53 verwaltet eine Zuweisung zwischen den Alias-Datensätzen und den IP-Adressen, wo sich die Amazon S3-Buckets befinden. Sie erstellen zwei Alias-Datensätze: einen für Ihre Stamm-Domäne und einen für Ihre Subdomäne.

So fügen Sie Ihrer Stamm-Domäne (**`example.com`**) einen Aliasdatensatz hinzu

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.

Note

Wenn Sie Route 53 nicht bereits verwenden, lesen Sie Schritt 1: Registrieren einer Domäne in der Entwicklerhandbuch für Amazon Route 53. Nachdem Sie Ihre Einrichtung abgeschlossen haben, können Sie mit den Anweisungen fortfahren.

2. Wählen Sie Hosted Zones (Gehostete Zonen) aus.
3. Wählen Sie in der Liste der gehosteten Zonen den Namen der gehosteten Zone aus, der dem Domänennamen entspricht.
4. Wählen Sie Create Record Set (Datensatz erstellen).
5. Geben Sie die folgenden Werte an:

   Name

   Akzeptieren Sie den Standardwert, bei dem es sich um den Namen Ihrer gehosteten Zone und Ihrer Domäne handelt.

   Für die Stamm-Domäne müssen Sie keine zusätzlichen Informationen in das Feld Name eingeben.

   Typ

   Wählen Sie A – IPv4 address (A – IPv4-Adresse) aus.

   Alias

   Wählen Sie Yes.

   Alias-Ziel

   Wählen Sie im Bereich S3 website endpoints (S3-Website-Endpunkte) der Liste Ihren Bucket-Namen aus.

   Der Bucket-Name sollte mit dem Namen übereinstimmen, der im Feld Name angezeigt wird. In der Liste Alias Target (Alias-Ziel) folgt dem Bucket-Namen beispielsweise der Amazon S3-Website-Endpunkt für die Region, in der der Bucket erstellt wurde, z. B. `example.com (s3-website-us-west-2)`. Alias Target (Alias-Ziel) listet einen Bucket auf, wenn:

   - Sie den Bucket als statische Website konfiguriert haben.
   - Der Name des Buckets mit dem Namen des Datensatzes übereinstimmt, den Sie anlegen.
   - Das aktuelle AWS-Konto den Bucket erstellt hat.

   Wenn Ihr Bucket nicht in der Liste Alias Target (Alias-Ziel) angezeigt wird, geben Sie den Amazon S3-Website-Endpunkt für die Region ein, in der der Bucket erstellt wurde, z. B. `s3-website-us-west-2`. Eine vollständige Liste der Amazon S3-Website-Endpunkte finden Sie unter Amazon S3-Website-Endpunkte. Weitere Informationen zum Aliasziel finden Sie unter AliasTarget im Entwicklerhandbuch für Amazon Route 53.

   Routing-Richtlinie

   Übernehmen Sie den Standardwert Simple.

   Evaluate Target Health

   Übernehmen Sie den Standardwert No.
6. Wählen Sie Create aus.

So fügen Sie Ihrer Subdomäne (**`www.example.com`**) einen Alias-Datensatz hinzu

1. Wählen Sie in der gehosteten Zone für Ihre Stamm-Domäne (`example.com`) die Option Create record set (Datensatz erstellen) aus.

2. Geben Sie die folgenden Werte an:

   Name

   Geben Sie als Subdomäne „www“ in das Feld ein.

   Typ

   Wählen Sie A – IPv4 address (A – IPv4-Adresse) aus.

   Alias

   Wählen Sie Yes.

   Alias-Ziel

   Wählen Sie im Bereich S3 website endpoints (S3-Website-Endpunkte) der Liste denselben Bucket-Namen aus, der im Feld Name angezeigt wird, z. B. `www.example.com (s3-website-us-west-2)`.

   Routing-Richtlinie

   Übernehmen Sie den Standardwert Simple.

   Evaluate Target Health

   Übernehmen Sie den Standardwert No.
3. Wählen Sie Create aus.

   Note

   Änderungen werden im Allgemeinen innerhalb von 60 Sekunden an alle Route 53-Server übertragen. Wenn die Übertragung abgeschlossen ist, können Sie den Datenverkehr an den Amazon S3-Bucket leiten, indem Sie den Namen des Alias-Datensatzes angeben, den Sie in diesem Verfahren erstellt haben.

Schritt 11: Testen der Website

Bestätigen Sie, dass die Website und die Umleitung korrekt funktionieren. Geben Sie Ihre URLs in Ihren Browser ein. In diesem Beispiel können Sie folgende URLs testen:

- Domäne (`http://example.com`) – Zeigt das Indexdokument im `example.com`-Bucket an.
- Subdomäne (`http://www.example.com`) – Leitet Ihre Anforderung an `http://example.com` weiter. Sie sehen das Indexdokument im `example.com`-Bucket.

In einigen Fällen müssen Sie möglicherweise den Cache Ihres Webbrowsers löschen, um das erwartete Verhalten zu sehen.

Sie können eine Amazon CloudFront-Verteilung einrichten, um die Leistung Ihrer Website zu verbessern und Protokolle bereitzustellen, mit denen Sie den Website-Datenverkehr überprüfen können. Weitere Informationen finden Sie unter Beschleunigung Ihrer Website mit Amazon CloudFront (p. 641).

Beschleunigung Ihrer Website mit Amazon CloudFront

Sie können Amazon CloudFront verwenden, um die Leistung Ihrer Website zu verbessern. CloudFront macht Ihrer Websitedateien (z. B. HTML, Bilder und Videos) aus Rechenzentren weltweit zugänglich (als Edge-Standorte bezeichnet). Wenn ein Besucher eine Datei von Ihrer Website anfordert, leitet CloudFront die Anforderung automatisch zu einer Kopie der Datei am nächstgelegenen Edge-Standort um. Das führt zu schnelleren Download-Zeiten, als wenn der Besucher den Inhalt von einem weitere entfernteren Rechenzentrum angefordert hätte.

CloudFront führt die Zwischenspeicherung von Inhalten an Edge-Standorten für einen von Ihnen festgelegten Zeitraum aus. Wenn die Besucher Inhalte anfordern, die über das Ablaufdatum hinaus zwischengespeichert wurden, prüft CloudFront auf dem Ursprungs-Server, ob eine neuere Version der Inhalte verfügbar ist. Wenn eine neuere Version verfügbar ist, kopiert CloudFront die neue Version auf den Edge-Standort. Änderungen an den ursprünglichen Inhalten werden zu den Edge-Standorten repliziert, indem die Besucher die Inhalte anfordern.

Themen

Schritt 1: Erstellen einer CloudFront-Verteilung

Erstellen Sie zuerst eine CloudFront-Verteilung. Damit steht Ihre Website in weltweit angesiedelten Rechenzentren zur Verfügung.

Erstellen einer Verteilung mit einem Amazon S3-Ursprung

1. Öffnen Sie die CloudFront-Konsole unter https://console.aws.amazon.com/cloudfront/.
2. Wählen Sie Create Distribution (Verteilung erstellen).
3. Wählen Sie auf der Seite Select a delivery method for your content (Bereitstellungsverfahren für Ihre Inhalte auswählen) in Web (Web) Get Started (Erste Schritte) aus.
4. Geben Sie auf der Seite Create Distribution (Verteilung erstellen) im Abschnitt Origin Settings (Ursprungseinstellungen) unter Origin Domain Name (Ursprungsdomänenname) den Amazon S3-Website-Endpunkt für Ihren Bucket ein, z. B. `example.com.s3-website.us-west-1.amazonaws.com`.

   CloudFront trägt den Wert von Origin ID (Ursprungs-ID) für Sie ein.
5. Behalten Sie für Default Cache Behavior Settings (Standardeinstellungen für das Zwischenspeicherverhalten) die Standardwerte bei.

   Weitere Informationen zu diesen Konfigurationsoptionen finden Sie unter Werte zur Angabe beim Erstellen oder Aktualisieren von Web-Verteilungen im Entwicklerhandbuch für Amazon CloudFront.
6. Gehen Sie unter Distribution Settings (Verteilungseinstellungen) wie folgt vor:

   a. Lassen Sie für die Option Price Class (Preisklasse) die Einstellung Use All Edge Locations (Best Performance) (Alle Edge-Standorte verwenden (Beste Leistung)) unverändert.

   b. Lassen Sie Alternate Domain Names (CNAMEs) (Alternative Domänennamen (CNAMEs)) leer.

   c. Geben Sie in Default Root Object (Standard-Root-Objekt den Namen Ihres Indexdokuments ein, z. B. `index.html`.

      Wenn die URL, die für den Zugriff auf die Verteilung verwendet wird, keinen Dateinamen enthält, gibt die CloudFront-Verteilung das Indexdokument zurück. Das Default Root Object (Standard-Root-Objekt) sollte genau mit dem Namen des Indexdokuments für Ihre statische Website übereinstimmen. Weitere Informationen finden Sie unter Konfigurieren eines Indexdokuments (p. 606).

   d. Wählen Sie für Logging (Protokollierung) die Option On (Ein).

   e. Wählen Sie als Bucket for Logs (Bucket für Protokolle) den Bucket zur Protokollierung aus, den Sie erstellt haben.

      Weitere Informationen zum Konfigurieren eines Protokoll-Buckets finden Sie unter (Optional) Protokollieren des Webdatenverkehrs (p. 611).

f. Wenn Sie die Protokolle speichern möchten, die durch den Datenverkehr zur CloudFront-Verteilung generiert werden, geben Sie in Log Prefix (Protokollpräfix) den Ordnernamen ein.

g. Behalten Sie für alle übrigen Einstellungen die Standardwerte bei.

7. Wählen Sie Create Distribution (Verteilung erstellen).
8. Um den aktuellen Status der Verteilung anzuzeigen, suchen Sie die Verteilung in der Konsole, und prüfen Sie die Spalte Status.

   Der Status `InProgress` gibt an, dass die Verteilung noch nicht vollständig bereitgestellt ist.

   Wenn die Verteilung bereitgestellt wurde, können Sie Ihren Inhalt mit dem neuen CloudFront-Domain-Namen referenzieren.
9. Notieren Sie den Wert für Domain Name (Domänenname), der in der CloudFront-Konsole angezeigt wird, z. B. `dj4p1rv6mvubz.cloudfront.net`.
10. Geben Sie den Domain-Namen der Verteilung in einen Web-Browser ein, um zu überprüfen, ob die CloudFront-Verteilung funktioniert.

    Wenn Ihre Website sichtbar ist, funktioniert die CloudFront-Verteilung. Wenn Ihre Website eine benutzerdefinierte Domäne hat, die bei Amazon Route 53 registriert ist, benötigen Sie den CloudFront-Domänennamen, um den Datensatz im nächsten Schritt zu aktualisieren.

Schritt 2: Aktualisieren der Datensätze für Ihre Domäne und Unterdomäne

Sie haben jetzt erfolgreich eine CloudFront-Verteilung erstellt. Als Nächstes müssen Sie den Aliasdatensatz in Route 53 so ändern, dass er auf die neue CloudFront-Verteilung zeigt.

So aktualisieren Sie den Aliasdatensatz so, dass er auf eine CloudFront-Verteilung verweist

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie auf der Seite Hosted Zones (Gehostete Zonen) die gehostete Zone aus, die Sie für Ihre Unterdomäne erstellt haben.
3. Wählen Sie Go to Record Sets (Zu den Datensätzen).
4. Wählen Sie den A-Datensatz aus, den Sie für Ihre Unterdomäne erstellt haben, z. B. `www.example.com`.
5. Wählen Sie unter Alias Target (Alias-Ziel) die CloudFront-Verteilung aus.
6. Wählen Sie Save Record Set (Datensatz speichern).
7. Wiederholen Sie dieses Verfahren, um den A-Datensatz für die Stamm-Domain zur CloudFront-Verteilung umzuleiten.

   Die Aktualisierung der Datensätze wird innerhalb von 2 bis 48 Stunden wirksam.
8. Um zu sehen, ob die neuen A-Datensätze wirksam sind, geben Sie in einem Webbrowser die URL Ihrer Unterdomäne ein, z. B. `http://www.example.com`.

   Wenn der Browser Sie nicht mehr zur Stammdomäne umleitet (z. B. `http://example.com`), sind die neuen A-Datensätze vorhanden. Wenn der neue A-Datensatz wirksam ist, wird der Datenverkehr, der von diesem an die CloudFront-Verteilung weitergeleitet wird, nicht zur Stamm-Domain umgeleitet. Besucher, die die Website über `http://www.example.com` oder CloudFront referenzieren, werden zum nächsten `http://example.com`-Edge-Standort umgeleitet, um schnellere Download-Zeiten zu erzielen.

   Tip

   Umleitungseinstellungen können von Browsern zwischengespeichert werden. Wenn Sie annehmen, dass die neuen A-Datensatzeinstellungen wirksam sind, aber trotzdem eine Umleitung von `http://www.example.com` nach `http://example.com` feststellen, löschen Sie zum Testen den Verlauf und den Cache des Browsers. Schließen Sie den

Browser und öffnen Sie ihn wieder oder verwenden Sie einen anderen Webbrowser, wenn Sie einen weiteren installiert haben.

(Optional) Schritt 3: Überprüfen der Protokolldateien

Die Zugriffsprotokolle teilen Ihnen mit, wie viele Menschen die Website besuchen. Sie enthalten auch wertvolle Geschäftsdaten, die Sie mithilfe anderer Services wie beispielsweise Amazon EMR analysieren können.

CloudFront-Protokolle werden in dem Bucket und Ordner gespeichert, den Sie beim Erstellen der CloudFront-Verteilung und beim Aktivieren der Protokollierung auswählen. CloudFront schreibt Protokolle innerhalb von 24 Stunden ab dem Zeitpunkt, an dem die entsprechenden Anforderungen gestellt werden, in Ihren Protokoll-Bucket.

Die Protokolldateien für Ihre Website anzeigen

1. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.
2. Wählen Sie den Protokoll-Bucket für Ihre Website aus.
3. Wählen Sie den Ordner „CloudFront-Protokolle“ aus.
4. Laden Sie die `.gzip`-Dateien herunter, die von CloudFront geschrieben wurden, bevor Sie diese öffnen.

   Wenn Sie die Website nur zur Übung erstellt haben, können Sie die von Ihnen zugewiesenen Ressourcen löschen, damit keine weiteren Kosten für Sie anfallen. Lesen Sie dazu den Abschnitt Bereinigung Ihrer Beispielressourcen (p. 644). Nachdem Sie die AWS-Ressourcen gelöscht haben, ist die Website nicht mehr verfügbar.

Bereinigung Ihrer Beispielressourcen

Wenn Sie die statische Website zur Übung erstellt haben, sollten Sie die zugewiesenen AWS-Ressourcen löschen, damit keine weiteren Kosten für Sie anfallen. Nach der Löschung der AWS-Ressourcen ist die Website nicht mehr verfügbar.

Aufgaben

Schritt 1: Löschen der Amazon CloudFront-Verteilung

Bevor Sie eine Amazon CloudFront-Verteilung löschen, müssen Sie sie deaktivieren. Eine deaktivierte Verteilung funktioniert nicht mehr und es fallen keine weiteren Kosten für sie an. Sie können eine deaktivierte Verteilung jederzeit wieder aktivieren. Nachdem Sie eine deaktivierte Verteilung gelöscht haben, ist sie nicht länger verfügbar.

Eine CloudFront-Verteilung deaktivieren und löschen

1. Öffnen Sie die CloudFront-Konsole unter https://console.aws.amazon.com/cloudfront/.
2. Klicken Sie mit der rechten Maustaste auf die Verteilung, die Sie deaktivieren möchten, und anschließend auf Disable (Deaktivieren).
3. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Yes, Disable (Ja, deaktivieren).
4. Wählen Sie die deaktivierte Verteilung aus, und klicken Sie dann auf Delete (Löschen).
5. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Yes, Delete.

Schritt 2: Löschen der gehosteten Route 53-Zone

Bevor Sie eine gehostete Zone löschen können, müssen Sie die zuvor erstellten Datensätze löschen. SOA-Einträge (Start of Authority, Autoritätsursprung) und NS-Einträge (Nameserver, Namenserver) müssen nicht von Ihnen entfernt werden, da dies automatisch beim Löschen der gehosteten Zone ausgeführt wird.

Sie löschen die Datensätze wie folgt:

1. Öffnen Sie die Route 53-Konsole unter https://console.aws.amazon.com/route53/.
2. Wählen Sie in der Liste der Domänennamen Ihren Domänennamenaus, und wählen Sie anschließend Go to Record Sets (Zu den Datensätzen).
3. Wählen Sie in der Liste der Datensatzsätze die A-Datensätze aus, die Sie erstellt haben.

   Der einzelne Datensatztyp wird in der Spalte Type (Typ) aufgeführt.
4. Wählen Sie Delete Record Set (Datensatz löschen).
5. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Confirm (Bestätigen).

So löschen Sie eine gehostete Route 53-Zone

1. Setzen Sie das vorherige Verfahren fort, indem Sie Back to Hosted Zones (Zurück zu gehosteten Zonen) auswählen.
2. Wählen Sie Ihren Domänennamen und dann Delete Hosted Zone (Gehostete Zone löschen).
3. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Confirm (Bestätigen).

Schritt 3: Deaktivieren der Protokollierung und Löschen Ihres S3-Buckets

Bevor Sie Ihren S3-Bucket löschen, stellen Sie sicher, dass die Protokollierung für den Bucket deaktiviert ist. Andernfalls schreibt AWS weiter Protokolle in Ihren Bucket, während Sie ihn löschen.

Deaktivieren Sie die Protokollierung für einen Bucket wie folgt:

1. Öffnen Sie die Amazon S3-Konsole unter der Adresse https://console.aws.amazon.com/s3/.
2. Wählen Sie unter Buckets (Buckets) den Namen Ihres Buckets aus. Wählen Sie anschließend Properties (Eigenschaften) aus.
3. Wählen Sie unter Properties (Eigenschaften) Logging (Protokollierung) aus.
4. Deaktivieren Sie das Kontrollkästchen Enabled (Aktiviert).
5. Wählen Sie Save aus.

Jetzt können Sie Ihren Bucket löschen. Weitere Informationen finden Sie unter Wie lösche ich einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Konfigurieren von Amazon S3-Ereignisbenachrichtigungen

Die Amazon S3-Benachrichtigungsfunktion ermöglicht Ihnen, Benachrichtigungen zu erhalten, wenn innerhalb Ihres Buckets bestimmte Ereignisse stattfinden. Um Benachrichtigungen zu aktivieren, müssen Sie zuerst eine Benachrichtigungskonfiguration hinzufügen, die die Ereignisse angibt, die Amazon S3 veröffentlichen soll, und die Ziele, an die Amazon S3 die Benachrichtigungen senden soll. Sie speichern dieses Konfiguration in der einem Bucket zugeordneten notification-Subressource. (Weitere Informationen finden Sie unter Optionen für die Bucket-Konfiguration (p. 61).) Amazon S3 stellt eine API zur Verwaltung dieser Subressource bereit.

> Important
>
> Amazon S3-Ereignisbenachrichtigungen sind so konzipiert, dass sie mindestens einmal zugestellt werden. Ereignisbenachrichtigungen stellen Ereignisse typischerweise in wenigen Sekunden bereit, manchmal kann dies aber auch eine Minute oder länger dauern.
> Wenn zwei Schreibvorgänge an einem einzelnen nicht versionsgesteuerten Objekt gleichzeitig vorgenommen werden, ist es möglich, dass nur eine einzelne Ereignisbenachrichtigung gesendet wird. Wenn Sie sicherstellen möchten, dass für jeden erfolgreichen Schreibvorgang eine Ereignisbenachrichtigung gesendet wird, können Sie die Versioning auf Ihrem Bucket aktivieren. Bei der Versioning erstellt jeder erfolgreiche Schreibvorgang eine neue Version Ihres Objekts und sendet auch eine Ereignisbenachrichtigung.

Themen

Übersicht über Benachrichtigungen

Zurzeit kann Amazon S3 Benachrichtigungen für die folgenden Ereignisse veröffentlichen:

- Neues Objekt erstellt — Amazon S3 unterstützt mehrere APIs für das Erstellen von Objekten. Sie können eine Benachrichtigung anfordern, die nur dann erstellt wird, wenn eine bestimmte API verwendet wird (z. B. `s3:ObjectCreated:Put`), oder einen Platzhalter verwenden (z. B. `s3:ObjectCreated:*`), um eine Benachrichtigung anzufordern, wenn ein Objekt erstellt wird, unabhängig von der verwendeten API.
- Entfernung eines Objekts— Amazon S3 unterstützt das Löschen von Objekten mit und ohne Versioning. Weitere Informationen zum Objekt-Versioning finden Sie unter Objekt-Versioning (p. 127) und Verwenden von Versioning (p. 514).

  Sie können eine Benachrichtigung anfordern, wenn ein Objekt gelöscht wird, oder wenn ein versionsfähiges Objekt permanent gelöscht wird. Dazu wird der Ereignistyp `s3:ObjectRemoved:Delete` verwendet. Sie können auch eine Benachrichtigung anfordern, wenn eine Löschmarkierung für ein versionsfähiges Objekt erstellt wird. Dazu wird der Ereignistyp

`s3:ObjectRemoved:DeleteMarkerCreated` verwendet. Sie können auch eine Wildcard `s3:ObjectRemoved:*` verwenden, um bei jedem Löschen eines Objekts eine Benachrichtigung anzufordern. Weitere Informationen zum Löschen von versionsfähigen Objekten finden Sie unter Löschen von Objektversionen (p. 527).

- Wiederherstellung von Objekten – Amazon S3 unterstützt die Wiederherstellung von in der Speicherklasse S3 Glacier archivierten Objekten. Sie können mittels `s3:ObjectRestore:Completed` Benachrichtigungen bei Abschluss von Objektwiederherstellungen anfordern. Sie können mittels `s3:ObjectRestore:Post` Benachrichtungen bei Initiierung einer Wiederherstellung anfordern.
- Verlust eines Reduced Redundancy Storage (RRS)-Objekts — Amazon S3 sendet eine Benachrichtigung, wenn der Verlust eines Objekts in der Speicherklasse RRS entdeckt wird.
- Replikationsereignisse — Amazon S3 sendet Ereignisbenachrichtigungen für Replikationskonfigurationen, bei denen S3 Replication Time Control (S3 RTC) aktiviert ist. Sie sendet diese Benachrichtigungen, wenn für ein Objekt die Replikation fehlschlägt, wenn ein Objekt den Schwellenwert von 15 Minuten überschreitet, wenn ein Objekt nach dem Schwellenwert von 15 Minuten repliziert wird und wenn ein Objekt nicht mehr durch Replikationsmetriken verfolgt wird. Es veröffentlicht ein zweites Ereignis, wenn dieses Objekt in die Zielregion repliziert.

Eine Liste der unterstützten Ereignistypen finden Sie unter Unterstützte Ereignistypen (p. 649).

Amazon S3 unterstützt die folgenden Ziele, wo es Ereignisse veröffentlichen kann:

- Amazon Simple Notification Service-(Amazon SNS)-Thema

  Amazon SNS ist ein flexibler, vollständig verwalteter Push-Messaging-Service. Mit diesem Service können Sie Mitteilungen an Mobilgeräte oder verteilte Dienste übertragen. Mit SNS können Sie eine Nachricht einmal veröffentlichen und ein- oder mehrmals übermitteln. Weitere Informationen zu SNS finden Sie auf der Produktdetailseite für Amazon SNS.
- Amazon Simple Queue Service-(Amazon SQS)-Warteschlange

  Amazon SQS ist ein skalierbarer und vollständig verwalteter Warteschlangenservice. Mit SQS können Sie beliebige Datenvolumen übertragen, ohne dass andere Services stets verfügbar sein müssen. In Ihrer Benachrichtigungskonfiguration können Sie anfordern, dass Amazon S3 Ereignisse in einer SQS-Warteschlange veröffentlichen kann.

  Derzeit ist nur die Standard-SQS-Warteschlange als Amazon S3-Ereignisbenachrichtigungsziel zugelassen, während die FIFO-SQS-Warteschlange nicht zugelassen ist. Weitere Informationen zu Amazon SQS finden Sie auf der Amazon SQS-Produktdetailseite.
- AWS Lambda

  AWS Lambda ist ein Datenverarbeitungsservice, mit dem Sie auf einfache Weise Anwendungen erstellen können, die schnell auf neue Informationen reagieren. AWS Lambda führt Ihren Code als Reaktion auf bestimmte Ereignisse aus, beispielsweise das Hochladen von Bildern, Aktivitäten in einer App, Websiteklicks oder Ausgaben aus verbundenen Geräten.

  Sie können AWS Lambda verwenden, um weitere AWS-Services mit benutzerdefinierter Logik bereitzustellen oder ein eigenes Backend zu erstellen, das mit der Größe, Leistung und Sicherheit von AWS arbeitet. Mit AWS Lambda können Sie ganz einfach unabhängige, ereignisabhängige Anwendungen erstellen, die nur ausgeführt werden, wenn sie benötigt werden, und die automatisch von einigen wenigen Anfragen pro Tag bis hin zu Tausenden von Anfragen pro Sekunde skaliert werden können.

  AWS Lambda als Reaktion auf Amazon S3-Bucket-Ereignisse benutzerdefinierten Code ausführen. Sie laden Ihren benutzerdefinierten Code zu AWS Lambda hoch und erstellen eine so genannte Lambda-Funktion. Wenn Amazon S3 ein Ereignis eines bestimmten Typs entdeckt (z. B. eine Objekterstellung), kann das Ereignis zu AWS Lambda veröffentlicht werden und Ihre Funktion in Lambda aufrufen. In Reaktion darauf führt AWS Lambda Ihre Funktion aus.

Warning

Wenn Ihre Benachrichtigung schließlich in den Bucket schreibt, der die Benachrichtigung auslöst, kann dies zu einer Ausführungsschleife führen. Wenn der Bucket z. B. eine Lambda-Funktion immer dann auslöst, wenn ein Objekt hochgeladen wird, und die Funktion ein Objekt in den Bucket hochlädt, löst sich die Funktion indirekt selbst aus. Um dies zu vermeiden, verwenden Sie zwei Buckets oder konfigurieren Sie den Auslöser so, dass er nur für einen Präfix gilt, der für eingehende Objekte verwendet wird.
Weitere Informationen und ein Beispiel für die Verwendung von Amazon S3-Benachrichtigungen mit AWS Lambda finden Sie unter Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda Developer Guide.

Vorgehensweise zum Aktivieren von Ereignisbenachrichtigungen

Die Aktivierung von Benachrichtigungen erfolgt auf Bucket-Ebene. Sie speichern die Benachrichtigungskonfiguration in der einem Bucket zugeordneten notification-Subressource. Sie können jede der folgenden Methoden verwenden, um die Benachrichtigungskonfiguration zu verwalten:

- Verwenden der Amazon S3-Konsole

  Die Benutzeroberfläche der Konsole ermöglicht Ihnen, eine Benachrichtigungskonfiguration für einen Bucket einzurichten, ohne Code schreiben zu müssen. Weitere Informationen finden Sie unter Wie kann ich Ereignisbenachrichtigungen für einen S3-Bucket aktivieren und konfigurieren? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.
- Programmgesteuerte Verwendung der AWS-SDKs

  Note

  Falls nötig, können Sie auch die REST API-Aufrufe in Amazon S3 direkt von Ihrem Code aus durchführen. Allerdings kann dies aufwändig sein, weil Sie Code zur Authentifizierung Ihrer Anfragen schreiben müssen.

  Intern rufen sowohl die Konsole als auch die SDKs die Amazon S3-REST-API auf, um die mit dem Bucket verknüpften notification-Unterressourcen zu verwalten. Informationen zur Benachrichtigungskonfiguration anhand von AWS SDK-Beispielen finden Sie unter Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema und SQS-Warteschlange) (p. 659).

  Unabhängig von der verwendeten Methode speichert Amazon S3 die Benachrichtigungskonfiguration als XML in der mit dem jeweiligen Bucket verknüpften Unterressource notification. Weitere Informationen zu Bucket-Subressourcen finden Sie unter Optionen für die Bucket-Konfiguration (p. 61).

  Standardmäßig sind für keinen Ereignistyp Benachrichtigungen aktiviert. Aus diesem Grund speichert die notification-Subressource anfänglich eine leere Konfiguration.

```
<NotificationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</NotificationConfiguration>
```

  Um die Benachrichtigungen für bestimmte Ereignistypen zu aktivieren, ersetzen Sie das XML durch die entsprechende Konfiguration, die die Ereignistypen identifiziert, die Amazon S3 veröffentlichen soll, sowie das Ziel, in dem die Ereignisse veröffentlicht werden sollen. Sie müssen für jedes Ziel eine entsprechende XML-Konfiguration hinzufügen. Zum Beispiel:

  - Veröffentlichung von Ereignismitteilungen in einer SQS-Warteschlange — Um eine SQS-Warteschlange als Benachrichtigungsziel für ein oder mehrere Ereignistypen einzurichten, fügen Sie die `QueueConfiguration` hinzu.

```
<NotificationConfiguration>
  <QueueConfiguration>
    <Id>optional-id-string</Id>
    <Queue>sqs-queue-arn</Queue>
    <Event>event-type</Event>
    <Event>event-type</Event>
     ...
  </QueueConfiguration>
   ...
</NotificationConfiguration>
```

- Veröffentlichung von Ereignismitteilungen in einem SNS-Thema — Um ein SNS-Thema als Benachrichtigungsziel für spezifische Ereignistypen einzurichten, fügen Sie die `TopicConfiguration` hinzu.

```
<NotificationConfiguration>
  <TopicConfiguration>
     <Id>optional-id-string</Id>
     <Topic>sns-topic-arn</Topic>
     <Event>event-type</Event>
     <Event>event-type</Event>
      ...
  </TopicConfiguration>
   ...
</NotificationConfiguration>
```

- Rufen Sie die AWS Lambda-Funktion auf und übergeben Sie ihr eine Ereignismitteilung als Argument — Um eine Lambda-Funktion als Benachrichtigungsziel für spezifische Ereignistypen einzurichten, fügen Sie die `CloudFunctionConfiguration` hinzu.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
     <Id>optional-id-string</Id>
     <Cloudcode>cloud-function-arn</Cloudcode>
     <Event>event-type</Event>
     <Event>event-type</Event>
      ...
  </CloudFunctionConfiguration>
   ...
</NotificationConfiguration>
```

Um alle für einen Bucket konfigurierten Benachrichtigungen zu entfernen, speichern Sie ein leeres `<NotificationConfiguration/>`-Element in der notification-Unterressource.

Wenn Amazon S3 ein Ereignis des spezifischen Typs erkennt, veröffentlicht es eine Meldung mit der Ereignisinformation. Weitere Informationen finden Sie unter Struktur von Ereignismeldungen (p. 665).

Ereignisbenachrichtigungstypen und -ziele

Dieser Abschnitt beschreibt die Ereignisbenachrichtigungstypen, die von Amazon S3 unterstützt werden, sowie den Typ der Ziele, wo die Benachrichtigungen veröffentlicht werden können.

Unterstützte Ereignistypen

Amazon S3 kann Ereignisse der folgenden Typen veröffentlichen. Sie spezifizieren diese Ereignistypen in der Benachrichtigungskonfiguration.

Ereignistypen	Beschreibung
s3:ObjectCreated:* s3:ObjectCreated:Put s3:ObjectCreated:Post s3:ObjectCreated:Copy s3:ObjectCreated:CompleteMultipartUpload	Amazon S3 APIs wie PUT, POST und COPY können ein Objekt erstellen. Unter Verwendung dieser Ereignistypen können Sie eine Benachrichtigung aktivieren, wenn ein Objekt mit einer spezifischen API erstellt wird. Sie können auch den Ereignistyp s3:ObjectCreated:* verwenden, um eine Benachrichtigung unabhängig von der API anzufordern, die zum Erstellen eines Objekts verwendet wurde. Für fehlgeschlagene Operationen erhalten Sie keine Ereignisbenachrichtigungen.
s3:ObjectRemoved:* s3:ObjectRemoved:Delete s3:ObjectRemoved:DeleteMarkerCreated	Durch Verwendung der ObjectRemoved-Ereignistypen können Sie eine Benachrichtigung aktivieren, wenn ein Objekt oder mehrere Objekte aus einem Bucket entfernt werden. Sie können eine Benachrichtigung anfordern, wenn ein Objekt gelöscht wird oder ein Objekt mit Versioning permanent gelöscht wird, indem Sie den Ereignistyp s3:ObjectRemoved:Delete verwenden. Sie können auch eine Benachrichtigung anfordern, wenn eine Löschmarkierung für ein Objekt mit Versioning erstellt wird, indem Sie den Ereignistyp s3:ObjectRemoved:DeleteMarkerCreated verwenden. Weitere Informationen zum Löschen von versionsfähigen Objekten finden Sie unter Löschen von Objektversionen (p. 527). Sie können auch eine Wildcard `s3:ObjectRemoved:*` verwenden, um bei jedem Löschen eines Objekts eine Benachrichtigung anzufordern. Für ein automatisches Löschen aufgrund von Lebenszyklusrichtlinien und für fehlgeschlagene Operationen erhalten Sie keine Ereignisbenachrichtigungen.
s3:ObjectRestore:Post s3:ObjectRestore:Completed	Mithilfe von Objektwiederherstellungs-Ereignistypen können Sie Benachrichtigungen für Initiierungs- und Abschlussereignisse erhalten, wenn Objekte aus der Speicherklasse S3 Glacier wiederhergestellt werden. Sie verwenden `s3:ObjectRestore:Post`, um Benachrichtigungen über die Initiierung von Objektwiederherstellungen zu erhalten. Sie verwenden `s3:ObjectRestore:Completed`, um Benachrichtigungen über den Abschluss von Objektwiederherstellungen zu erhalten.
s3:ReducedRedundancyLostObject	Sie können diesen Ereignistyp verwenden, um Amazon S3 zum Senden einer Benachrichtigung aufzufordern, wenn Amazon S3 den Verlust eines Objekts der RRS-Speicherklasse entdeckt.
s3:Replication:OperationFailedReplication	Sie erhalten dieses Benachrichtigungsereignis, wenn ein Objekt, das für die Replikation mit Amazon S3 Replication Time Control berechtigt war, nicht repliziert werden konnte.
s3:Replication:OperationMissedThreshold	Sie erhalten dieses Benachrichtigungsereignis für ein Objekt, bei dem eine Replikation mithilfe der Funktion Amazon S3 Replication Time Control berechtigt war und bei dem der 15-minütigen Schwellenwert für die Replikation überschritten wurde.

Ereignistypen	Beschreibung
s3:Replication:OperationReplicatedAfterThreshold	Sie erhalten dieses Benachrichtigungsereignis für ein Objekt, bei dem eine Replikation mithilfe der Funktion Amazon S3 Replication Time Control berechtigt war und nach dem 15-minütigen Schwellenwert repliziert wurde.
s3:Replikation:OperationNotTracked	Sie erhalten dieses Benachrichtigungsereignis für ein Objekt, das für die Replikation mit der Funktion Amazon S3 Replication Time Control berechtigt war, aber nicht mehr von Replikationsmetriken verfolgt wird.

Unterstützte Ziele

Amazon S3 kann Ereignisbenachrichtigungsmeldungen an die folgenden Ziele senden. Sie spezifizieren den ARN-Wert dieser Ziele in der Benachrichtigungskonfiguration.

- Veröffentlichung von Ereignismitteilungen zu einem Amazon Simple Notification Service (Amazon SNS)-Thema
- Veröffentlichung von Ereignismitteilungen zu einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange

  Note

  Wenn die Zielwarteschlange oder das Zielthema SSE verwendet, benötigt Amazon S3 Zugriff auf den zugehörigen AWS Key Management Service (AWS KMS) Kundenhauptschlüssel (CMK), um die Nachrichtenverschlüsselung zu aktivieren.
- Veröffentlichung von Ereignismitteilungen zu AWS Lambda durch Aufruf einer Lambda-Funktion und Übergabe der Ereignismitteilung als Argument

Sie müssen Amazon S3 Berechtigungen zum Veröffentlichen von Nachrichten an ein Amazon SNS-Thema oder eine Amazon SQS-Warteschlange gewähren. Außerdem müssen Sie Amazon S3 die Berechtigung zum Aufruf einer AWS Lambda in Ihrem Namen gewähren. Informationen zum erteilen dieser Berechtigungen finden Sie unter Erteilen von Berechtigungen zur Veröffentlichung von Ereignisbenachrichtigungsmeldungen an einem Ziel (p. 656).

Konfigurieren von Benachrichtigungen mit Objektschlüsselnamenfilterung

Sie können Benachrichtigungen konfigurieren, die nach dem Präfix und dem Suffix des Objektschlüsselnamens gefiltert werden. Sie können beispielsweise eine Konfiguration so einrichten, dass Sie nur eine Benachrichtigung erhalten, wenn einem Bucket Bilddateien mit der Erweiterung „`.jpg`“ hinzugefügt werden. Sie können auch eine Konfiguration verwenden, die eine Benachrichtigung an ein Amazon SNS-Thema sendet, wenn dem Bucket ein Objekt mit dem Präfix „`images/`“ hinzugefügt wird, während Benachrichtigungen für Objekte mit dem Präfix „`logs/`“ im selben Bucket an eine AWS Lambda-Funktion gesendet werden.

Sie können Benachrichtigungskonfigurationen, die nach Objektschlüsselnamen filtern, über die Amazon S3-Konsole und unter direkter Verwendung von Amazon S3-APIs über die AWS-SDKs oder die REST-APIs einrichten. Informationen zur Verwendung der Konsolenoberfläche für die Einrichtung einer Benachrichtigungskonfiguration für einen Bucket finden Sie unter Wie kann ich Ereignisbenachrichtigungen

für einen S3-Bucket aktivieren und konfigurieren? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Amazon S3 speichert die Benachrichtigungskonfiguration als XML in der einem Bucket zugeordneten notification-Unterressource wie in Vorgehensweise zum Aktivieren von Ereignisbenachrichtigungen (p. 648) beschrieben. Sie können die `Filter` XML-Struktur verwenden, um Regeln für die nach dem Präfix und/oder dem Suffix eines Objektschlüsselnamens gefilterten Benachrichtigungen zu definieren. Informationen zu den Details der XML-Struktur von `Filter` finden Sie unter PUT Bucket-Benachrichtigung im Amazon Simple Storage Service API Reference.

Benachrichtigungskonfigurationen, die `Filter` verwenden, können keine Filterregeln mit überlappenden Präfixen, überlappenden Suffixen oder Präfix- und Suffix-Überlappung definieren. Die folgenden Abschnitte enthalten Beispiele für gültige Benachrichtigungskonfigurationen mit Objektschlüssel-Namensfilterung. Sie enthalten auch Beispiele für Benachrichtigungskonfigurationen, die wegen der Überlappung von Präfix/Suffix ungültig sind.

Beispiele für gültige Benachrichtigungskonfigurationen mit Filterung nach dem Objektschlüsselnamen

Die folgende Benachrichtigungskonfiguration enthält eine Warteschlangekonfiguration, die eine Amazon SQS-Warteschlange für Amazon S3 identifiziert, um Ereignisse des Typs `s3:ObjectCreated:Put` zu veröffentlichen. Die Ereignisse werden veröffentlicht, wenn ein Objekt mit dem Präfix `images/` und dem Suffix `jpg` mit PUT in einen Bucket geschrieben wird.

```
<NotificationConfiguration>
  <QueueConfiguration>
      <Id>1</Id>
      <Filter>
          <S3Key>
              <FilterRule>
                  <Name>prefix</Name>
                  <Value>images/</Value>
              </FilterRule>
              <FilterRule>
                  <Name>suffix</Name>
                  <Value>jpg</Value>
              </FilterRule>
          </S3Key>
     </Filter>
     <Queue>arn:aws:sqs:us-west-2:444455556666:s3notificationqueue</Queue>
     <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
</NotificationConfiguration>
```

Die folgende Benachrichtigungskonfigurationen hat mehrere nicht überlappende Präfixe. Die Konfiguration definiert, dass Benachrichtigungen für PUT-Anforderungen im Ordner `images/` in queue-A geschrieben werden, während Benachrichtigungen für PUT-Anforderungen im Ordner `logs/` in queue-B geschrieben werden.

```
<NotificationConfiguration>
  <QueueConfiguration>
     <Id>1</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images/</Value>
                </FilterRule>
            </S3Key>
     </Filter>
```

```
    <Queue>arn:aws:sqs:us-west-2:444455556666:sqs-queue-A</Queue>
    <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
  <QueueConfiguration>
    <Id>2</Id>
    <Filter>
           <S3Key>
               <FilterRule>
                   <Name>prefix</Name>
                   <Value>logs/</Value>
               </FilterRule>
           </S3Key>
    </Filter>
    <Queue>arn:aws:sqs:us-west-2:444455556666:sqs-queue-B</Queue>
    <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
</NotificationConfiguration>
```

Die folgende Benachrichtigungskonfigurationen hat mehrere nicht überlappende Suffixe. Die Konfiguration definiert, dass alle dem Bucket neu hinzugefügten `.jpg`-Bilder über Lambda cloud-function-A und alle neu hinzugefügten `.png`-Bilder über cloud-function-B verarbeitet werden. Die Suffixe `.png` und `.jpg` überschneiden sich nicht, obwohl ihr letzter Buchstabe identisch ist. Zwei Suffixe werden als überlappend betrachtet, wenn eine Zeichenfolge mit beiden Suffixen enden kann. Da eine Zeichenfolge nicht mit `.png` und `.jpg` gleichzeitig enden kann, sind die Suffixe in der Beispielkonfiguration keine überlappenden Suffixe.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
    <Id>1</Id>
    <Filter>
           <S3Key>
               <FilterRule>
                   <Name>suffix</Name>
                   <Value>.jpg</Value>
               </FilterRule>
           </S3Key>
    </Filter>
    <Cloudcode>arn:aws:lambda:us-west-2:444455556666:cloud-function-A</Cloudcode>
    <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
  <CloudFunctionConfiguration>
    <Id>2</Id>
    <Filter>
           <S3Key>
               <FilterRule>
                   <Name>suffix</Name>
                   <Value>.png</Value>
               </FilterRule>
           </S3Key>
    </Filter>
    <Cloudcode>arn:aws:lambda:us-west-2:444455556666:cloud-function-B</Cloudcode>
    <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
</NotificationConfiguration>
```

Ihre Benachrichtigungskonfigurationen, die `Filter` verwenden, können keine Filterregeln mit überlappenden Präfixen für dieselben Ereignistypen definieren, es sei denn, die überlappenden Präfixe werden mit nicht überlappenden Suffixen verwendet. Die folgende Beispielkonfiguration zeigt, wie Objekte, die mit einem gemeinsamen Präfix, aber nicht überlappenden Suffixen erstellt werden, an unterschiedliche Ziele geliefert werden können.

```
<NotificationConfiguration>
```

```
  <CloudFunctionConfiguration>
     <Id>1</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.jpg</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <Cloudcode>arn:aws:lambda:us-west-2:444455556666:cloud-function-A</Cloudcode>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
  <CloudFunctionConfiguration>
     <Id>2</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.png</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <Cloudcode>arn:aws:lambda:us-west-2:444455556666:cloud-function-B</Cloudcode>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
</NotificationConfiguration>
```

Beispiele für Benachrichtigungskonfigurationen mit ungültiger Präfix/Suffix-Überlappung

Ihre Benachrichtigungskonfigurationen, die `Filter` verwenden, können größtenteils keine Filterregeln mit überlappenden Präfixen, überlappenden Suffixen oder überlappenden Präfix- und Suffix-Kombinationen für dieselben Ereignistypen definieren. (Überlappende Präfixe sind möglich, solange sich die Suffixe nicht überlappen. Ein Beispiel finden Sie unter Konfigurieren von Benachrichtigungen mit Objektschlüsselnamenfilterung (p. 651).)

Sie können überlappende Objektschlüsselnamensfilter mit unterschiedlichen Ereignistypen verwenden. Beispielsweise könnten sie eine Benachrichtigungskonfiguration erstellen, die das Präfix `image/` für den Ereignistyp `ObjectCreated:Put` und das Präfix `image/` für den Ereignistyp `ObjectDeleted:*` verwendet.

Sie erhalten einen Fehler, wenn Sie versuchen, eine Benachrichtigungskonfiguration mit ungültigen, sich überschneidenden Namensfiltern für dieselben Ereignistypen zu speichern, wenn Sie die Amazon S3-Konsole oder -API verwenden. Dieser Abschnitt zeigt Beispiele für Benachrichtigungskonfigurationen, die aufgrund überlappender Namensfilter ungültig sind.

Es wird angenommen, dass eine vorhandene Benachrichtigungskonfigurationsregel ein Standardpräfix und -suffix hat, die mit allen anderen Präfixen bzw. Suffixen übereinstimmen. Die folgende Benachrichtigungskonfiguration ist ungültig, da sie überlappende Präfixe hat, wobei das Root-Präfix sich mit allen anderen Präfixen überlappt. (Dasselbe würde gelten, wenn Sie in diesem Beispiel ein Suffix statt des Präfix verwenden würden. Das Root-Suffix überlappt mit allen anderen Suffixen.)

```
<NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-notification-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-notification-two</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>images</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration>
```

Die folgende Benachrichtigungskonfiguration ist ungültig, weil sie überlappende Suffixe hat. Zwei Suffixe werden als überlappend betrachtet, wenn eine Zeichenfolge mit beiden Suffixen enden kann. Eine Zeichenkette kann mit `jpg` und `pg` enden, die Suffixe überlappen sich also. (Gleiches gilt für Präfixe. Zwei Präfixe werden als überlappend betrachtet, wenn eine Zeichenfolge mit beiden Präfixen beginnen kann.)

```
 <NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>jpg</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-two</Topic>
         <Event>s3:ObjectCreated:Put</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>pg</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration
```

Die folgende Benachrichtigungskonfiguration ist ungültig, weil sie überlappende Präfixe und Suffixe hat.

```
<NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>images</Value>
```

```
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>jpg</Value>
                </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-two</Topic>
         <Event>s3:ObjectCreated:Put</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>jpg</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration>
```

Erteilen von Berechtigungen zur Veröffentlichung von Ereignisbenachrichtigungsmeldungen an einem Ziel

Damit Amazon S3 Nachrichten zu einem Ziel veröffentlichen kann, müssen Sie dem Amazon S3-Prinzipal die notwendigen Berechtigungen zum Aufruf der jeweiligen API für die Veröffentlichung von Nachrichten zu einem SNS-Thema, einer SQS-Warteschlange oder einer Lambda-Funktion erteilen.

Erteilen von Berechtigungen zum Aufrufen einer AWS Lambda-Funktion

Amazon S3 veröffentlicht Ereignisnachrichten an AWS Lambda durch den Aufruf einer Lambda-Funktion und die Bereitstellung der Ereignisnachricht als Argument.

Wenn Sie die Amazon S3-Konsole zur Konfiguration von Ereignisbenachrichtigungen in einem Amazon S3-Bucket für eine Lambda-Funktion verwenden, richtet die Konsole die notwendigen Berechtigungen für die Lambda-Funktion ein, sodass Amazon S3 Berechtigungen besitzt, die Funktion vom Bucket aus aufzurufen. Weitere Informationen finden Sie unter Wie kann ich Ereignisbenachrichtigungen für einen S3-Bucket aktivieren und konfigurieren? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Sie können Amazon S3 auch Berechtigungen aus AWS Lambda erteilen, um Ihre Lambda-Funktion aufzurufen. Weitere Informationen finden Sie unter Tutorial: Verwendung von AWS Lambda mit Amazon S3 im AWS Lambda Developer Guide.

Erteilen von Berechtigungen, Meldungen in einem SNS-Thema oder einer SQS-Warteschlange zu veröffentlichen

Um Amazon S3 die Erlaubnis zu erteilen, Nachrichten im SNS-Thema oder in der SQS-Warteschlange zu veröffentlichen, fügen Sie eine AWS Identity and Access Management-(IAM)-Richtlinie an das SNS-Zielthema oder die SQS-Warteschlange an.

Ein Beispiel dafür, wie Sie einem SNS-Thema oder einer SQS-Warteschlange eine Richtlinie zuordnen, finden Sie unter Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema und SQS-Warteschlange) (p. 659). Weitere Informationen über Berechtigungen finden Sie in den folgenden Themen:

- Beispiele für die Amazon SNS-Zugriffskontrolle im Entwicklerhandbuch für Amazon Simple Notification Service
- Zugriffskontrolle mit AWS Identity and Access Management (IAM) im Amazon Simple Queue Service-Entwicklerhandbuch

IAM-Richtlinie für ein SNS-Zielthema

Im Folgenden finden Sie ein Beispiel für eine IAM-Richtlinie, die Sie dem Ziel-SNS-Thema zuordnen.

```
{
 "Version": "2008-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
   "Effect": "Allow",
   "Principal": {
     "Service": "s3.amazonaws.com"
   },
   "Action": [
    "SNS:Publish"
   ],
   "Resource": "arn:aws:sns:REGION:ACCOUNT-ID:TOPICNAME",
   "Condition": {
      "ArnLike": { "aws:SourceArn": "arn:aws:s3:::bucket-name" },
      "StringEquals": { "aws:SourceAccount": "bucket-owner-account-id" }
   }
  }
 ]
}
```

IAM-Richtlinie für eine SQS-Zielwarteschlange

Im Folgenden finden Sie ein Beispiel für eine IAM-Richtlinie, die Sie der Ziel-SQS-Warteschlange zuordnen.

```
{
 "Version": "2008-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
```

```
   "Effect": "Allow",
   "Principal": {
     "Service": "s3.amazonaws.com"
   },
   "Action": [
    "SQS:SendMessage"
   ],
   "Resource": "arn:aws:sqs:REGION:ACCOUNT-ID:QUEUENAMEHERE",
   "Condition": {
      "ArnLike": { "aws:SourceArn": "arn:aws:s3:*:*:bucket-name" }
   }
  }
 ]
}
```

Beachten Sie, dass Sie für die Amazon SNS- und Amazon SQS-IAM-Richtlinien anstelle der Bedingung `ArnLike` die Bedingung `StringLike` in der Richtlinie angeben können.

```
"Condition": {
  "StringLike": { "aws:SourceArn": "arn:aws:s3:*:*:bucket-name" }
  }
```

AWS KMS-Schlüsselrichtlinie

Wenn die SQS-Warteschlange oder SNS-Themen mit einem vom Kunden verwalteten AWS Key Management Service-(AWS KMS-)Kundenhauptschlüssel (Customer Master Key, CMK) verschlüsselt sind, müssen Sie dem Amazon S3-Service-Prinzipal die Berechtigung erteilen, mit den verschlüsselten Themen und/oder der Warteschlange zu arbeiten. Um dem Amazon S3-Service-Prinzipal die Berechtigung zu erteilen, fügen Sie der Schlüsselrichtlinie für den vom Kunden verwalteten CMK die folgende Anweisung hinzu:

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

Weitere Informationen zu AWS KMS-Schlüsselrichtlinien finden Sie unter Verwenden von Schlüsselrichtlinien in AWS KMS im AWS Key Management Service Developer Guide. Weitere Informationen zum Verwenden der serverseitigen Verschlüsselung mit AWS KMS für Amazon SQS und Amazon SNS, finden Sie in den folgenden Informationen:

- Konfigurieren von AWS KMS-Berechtigungen für Amazon SNS im Entwicklerhandbuch für Amazon Simple Notification Service.
- Konfigurieren von AWS KMS-Berechtigungen für Amazon SQS im Amazon Simple Queue Service-Entwicklerhandbuch.

- Verschlüsseln von Nachrichten, die mit AWS KMS in Amazon SNS im AWS Compute Blog veröffentlicht werden.

Walkthrough: Konfigurieren eines Buckets für Benachrichtigungen (SNS-Thema und SQS-Warteschlange)

Themen

Walkthrough-Übersicht

In diesem Beispiel fügen Sie eine Benachrichtigungskonfiguration für einen Bucket hinzu, in von Amazon S3 die folgenden Schritte anfordert:

- Veröffentlichung von Ereignissen des Typs `s3:ObjectCreated:*` zu einer Amazon SQS-Warteschlange.
- Veröffentlichung von Ereignissen des Typs `s3:ReducedRedundancyLostObject` in einem Amazon SNS-Thema.

Weitere Informationen zur Benachrichtigungskonfiguration finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646)

Alle diese Schritte können Sie auf der Konsole erledigen, ohne Code schreiben zu müssen. Darüber hinaus sind hierin Codebeispiele mit AWS SDKs for Java und .NET enthalten, um Sie beim programmgesteuerten Hinzufügen von Benachrichtigungskonfigurationen zu unterstützen.

In dieser schrittweisen Anleitung gehen Sie wie folgt vor:

1. Ein Amazon SNS-Thema erstellen

   Mit der Amazon SNS-Konsole erstellen Sie ein SNS-Thema und abonnieren dieses, sodass alle darin veröffentlichten Ereignisse an Sie weitergegeben werden. Sie geben als Kommunikationsprotokoll E-Mail an. Nachdem Sie ein Thema erstellt haben, sendet Amazon SNS eine E-Mail. Sie müssen auf den Link in der E-Mail klicken, um das Abonnement des Themas zu bestätigen.

   Sie ordnen dem Thema eine Zugriffsrichtlinie zu, um Amazon S3 die Berechtigung zu erteilen, Meldungen zu veröffentlichen.
2. Erstellen einer Amazon SQS-Warteschlange.

   Mit der Amazon SQS-Konsole erstellen Sie eine SQS-Warteschlange. Sie können auf alle Meldungen zugreifen, die Amazon S3 programmgesteuert an die Warteschlange sendet. Für diese schrittweise Anleitung überprüfen Sie die Benachrichtigungsmitteilungen in der Konsole.

   Sie ordnen dem Thema eine Zugriffsrichtlinie zu, um Amazon S3 die Berechtigung zu erteilen, Meldungen zu veröffentlichen.

3. Fügen Sie einem Bucket eine Benachrichtigungskonfiguration hinzu.

Schritt 1: Erstellen eines Amazon SNS-Themas

Gehen Sie wie folgt vor, um ein Amazon Simple Notification Service (Amazon SNS)-Thema zu erstellen und zu abonnieren.

1. Erstellen Sie mit der Amazon SNS-Konsole ein Thema. Anleitung finden Sie unter Erstellen eines Themas im Entwicklerhandbuch für Amazon Simple Notification Service.
2. Abonnieren Sie das Thema. Für diese Übung geben Sie email als Kommunikationsprotokoll an. Anleitung finden Sie unter Abonnieren eines Themas im Entwicklerhandbuch für Amazon Simple Notification Service.

   Sie erhalten eine E-Mail, in der Sie aufgefordert werden, das Abonnement des Themas zu bestätigen. Bestätigen Sie das Abonnement.
3. Ersetzen Sie die dem Thema zugeordnete Zugriffsrichtlinie durch die folgende Richtlinie. Sie müssen die Richtlinie aktualisieren, indem Sie den Amazon-Ressourcenname (ARN) Ihres SNS-Themas und den Bucket-Namen angeben.

```
{
 "Version": "2008-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
   "Effect": "Allow",
   "Principal": {
    "AWS":"*"
   },
   "Action": [
    "SNS:Publish"
   ],
   "Resource": "SNS-topic-ARN",
   "Condition": {
      "ArnLike": { "aws:SourceArn": "arn:aws:s3:*:*:bucket-name" }
   }
  }
 ]
}
```

4. Notieren Sie den ARN des Themas.

   Das von Ihnen erstellte SNS-Thema ist eine weitere Ressource in Ihrem AWS-Konto. Sie hat einen eindeutigen ARN (Amazon-Ressourcenname). Sie benötigen diesen ARN im nächsten Schritt. Der ARN hat das folgende Format:

```
arn:aws:sns:aws-region:account-id:topic-name
```

Schritt 2: Erstellen einer Amazon SQS-Warteschlange

Gehen Sie wie folgt vor, um eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange zu erstellen und zu abonnieren.

1. Verwendung der Amazon SQS-Konsole zum Erstellen einer Warteschlange. Anleitungen finden Sie unter Erste Schritte mit Amazon SQS im Amazon Simple Queue Service-Entwicklerhandbuch.

2. Ersetzen Sie die der Warteschlange zugeordnete Zugriffsrichtlinie durch die folgende Richtlinie. (Wählen Sie in der Amazon SQS-Konsole die Warteschlange aus. Wählen Sie dann auf der Registerkarte Permissions (Berechtigungen) die Option Edit Policy Document (Advanced) (Richtliniendokument bearbeiten (Erweitert)) aus.

```
{
 "Version": "2012-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
   "Effect": "Allow",
   "Principal": {
    "AWS":"*"
   },
   "Action": [
    "SQS:SendMessage"
   ],
   "Resource": "SQS-queue-ARN",
   "Condition": {
      "ArnLike": { "aws:SourceArn": "arn:aws:s3:*:*:bucket-name" }
   }
  }
 ]
}
```

3. (Optional) Wenn die Amazon SQS-Warteschlange oder das Amazon SNS-Thema die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS) verwendet, fügen Sie die folgende Richtlinie zu dem entsprechenden symmetrischen vom Kunden verwalteten AWS KMS-CMK hinzu.

   Sie müssen die Richtlinie einem vom Kunden verwalteten CMK hinzufügen, da Sie den von AWS verwalteten CMK für Amazon SQS oder Amazon SNS nicht ändern können.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

   Weitere Informationen zur Verwendung von SSE für Amazon SQS und Amazon SNS mit AWS KMS finden Sie unter:

   - Konfigurieren von AWS KMS-Berechtigungen für Amazon SNS im Entwicklerhandbuch für Amazon Simple Notification Service.
   - Konfigurieren von AWS KMS-Berechtigungen für Amazon SQS im Amazon Simple Queue Service-Entwicklerhandbuch.

4. Notieren Sie den ARN der Warteschlange.

Die von Ihnen erstellte SQS-Warteschlange ist eine weitere Ressource in Ihrem AWS-Konto. Sie hat einen eindeutigen ARN (Amazon-Ressourcenname). Sie benötigen diesen ARN im nächsten Schritt. Der ARN hat das folgende Format:

```
arn:aws:sqs:aws-region:account-id:queue-name
```

Schritt 3: Hinzufügen einer Benachrichtigungskonfiguration zu Ihrem Bucket

Sie können Bucket-Benachrichtigungen entweder über die Amazon S3-Konsole oder programmgesteuert über AWS SDKs aktivieren. Wählen Sie eine der Optionen für die Konfiguration von Benachrichtigungen über Ihren Bucket. Dieser Abschnitt zeigt Beispiele für die Verwendung des AWS SDKs for Java und .NET.

Schritt 3 (Option a): Aktivieren von Benachrichtigungen über einen Bucket unter Verwendung der Konsole

Fügen Sie mithilfe der Amazon S3-Konsole eine Benachrichtigungskonfiguration hinzu, die Amazon S3 zu Folgendem auffordert:

- Veröffentlichung von Ereignissen des Typs `s3:ObjectCreated:*` zu Ihrer Amazon SQS-Warteschlange
- Veröffentlichung von Ereignissen des Typs `s3:ReducedRedundancyLostObject` zu Ihrem Amazon SNS-Thema

Nachdem Sie die Benachrichtigungskonfiguration gespeichert haben, veröffentlicht Amazon S3 eine Testmeldung, die Sie per E-Mail erhalten.

Anleitungen finden Sie unter Wie kann ich Ereignisbenachrichtigungen für einen S3-Bucket aktivieren und konfigurieren? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Schritt 3 (Option b): Aktivieren von Benachrichtigungen über einen Bucket unter Verwendung des AWS SDK for .NET

Das folgende C#-Codebeispiel zeigt ein vollständiges Listing, das einem Bucket eine Benachrichtigungskonfiguration hinzufügt. Sie müssen den Code aktualisieren und Ihren Bucket-Namen und den ARN des SNS-Themas angeben. Weitere Informationen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class EnableNotificationsTest
    {
        private const string bucketName = "*** bucket name ***";
```

```
        private const string snsTopic = "*** SNS topic ARN ***";
        private const string sqsQueue = "*** SQS topic ARN ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            EnableNotificationAsync().Wait();
        }

        static async Task EnableNotificationAsync()
        {
            try
            {
               PutBucketNotificationRequest request = new PutBucketNotificationRequest
                {
                    BucketName = bucketName
                };

                TopicConfiguration c = new TopicConfiguration
                {
                    Events = new List<EventType> { EventType.ObjectCreatedCopy },
                    Topic = snsTopic
                };
                request.TopicConfigurations = new List<TopicConfiguration>();
                request.TopicConfigurations.Add(c);
                request.QueueConfigurations = new List<QueueConfiguration>();
                request.QueueConfigurations.Add(new QueueConfiguration()
                {
                    Events = new List<EventType> { EventType.ObjectCreatedPut },
                    Queue = sqsQueue
                });

                PutBucketNotificationResponse response = await
 client.PutBucketNotificationAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' ",
 e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown error encountered on server. Message:'{0}' ",
 e.Message);
            }
        }
    }
}
```

Schritt 3 (Option c): Aktivieren von Benachrichtigungen über einen Bucket unter Verwendung des AWS SDK for Java

Die folgende Beispiel veranschaulicht, wie Sie einem Bucket eine Benachrichtigungskonfiguration hinzufügen. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.EnumSet;

public class EnableNotificationOnABucket {

    public static void main(String[] args) throws IOException {
        String bucketName = "*** Bucket name ***";
        Regions clientRegion = Regions.DEFAULT_REGION;
        String snsTopicARN = "*** SNS Topic ARN ***";
        String sqsQueueARN = "*** SQS Queue ARN ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();
            BucketNotificationConfiguration notificationConfiguration = new
 BucketNotificationConfiguration();

            // Add an SNS topic notification.
            notificationConfiguration.addConfiguration("snsTopicConfig",
                    new TopicConfiguration(snsTopicARN,
 EnumSet.of(S3Event.ObjectCreated)));

            // Add an SQS queue notification.
            notificationConfiguration.addConfiguration("sqsQueueConfig",
                    new QueueConfiguration(sqsQueueARN,
 EnumSet.of(S3Event.ObjectCreated)));

            // Create the notification configuration request and set the bucket
 notification configuration.
            SetBucketNotificationConfigurationRequest request = new
 SetBucketNotificationConfigurationRequest(
                    bucketName, notificationConfiguration);
            s3Client.setBucketNotificationConfiguration(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Schritt 4: Testen der Einrichtung

Jetzt können Sie die Einrichtung testen, indem Sie ein Objekt in Ihren Bucket hochladen und die Ereignisbenachrichtigung in der Amazon SQS-Konsole überprüfen. Anleitungen finden Sie unter Empfangen einer Nachricht im Amazon Simple Queue Service-Entwicklerhandbuch, Abschnitt „Erste Schritte”..

Struktur von Ereignismeldungen

Die Benachrichtigung, die Amazon S3 sendet, um ein Ereignis zu veröffentlichen, verwendet das JSON-Format. Das folgende Beispiel zeigt die Struktur der JSON-Nachricht.

Beachten Sie Folgendes im Zusammenhang mit dem Beispiel:

- Der `eventVersion`-Schlüsselwert enthält eine Haupt- und eine Nebenversion im Format `<major>.<minor>`.

  Die Hauptversion wird erhöht, wenn Amazon S3 eine Änderung an der Ereignisstruktur vornimmt, die nicht abwärtskompatibel ist. Dies beinhaltet das Entfernen eines JSON-Feldes, das bereits vorhanden ist, oder das Ändern, wie die Inhalte eines Feldes dargestellt werden (Beispiel: ein Datumsformat).

  Die Nebenversion wird erhöht, wenn Amazon S3 neue Felder zur Ereignisstruktur hinzufügt. Dies kann geschehen, wenn neue Informationen für einige oder alle vorhandenen Ereignisse oder neue Informationen zu neu eingeführten Ereignistypen bereitgestellt werden. Anwendungen sollten neue Felder ignorieren, um weiter aufwärtskompatibel mit neuen Nebenversionen der Ereignisstruktur zu sein.

  Falls neue Ereignistypen eingeführt werden, aber die Struktur des Ereignisses anderweitig unmodifiziert ist, ändert sich die Ereignisversion nicht.

  Um sicherzustellen, dass Ihre Anwendungen die Ereignisstruktur ordnungsgemäß analysieren können, empfehlen wir, dass Sie einen Vergleich mit der Hauptversionsnummer durchführen. Um sicherzustellen, dass die von Ihrer Anwendung erwarteten Felder vorhanden sind, empfehlen wir zudem, die Durchführung eines Größer-oder-Gleich-Vergleichs mit der Hauptversion.
- Der Schlüsselwert `responseElements` ist nützlich, wenn Sie eine Anfrage mit AWS Support nachverfolgen möchten. `x-amz-request-id` und `x-amz-id-2` helfen Amazon S3, eine einzelne Anfrage nachzuverfolgen. Diese Werte sind dieselben, die Amazon S3 in der Antwort auf die Anfrage zurückgibt, die die Ereignisse initiiert. Daher können sie verwendet werden, um das Ereignis der Anfrage zuzuordnen.
- Der `s3`-Schlüssel bietet Informationen über den Bucket und das Objekt, die an dem Ereignis beteiligt sind. Der Wert des Objektschlüsselnamens ist URL-kodiert. Beispielsweise wird „red flower.jpg“ zu „red +flower.jpg“ (Amazon S3 gibt als Inhaltstyp in der Antwort „`application/x-www-form-urlencoded`“ zurück).
- Der `sequencer`-Schlüssel bietet eine Möglichkeit, die Reihenfolge von Ereignissen zu bestimmen. Ereignisbenachrichtigungen kommen nicht garantiert in der Reihenfolge an, in der die Ereignisse aufgetreten sind. Benachrichtigungen von Ereignissen, die Objekte erstellen (`PUTs`) und Objekte Löschen, enthalten jedoch einen `sequencer`, der verwendet werden kann, um die Reihenfolge von Ereignissen für einen bestimmten Objektschlüssel zu bestimmen.

  Wenn Sie die `sequencer`-Zeichenfolgen von zwei Ereignisbenachrichtigungen für denselben Objektschlüssel vergleichen, ist die Ereignisbenachrichtigung mit dem größeren hexadezimalen Wert von `sequencer` das später aufgetretene Ereignis. Wenn Sie Ereignisbenachrichtigungen verwenden, um eine separate Datenbank oder einen Index Ihrer Amazon S3-Objekte zu verwalten, sollten Sie während der Verarbeitung der einzelnen Benachrichtigungen die `sequencer`-Werte vergleichen und speichern.

  Beachten Sie Folgendes:

  - Sie können `sequencer` nicht verwenden, um die Reihenfolge von Ereignissen für unterschiedliche Objektschlüssel zu bestimmen.
  - Die Sequenzer können unterschiedliche Längen haben. Um diese Werte zu vergleichen, füllen Sie zuerst den kürzeren Wert links mit Nullen auf und führen dann einen alphabetischen Vergleich durch.
- Der `glacierEventData`-Schlüssel ist nur für `s3:ObjectRestore:Completed`-Ereignisse sichtbar.
- Der Schlüssel `restoreEventData` enthält Attribute, die sich auf Ihre Wiederherstellungsanfrage beziehen.

- Der Schlüssel `replicationEventData` ist nur für Replikationsereignisse sichtbar.

Das folgende Beispiel zeigt Version 2.2 der JSON-Struktur der Ereignisbenachrichtigung. Dies ist die zurzeit von Amazon S3 verwendete Version.

```
{
   "Records":[
      {
         "eventVersion":"2.2",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":The time, in ISO-8601 format, for example, 1970-01-01T00:00:00.000Z,
 when Amazon S3 finished processing the request,
         "eventName":"event-type",
         "userIdentity":{
            "principalId":"Amazon-customer-ID-of-the-user-who-caused-the-event"
         },
         "requestParameters":{
            "sourceIPAddress":"ip-address-where-request-came-from"
         },
         "responseElements":{
            "x-amz-request-id":"Amazon S3 generated request ID",
            "x-amz-id-2":"Amazon S3 host that processed the request"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"ID found in the bucket notification configuration",
            "bucket":{
               "name":"bucket-name",
               "ownerIdentity":{
                  "principalId":"Amazon-customer-ID-of-the-bucket-owner"
               },
               "arn":"bucket-ARN"
            },
            "object":{
               "key":"object-key",
               "size":object-size,
               "eTag":"object eTag",
               "versionId":"object version if bucket is versioning-enabled, otherwise
 null",
               "sequencer": "a string representation of a hexadecimal value used to
 determine event sequence,
                    only used with PUTs and DELETEs"
            }
         },
         "glacierEventData": {
            "restoreEventData": {
               "lifecycleRestorationExpiryTime": "The time, in ISO-8601 format, for
 example, 1970-01-01T00:00:00.000Z, of Restore Expiry",
               "lifecycleRestoreStorageClass": "Source storage class for restore"
            }
         }
      }
   ]
}
```

Das folgende Beispiel zeigt Version 2.0 der Struktur der Ereignisnachricht. Diese Version wird nicht mehr von Amazon S3 verwendet.

```
{
   "Records":[
      {
         "eventVersion":"2.0",
```

```
        "eventSource":"aws:s3",
        "awsRegion":"us-west-2",
        "eventTime":The time, in ISO-8601 format, for example, 1970-01-01T00:00:00.000Z,
when S3 finished processing the request,
        "eventName":"event-type",
        "userIdentity":{
           "principalId":"Amazon-customer-ID-of-the-user-who-caused-the-event"
        },
        "requestParameters":{
           "sourceIPAddress":"ip-address-where-request-came-from"
        },
        "responseElements":{
           "x-amz-request-id":"Amazon S3 generated request ID",
           "x-amz-id-2":"Amazon S3 host that processed the request"
        },
        "s3":{
           "s3SchemaVersion":"1.0",
           "configurationId":"ID found in the bucket notification configuration",
           "bucket":{
              "name":"bucket-name",
              "ownerIdentity":{
                 "principalId":"Amazon-customer-ID-of-the-bucket-owner"
              },
              "arn":"bucket-ARN"
           },
           "object":{
              "key":"object-key",
              "size":object-size,
              "eTag":"object eTag",
              "versionId":"object version if bucket is versioning-enabled, otherwise
null",
              "sequencer": "a string representation of a hexadecimal value used to
determine event sequence,
                   only used with PUTs and DELETEs"
           }
        }
     }
  ]
}
```

Hier einige Beispielmeldungen:

- Testnachricht – Wenn Sie eine Ereignisbenachrichtigung für einen Bucket konfigurieren, sendet Amazon S3 die folgende Testnachricht.

```
{
   "Service":"Amazon S3",
   "Event":"s3:TestEvent",
   "Time":"2014-10-13T15:57:02.089Z",
   "Bucket":"bucketname",
   "RequestId":"5582815E1AEA5ADF",
   "HostId":"8cLeGAmw098X5cv4Zkwcmo8vvZa3eH3eKxsPzbB9wrR+YstdA6Knx4Ip8EXAMPLE"
}
```

- Beispielnachricht, wenn ein Objekt mit der PUT-Anfrage erstellt wird – Die folgende Meldung ist ein Beispiel für eine Meldung, die Amazon S3 sendet, um ein `s3:ObjectCreated:Put`-Ereignis zu veröffentlichen.

```
{
   "Records":[
      {
         "eventVersion":"2.1",
         "eventSource":"aws:s3",
```

```
         "awsRegion":"us-west-2",
         "eventTime":"1970-01-01T00:00:00.000Z",
         "eventName":"ObjectCreated:Put",
         "userIdentity":{
            "principalId":"AIDAJDPLRKLG7UEXAMPLE"
         },
         "requestParameters":{
            "sourceIPAddress":"127.0.0.1"
         },
         "responseElements":{
            "x-amz-request-id":"C3D13FE58DE4C810",
            "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"testConfigRule",
            "bucket":{
               "name":"mybucket",
               "ownerIdentity":{
                  "principalId":"A3NL1KOZZKExample"
               },
               "arn":"arn:aws:s3:::mybucket"
            },
            "object":{
               "key":"HappyFace.jpg",
               "size":1024,
               "eTag":"d41d8cd98f00b204e9800998ecf8427e",
               "versionId":"096fKKXTRTtl3on89fVO.nfljtsv6qko",
               "sequencer":"0055AED6DCD90281E5"
            }
         }
      }
   ]
}
```


Eine Definition der einzelnen IAM-Identifikationspräfixe (AIDA, AROA, AGPA etc.) finden Sie unter Grundlegendes zu eindeutigen ID-Präfixen.

Replikation

Die Replikation ermöglicht ein automatisches, asynchrones Kopieren von Objekten über Amazon S3-Buckets hinweg. Buckets, die für die Objektreplikation konfiguriert sind, können sich im Besitz desselben AWS-Kontos oder unterschiedlicher Konten befinden. Sie können Objekte zwischen verschiedenen AWS-Regionen oder innerhalb derselben Region kopieren.

Zum Aktivieren der Objektreplikation fügen Sie eine Replikationskonfiguration zu Ihrem Quell-Bucket hinzu. In der Minimalkonfiguration muss Folgendes angegeben sein:

- der Ziel-Bucket, in dem Amazon S3 die Objekte replizieren soll
- eine AWS Identity and Access Management (IAM) -Rolle, die Amazon S3 annimmt, um Objekte in Ihrem Namen zu replizieren

Zusätzliche Konfigurationsoptionen sind verfügbar. Weitere Informationen finden Sie unter Zusätzliche Replikationskonfigurationen (p. 687).

Arten der Objektreplikation

Sie können Objekte zwischen unterschiedlichen AWS-Regionen oder innerhalb derselben AWS-Region replizieren.

- Die regionsübergreifende Replikation (CRR, Cross-Region Replication) wird verwendet, um Objekte über Amazon S3-Buckets hinweg in verschiedene AWS-Regionen zu kopieren.
- Die Replikation innerhalb derselben Region (SRR, Same-Region Replication) wird verwendet, um Objekte über Amazon S3-Buckets hinweg in derselben AWS-Region zu kopieren.

Gründe zur Verwendung der Replikation

Die Replikation unterstützt Sie bei Folgendem:

- Replikation von Objekten unter Beibehaltung von Metadaten – Sie können mithilfe der Replikation Kopien Ihrer Objekte erstellen, die alle Metadaten enthalten, z. B. die ursprüngliche Objekterstellungszeit und Versions-IDs. Diese Funktion ist wichtig, wenn Sie sicherstellen müssen, dass Ihr Replikat mit dem Quellobjekt identisch ist

- Replizieren von Objekten in verschiedene Speicherklassen – Sie können mit der Replikation Objekte direkt in S3 Glacier, S3 Glacier Deep Archive oder andere Speicherklassen im Ziel-Bucket platzieren. Sie können Ihre Daten auch in dieselbe Speicherklasse replizieren und Lebenszyklusrichtlinien im Ziel-Bucket verwenden, um Ihre Objekte bei fortschreitendem Alter in eine seltener genutzte Speicherklasse zu verschieben.

- Pflegen von Objektkopien unter verschiedener Eigentümerschaft – Unabhängig davon, wer das Quellobjekt besitzt, können Sie Amazon S3 anweisen, den Replikateigentümer auf das AWS-Konto

zu ändern, das den Ziel-Bucket besitzt. Diese Instance wird als Eigentümer-Überschreibungs-Option bezeichnet. Sie können diese Option nutzen, um den Zugriff auf Objektreplikate einzuschränken.

- Replizieren von Objekten innerhalb von 15 Minuten – Sie können Ihre Daten mit S3 Replication Time Control (S3 RTC) in derselben AWS-Region oder in verschiedenen Regionen in einem vorhersehbaren Zeitraum replizieren. S3 RTC repliziert 99,99 Prozent der neuen in Amazon S3 gespeicherten Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement). Weitere Informationen finden Sie unter the section called “S3 Replication Time Control (S3 RTC)” (p. 687).

Wann sollte CRR verwendet werden?

Die regionsübergreifende Replikation unterstützt Sie bei Folgendem:

- Einhalten der Compliance-Anforderungen – Auch wenn Amazon S3 Ihre Daten standardmäßig in mehreren entfernten Availability Zones speichert, machen es die Compliance-Anforderungen möglicherweise erforderlich, Daten in noch größeren Entfernungen zu speichern. Die regionsübergreifende Replikation ermöglicht es, Daten zwischen entfernten AWS-Regionen zu replizieren, um diese Anforderungen zu erfüllen.

- Minimieren der Latenz – Wenn Sich Ihre Kunden an zwei geografischen Standorten befinden, können Sie die Latenz beim Zugriff auf Objekte verringern. Belassen Sie dazu Objektkopien in AWS-Regionen, die geografisch näher bei Ihren Benutzern liegen.

- Erhöhen der betrieblichen Effizienz – Wenn Sie über Datenverarbeitungscluster in zwei verschiedenen AWS-Regionen verfügen, können Sie Objektkopien in diesen Regionen aufbewahren.

Wann sollte SRR verwendet werden?

Die Replikation in derselben Region unterstützt Sie bei Folgendem:

- Aggregieren von Protokollen in einen einzelnen Bucket – Wenn Sie Protokolle in mehrere Buckets oder kontoübergreifend speichern, können Sie Protokolle ganz einfach in einen Bucket innerhalb derselben Region replizieren. Dies ermöglicht eine einfachere Protokollverarbeitung an einem einzelnen Standort.

- Konfigurieren der Live-Replikation zwischen Produktions- und Testkonten – Wenn Sie oder Ihre Kunden Produktions- und Testkonten haben, die dieselben Daten nutzen, können Sie Objekte kontoübergreifend replizieren und dabei die Objekt-Metadaten erhalten, indem Sie SRR-Regeln implementieren.

- Einhaltung der Datenschutzgesetze Möglicherweise – Sie müssen mehrere Kopien Ihrer Daten in separaten AWS-Konten innerhalb einer bestimmten Region speichern. Mit der Replikation innerhalb derselben Region können Sie automatisch kritische Daten replizieren, wenn Ihre Daten aufgrund von Compliance-Regelungen im Land bleiben müssen.

Anforderungen für die Replikation

Für die Replikation ist Folgendes erforderlich:

- Für das Konto des Quell-Bucket-Eigentümers müssen die AWS-Quellregion und die -Zielregion aktiviert sein. Für das Konto des Ziel-Bucket-Eigentümers muss die Zielregion aktiviert sein. Weitere Informationen zum Aktivieren oder Deaktivieren einer AWS-Region finden Sie unter AWS-Service-Endpunkte im AWS General Reference.
- Für Quell- und Ziel-Buckets muss das Versioning aktiviert sein.
- Amazon S3 muss über die Berechtigungen verfügen, Objekte aus dem Quell-Bucket in Ihrem Namen in den Ziel-Bucket zu replizieren.
- Wenn der Eigentümer des Quell-Buckets das Objekt im Bucket nicht besitzt, muss der Objekteigentümer dem Bucket-Eigentümer in der Zugriffskontrollliste (ACL) des Objekts die Berechtigungen `READ` und `READ_ACP` gewähren. Weitere Informationen finden Sie unter Zugriffsverwaltung mit ACLs (p. 479).
- Wenn für den Quell-Bucket S3 Objektsperre aktiviert ist, muss S3 Objektsperre auch für den Ziel-Bucket aktiviert sein. Weitere Informationen finden Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).

  Um die Replikation auf einem Bucket mit aktivierter Objektsperre zu aktivieren, wenden Sie sich an den AWS Support.

Weitere Informationen finden Sie unter Übersicht über die Replikationseinrichtung (p. 674).

Wenn Sie die Replikationskonfiguration in einem kontenübergreifenden Szenario festlegen, in dem sich Quell- und Ziel-Bucket im Besitz von verschiedenen AWS-Konten befinden, gelten die folgenden zusätzlichen Anforderungen:

- Der Eigentümer des Ziel-Buckets muss dem Eigentümer des Quell-Buckets die Berechtigungen zum Replizieren von Objekten mit einer Bucket-Richtlinie erteilen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, wenn sich Quell- und Ziel-Buckets im Besitz verschiedener AWS-Konten befinden (p. 686).

Was wird von Amazon S3 repliziert?

Amazon S3 repliziert nur bestimmte Elemente in Buckets, die für die Replikation konfiguriert sind.

Was wird repliziert?

Standardmäßig repliziert Amazon S3 Folgendes:

- Objekte, die nach dem Hinzufügen einer Replikationskonfiguration erstellt wurden.
- Unverschlüsselte Objekte.
- Objekte, die im Ruhezustand mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) oder Kundenhauptschlüsseln (CMKs) verschlüsselt, die in AWS Key Management Service (SSE-KMS) gespeichert sind. Um Objekte zu replizieren, die mit CMKs verschlüsselt sind, die in AWS KMS gespeichert sind, müssen Sie die Option explizit aktivieren. Die replizierte Kopie des Objekts wird mit derselben serverseitigen Verschlüsselung verschlüsselt, die für das Quellobjekt verwendet wurde. Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290).
- Objekt-Metadaten.

- Nur die Objekte im Quell-Bucket, für die der Bucket-Eigentümer über Berechtigungen zum Lesen von Objekten und von ACLs verfügt. Weitere Informationen zum Eigentum an Ressourcen finden Sie unter Eigentümerschaft an Amazon S3 Buckets und Objekten (p. 331).

- Aktualisierungen der Objekt-ACL, es sei denn, Sie weisen Amazon S3 an, die Replikatseigentümerschaft zu ändern, wenn sich Quell- und Ziel-Buckets nicht im Besitz derselben Konten befinden Weitere Informationen finden Sie unter Ändern des Replikateigentümers (p. 690).

  Es kann ein wenig dauern, bis Amazon S3 die beiden ACLs synchronisiert hat. Dies gilt nur für Objekte, die nach dem Hinzufügen einer Replikationskonfiguration im Bucket erstellt wurden.

- Objekt-Tags, sofern vorhanden.

- S3 Objektsperre-Aufbewahrungsinformationen, sofern vorhanden. Wenn Amazon S3 Objekte mit angewandten Aufbewahrungsinformationen repliziert, wendet es dieselben Aufbewahrungssteuerungen auf Ihre Replikate an, wodurch der für Ihren Ziel-Bucket konfigurierte Aufbewahrungszeitraum überschrieben wird. Wenn auf die Objekte in Ihrem Quell-Bucket keine Aufbewahrungssteuerungen angewandt sind und Sie sie in einen Ziel-Bucket mit festgelegtem Standard-Aufbewahrungszeitraum replizieren, wir der Standard-Aufbewahrungszeitraum des Ziel-Buckets auf Ihre Objekt-Replikate angewandt. Weitere Informationen finden Sie unter Sperren von Objekten mit S3 Objektsperre (p. 537).

Auswirkungen von Löschvorgängen auf die Replikation

Wenn Sie ein Objekt aus dem Quell-Bucket löschen, geschieht Folgendes:

- Wenn Sie eine DELETE-Anforderung vornehmen, ohne eine Objektversions-ID anzugeben, fügt Amazon S3 eine Löschmarkierung hinzu. Amazon S3 geht mit der Löschmarkierung wie folgt um:
  - Wenn Sie die aktuelle Version der Replikationskonfiguration verwenden, d. h. das `Filter`-Element in einer Regel der Replikationskonfigurationsregel angeben, repliziert Amazon S3 die Löschmarkierung nicht.
  - Wenn Sie das `Filter`-Element nicht angeben, geht Amazon S3 davon aus, dass die Replikationskonfiguration aus einer Version vor V1 stammt. In den früheren Versionen bearbeitete Amazon S3 die Replikation von Löschmarkierungen anders. Weitere Informationen finden Sie unter Abwärtskompatibilität (p. 683).
- Wenn Sie angeben, dass eine Objektversions-ID in einer DELETE-Anforderung gelöscht werden soll, löscht Amazon S3 diese Objektversion im Quell-Bucket. Es repliziert die Löschung aber nicht im Ziel-Bucket. Anders ausgedrückt: Dieselbe Objektversion wird im Ziel-Bucket nicht gelöscht. Dies schützt Daten vor missbräuchlichen Löschungen.

Was wird nicht repliziert?

Standardmäßig repliziert Amazon S3 nicht Folgendes:

- Objekte, die bereits vor dem Hinzufügen der Replikationskonfiguration zum Bucket vorhanden waren. Anders ausgedrückt: Amazon S3 repliziert Objekte nicht rückwirkend.

- Die folgenden verschlüsselten Objekte:
  - Objekte, die mit der serverseitige Verschlüsselung unter Verwendung der vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) erstellt wurden.
  - Objekte, die mit serverseitiger Verschlüsselung unter Verwendung von CMKs erstellt wurden, die in AWS KMS gespeichert sind. Standardmäßig repliziert Amazon S3 keine Objekte, die mit KMS-CMKs verschlüsselt sind. Sie können die Replikation dieser Objekte in der Replikationskonfiguration aber explizit aktivieren und relevante Informationen bereitstellen, damit Amazon S3 diese Objekte replizieren kann.

  Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung (p. 290).

- Objekte, die in der S3 Glacier- oder S3 Glacier Deep Archive-Speicherklasse gespeichert sind. Weitere Informationen zum Amazon S3 Glacier-Service finden Sie im Amazon S3 Glacier-Entwicklerhandbuch.

- Objekte im Quell-Bucket, für die der Bucket-Eigentümer keine Berechtigungen hat (wenn der Bucket-Eigentümer nicht der Eigentümer des Objekts ist). Weitere Informationen darüber, wie ein Objekt-Eigentümer einem Bucket-Eigentümer Berechtigungen erteilen kann, finden Sie unter Erteilung von kontenübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt (p. 453).

- Aktualisierungen von Unterressourcen auf Bucket-Ebene. Wenn Sie beispielsweise die Lebenszykluskonfiguration ändern oder eine Benachrichtigungskonfiguration zu Ihrem Quell-Bucket hinzufügen, werden diese Änderungen nicht auf den Ziel-Bucket angewendet. Dadurch ist es möglich, für den Quell- und den Ziel-Bucket verschiedene Konfigurationen zu nutzen.

- Aktionen, die von der Lebenszykluskonfiguration durchgeführt werden.

  Wenn eine Lebenszykluskonfiguration beispielsweise nur auf Ihrem Quell-Bucket aktiviert ist, erstellt Amazon S3 Löschmarkierungen für abgelaufene Objekte, repliziert diese Markierungen jedoch nicht. Wenn Sie dieselbe Lebenszykluskonfiguration sowohl auf den Quell- als auch auf den Ziel-Bucket anwenden möchten, aktivieren Sie für beide Buckets dieselbe Lebenszykluskonfiguration.

  Weitere Informationen zur Lebenszykluskonfiguration finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

  > **Note**
  >
  > Wenn Sie die aktuelle Version der Replikationskonfiguration verwenden (die XML gibt `Filter` als untergeordnetes Element von `Rule` an), werden Löschmarkierungen, die durch eine Benutzeraktion oder von Amazon S3 als Teil der Lebenszyklusaktion erstellt wurden, nicht repliziert. Wenn Sie jedoch eine frühere Version der Replikationskonfiguration verwenden (die XML gibt `Prefix` als untergeordnetes Element von `Rule` an), werden Löschmarkierungen, die von Benutzeraktionen stammen, repliziert. Weitere Informationen finden Sie unter Abwärtskompatibilität (p. 683).

- Objekte im Quell-Bucket, bei denen es sich um Replikate handelt, die von einer anderen Replikationsregel erstellt wurden.

  Sie können Objekte aus einem Quell-Bucket in nur einem Ziel-Bucket replizieren. Nachdem Amazon S3 ein Objekt repliziert hat, kann es nicht erneut repliziert werden. Wenn Sie beispielsweise den Ziel-Bucket in einer vorhandenen Replikationskonfiguration ändern, repliziert Amazon S3 dieses Objekt nicht erneut.

Ein weiteres Beispiel: Sie konfigurieren eine Replikation, bei der Bucket A die Quelle und Bucket B das Ziel ist. Nehmen wir jetzt an, Sie fügen eine weitere Replikationskonfiguration hinzu, bei der Bucket B die Quelle und Bucket C das Ziel ist. In diesem Fall werden Objekte in Bucket B, die Replikate von Objekten in Bucket A sind, nicht in Bucket C repliziert.

Replizieren vorhandener Objekte

Um die vorhandene Objektreplikation für Ihr Konto zu aktivieren, müssen Sie sich an den AWS Support wenden. Um zu verhindern, dass Ihre Anfrage verzögert wird, geben Sie als Titel für Ihren AWS Support-Fall „Replikation für vorhandene Objekte“ an und schließen Sie unbedingt die folgenden Informationen ein:

- Quell-Bucket
- Ziel-Bucket
- Geschätztes zu replizierende Speichervolumen (in Terabyte)
- Geschätzte Anzahl der zu replizierenden Speicherobjekte

Verwandte Themen

- Replikation (p. 669)
- Übersicht über die Replikationseinrichtung (p. 674)
- Informationen zum Replikationsstatus (p. 721)

Übersicht über die Replikationseinrichtung

Zum Aktivieren der Replikation fügen Sie einfach eine Replikationskonfiguration zu Ihrem Quell-Bucket hinzu. Die Konfiguration weist Amazon S3 an, Objekte wie angegeben zu replizieren. In der Replikationskonfiguration müssen Sie Folgendes angeben:

- den Ziel-Bucket – Der Bucket, in dem Amazon S3 die Objekte replizieren soll.

- Die Objekte, die Sie replizieren möchten – Sie können alle Objekte im Quell-Bucket replizieren oder nur eine Teilmenge davon. Teilmengen identifizieren Sie, indem Sie ein Schlüsselnamenpräfix, mindestens ein Objekt-Tag oder beides in der Konfiguration angeben.

  Wenn Sie beispielsweise eine Replikationsregel konfigurieren, um nur Objekte mit dem Schlüsselnamenpräfix `Tax/` zu replizieren, repliziert Amazon S3 Objekte mit Schlüsseln wie `Tax/doc1` oder `Tax/doc2`. Es repliziert aber keine Objekte mit dem Schlüssel `Legal/doc3`. Wenn Sie sowohl einen Präfix als auch mindestens ein Tag angeben, repliziert Amazon S3 nur Objekte, die dieses Schlüsselpräfix und diese Tags aufweisen.

Ein Replikat besitzt dieselben Schlüsselnamen und Metadaten (z. B. Zeitpunkt der Erstellung, benutzerdefinierte Metadaten oder Versions-ID) wie das Originalobjekt. Amazon S3 verschlüsselt alle Daten bei der Übertragung mit Secure Sockets Layer (SSL).

Zusätzlich zu diesen Mindestanforderungen können Sie die folgenden Optionen auswählen:

- Standardmäßig speichert Amazon S3 Objektreplikate mit derselben Speicherklasse wie das Quellobjekt. Sie können eine andere Speicherklasse für die Replikate festlegen.

- Amazon S3 geht davon aus, dass ein Objektreplikat weiterhin im Besitz des Besitzers des Quellobjekts bleibt. Wenn es also Objekte repliziert, wird auch die entsprechende Objekt-Zugriffskontrollliste (ACL) repliziert. Wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener AWS-Konten befinden, können Sie die Replikation so konfigurieren, dass der Besitzer eines Replikats auf das AWS-Konto geändert wird, dem der Ziel-Bucket gehört.

Zusätzliche Konfigurationsoptionen sind verfügbar. Weitere Informationen finden Sie unter Zusätzliche Replikationskonfigurationen (p. 687).

Amazon S3 stellt auch APIs zur Unterstützung der Einrichtung von Replikationsregeln bereit. Weitere Informationen finden Sie unter den folgenden Themen im Amazon Simple Storage Service API Reference:

- PUT Bucket replication
- GET Bucket replication
- DELETE Bucket replication

Anstatt diese API-Aufrufe direkt aus Ihrem Code heraus auszuführen, können Sie mit dem AWS-SDK, AWS CLI oder der Amazon S3-Konsole eine Replikationskonfiguration zu einem Bucket hinzufügen. Es ist am einfachsten, die Konsole zu verwenden. Beispiele mit Schritt-für-Schritt-Anleitungen finden Sie unter Replikations-Walkthroughs (p. 698).

Wenn Sie noch keine Erfahrung mit der Replikationskonfiguration haben, empfehlen wir Ihnen, zunächst die folgenden Abschnitte zu lesen, bevor Sie sich mit den Beispielen und optionalen Konfigurationen beschäftigen. Beispiele mit Schritt-für-Schritt-Anleitungen zum Einrichten grundlegender Replikationskonfigurationen finden Sie unter Übersicht über die Replikationskonfiguration (p. 675).

Themen

Übersicht über die Replikationskonfiguration

Amazon S3 speichert die Replikationskonfiguration als XML. In der XML-Datei mit der Replikationskonfiguration legen Sie eine AWS Identity and Access Management (IAM)-Rolle und mindestens eine Regel fest.

```
<ReplicationConfiguration>
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
</ReplicationConfiguration>
```

Amazon S3 kann ohne Berechtigung durch Sie keine Objekte replizieren. Sie erteilen Berechtigungen mit der IAM-Rolle, die Sie in der Replikationskonfiguration festlegen. Amazon S3 nimmt die IAM-Rolle an, um in Ihrem Namen Objekte zu replizieren. Sie müssen der IAM-Rolle zunächst die erforderlichen Berechtigungen erteilen. Weitere Informationen zum Verwalten von Berechtigungen finden Sie unter Einrichten von Berechtigungen für die Replikation (p. 684).

In den folgenden Szenarien fügen Sie eine Regel zur Replikationskonfiguration hinzu:

- Sie möchten alle Objekte replizieren.

- Sie möchten eine Teilmenge der Objekte replizieren. Sie identifizieren die Teilmenge der Objekte, indem Sie einen Filter zur Regel hinzufügen. In dem Filter geben Sie ein Objektschlüsselpräfix, Tags oder eine Kombination aus beidem an, um die Objektteilmenge zu identifizieren, für die die Regel gilt.

Sie fügen mehrere Regeln zu einer Replikationskonfiguration hinzu, wenn Sie eine andere Teilmenge von Objekten auswählen möchten. In jeder Regel geben Sie einen Filter an, der eine andere Teilmenge von Objekten auswählt. Beispiel: Sie möchten Objekte mit dem Schlüsselpräfix `tax/` oder `document/` replizieren. In diesem Fall fügen Sie zwei Regeln hinzu und geben in einer Regel als Filter das Schlüsselpräfix `tax/` und in der anderen Regel das Schlüsselpräfix `document/` an.

In den folgenden Abschnitten finden Sie zusätzliche Informationen.

Themen

Grundlegende Regelkonfiguration

Jede Regel muss Status und Priorität der Regel enthalten und angeben, ob Löschmarkierungen repliziert werden sollen.

- `Status` gibt an, ob die Regel aktiviert oder deaktiviert ist. Wenn eine Regel deaktiviert ist, führt Amazon S3 die in der Regel angegebenen Aktionen nicht durch.
- `Priority` gibt an, welche Regel Priorität hat, wenn mehrere Regeln für ein Objekt gelten.
- Aktuell werden Löschmarkierungen nicht repliziert. Daher müssen Sie `DeleteMarkerReplication` auf `Disabled` festlegen.

In der Zielkonfiguration müssen Sie den Namen des Buckets angeben, in den Amazon S3 Objekte replizieren soll.

Der folgende Code zeigt die Minimalanforderungen für eine Regel.

```
...
    <Rule>
        <ID>Rule-1</ID>
        <Status>rule-Enabled-or-Disabled</Status>
        <Priority>integer</Priority>
        <DeleteMarkerReplication>
           <Status>Disabled</Status>
        </DeleteMarkerReplication>
        <Destination>
           <Bucket>arn:aws:s3:::bucket-name</Bucket>
        </Destination>
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
...
```

Sie können auch andere Konfigurationsoptionen festlegen. Beispiel: Sie möchten für Objektreplikate eine andere Speicherklasse verwenden als für das Quellobjekt.

Optional: Festlegen eines Filters

Um eine Teilmenge von Objekten auszuwählen, für die die Regel gilt, fügen Sie einen optionalen Filter hinzu. Sie können nach Objektschlüsselpräfix, Objekt-Tags oder einer Kombination aus beidem filtern. Wenn Sie sowohl nach Schlüsselpräfix als auch nach Objekt-Tags filtern, kombiniert Amazon S3 die Filter mit einem logischen UND-Operator. Anders ausgedrückt: Die Regel wird auf eine Teilmenge von Objekten mit einem bestimmten Schlüsselpräfix und bestimmten Tags angewendet.

Um eine Regel mit einem Filter nach Objektschlüsselpräfix festzulegen, verwenden Sie den folgenden Code. Sie können nur ein Präfix angeben.

```
<Rule>
    ...
    <Filter>
        <Prefix>key-prefix</Prefix>
    </Filter>
    ...
</Rule>
...
```

Um eine Regel mit einem Filter nach Objekt-Tags festzulegen, verwenden Sie den folgenden Code. Sie können einen oder mehrere Objekt-Tags festlegen.

```
<Rule>
    ...
    <Filter>
        <And>
            <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
            </Tag>
            <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
            </Tag>
             ...
        </And>
    </Filter>
    ...
</Rule>
...
```

Um eine Regel mit einem Filter nach einer Kombination aus Schlüsselpräfix und Objekt-Tags festzulegen, verwenden Sie den folgenden Code. Sie umschließen diese Filter in einem übergeordneten `AND`-Element. Amazon S3 führt eine logische `AND`-Operation durch, um diese Filter zu kombinieren.//// Anders ausgedrückt: Die Regel wird auf eine Teilmenge von Objekten mit einem bestimmten Schlüsselpräfix und bestimmten Tags angewendet.

```
<Rule>
    ...
    <Filter>
        <And>
            <Prefix>key-prefix</Prefix>
            <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
            </Tag>
            <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
            </Tag>
```

```
            ...
    </Filter>
    ...
</Rule>
...
```

Zusätzliche Zielkonfigurationen

In der Zielkonfiguration müssen Sie den Bucket angeben, in den Amazon S3 Objekte replizieren soll. Sie können Konfigurationen einrichten, um Objekte aus einem Quell-Bucket in einen Ziel-Bucket zu replizieren. Wenn Sie mehrere Regeln zu einer Replikationskonfiguration hinzufügen, müssen alle Regeln denselben Ziel-Bucket angeben.

```
...
<Destination>
    <Bucket>arn:aws:s3:::destination-bucket</Bucket>
</Destination>
...
```

Sie können die folgenden Optionen im `<Destination>`-Element hinzufügen:

- Sie können die Speicherklasse für die Objektreplikate festlegen. Standardmäßig verwendet Amazon S3 wie im folgenden Beispiel die Speicherklasse des Quellobjekts zum Erstellen von Objektreplikaten.

  ```
  ...
  <Destination>
        <Bucket>arn:aws:s3:::destinationbucket</Bucket>
        <StorageClass>storage-class</StorageClass>
  </Destination>
  ...
  ```

- Wenn Quell- und Ziel-Bucket nicht denselben Konten gehören, können Sie den Besitzer des Replikats auf das AWS-Konto ändern, das den Ziel-Bucket besitzt, indem Sie das Element `AccessControlTranslation` hinzufügen.

  ```
  ...
  <Destination>
     <Bucket>arn:aws:s3:::destinationbucket</Bucket>
     <Account>destination-bucket-owner-account-id</Account>
     <AccessControlTranslation>
         <Owner>Destination</Owner>
     </AccessControlTranslation>
  </Destination>
  ...
  ```

  Wenn Sie dieses Element nicht zur Replikationskonfiguration hinzufügen, gehören die Replikate demselben AWS-Konto, dem das Quellobjekt gehört. Weitere Informationen finden Sie unter Ändern des Replikateigentümers (p. 690).

- Sie können S3 Replication Time Control (S3 RTC) in Ihrer Replikationskonfiguration aktivieren. S3 RTC repliziert die meisten Objekte in Sekunden und 99,99 Prozent der Objekte innerhalb von 15 Minuten (gestützt auf ein Service Level Agreement).

  Note

  Nur ein gültiger Wert von `<Minutes>15</Minutes>` wird für `EventThreshold` und `Time` akzeptiert.

  ```
  ...
  • <Destination>
  ```

```
• ...
•         <Bucket>arn:aws:s3:::destinationbucket</Bucket>
•              <Metrics>
•                  <Status>Enabled</Status>
•                  <EventThreshold>
•                      <Minutes>15</Minutes>
•                  </EventThreshold>
•              </Metrics>
•              <ReplicationTime>
•                  <Status>Enabled</Status>
•                  <Time>
•                      <Minutes>15</Minutes>
•                  </Time>
•              <ReplicationTime>...
• </Destination>

...
```

Weitere Informationen finden Sie unter Replizieren von Objekten mit S3 Replication Time Control (S3 RTC) (p. 687). API-Beispiele finden Sie unter PutBucketReplication im Amazon Simple Storage Service API Reference.

- Ihr Quell-Bucket enthält möglicherweise Objekte, die mit serverseitiger Verschlüsselung mittels der in AWS KMS gespeicherten Schlüssel erstellt wurden. Standardmäßig repliziert Amazon S3 diese Objekte nicht. Optional können Sie Amazon S3 mit dem Replizieren dieser Objekte beauftragen. Entscheiden Sie sich zunächst explizit für diese Funktion, indem Sie das `SourceSelectionCriteria`-Element hinzufügen, und stellen Sie dann den AWS KMS-CMK (für die AWS-Region des Ziel-Buckets) zur Verschlüsselung von Objektreplikaten bereit.

```
...
<SourceSelectionCriteria>
  <SseKmsEncryptedObjects>
    <Status>Enabled</Status>
  </SseKmsEncryptedObjects>
</SourceSelectionCriteria>
<Destination>
  <Bucket>arn:aws:s3:::dest-bucket-name</Bucket>
  <EncryptionConfiguration>
    <ReplicaKmsKeyID>AWS KMS key IDs to use for encrypting object replicas</
ReplicaKmsKeyID>
  </EncryptionConfiguration>
</Destination>
...
```

Weitere Informationen finden Sie unter Replizieren von Objekten, die mit der serverseitigen Verschlüsselung (SSE) unter Verwendung von in AWS KMS gespeicherten Verschlüsselungsschlüsseln erstellt wurden (p. 693).

Beispiele für Replikationskonfigurationen

Fügen Sie zunächst die folgenden Beispiel-Replikationskonfigurationen zu Ihrem Bucket hinzu.

> **Important**
>
> Um eine Replikationskonfiguration zu einem Bucket hinzuzufügen, benötigen Sie die `iam:PassRole`-Berechtigung. Diese Berechtigung erlaubt es Ihnen, die IAM-Rolle weiterzugeben, die Amazon S3 die Replikationsberechtigungen erteilt. Sie legen die IAM-Rolle fest, indem Sie den Amazon-Ressourcennamen (ARN) angeben, der im `Role`-Element in der XML-Datei der Replikationskonfiguration verwendet wird. Weitere Informationen finden Sie unter Erteilen von Benutzerberechtigungen zur Übergabe einer Rolle an einen AWS-Service in der IAM-Benutzerhandbuch.

Example 1: Replikationskonfiguration mit einer Regel

Die folgende grundlegende Replikationskonfiguration legt eine Regel fest. Diese Regel legt eine IAM-Rolle, die Amazon S3 annehmen kann, sowie einen Ziel-Bucket für die Objektreplikate an. Der `Status` der Regel gibt an, dass die Regel aktiviert ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::AcctID:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>

    <Destination><Bucket>arn:aws:s3:::destinationbucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Um eine Teilmenge von Objekten für die Replikation auszuwählen, können Sie einen Filter hinzufügen. In der folgenden Konfiguration gibt der Filter ein Objektschlüsselpräfix an. Diese Regel gilt für Objekte mit dem Schlüsselnamenpräfix `Tax/`.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::AcctID:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>

    <Filter>
       <Prefix>Tax/</Prefix>
    </Filter>

    <Destination><Bucket>arn:aws:s3:::destinationbucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Wenn Sie das `Filter`-Element angeben, müssen Sie auch die Elemente `Priority` und `DeleteMarkerReplication` einschließen. In diesem Beispiel ist die Priorität nicht relevant, da nur eine Regel vorhanden ist.

In der folgenden Konfiguration gibt der Filter ein Präfix und zwei Tags an. Die Regel gilt für eine Untermenge der Objekte, die das angegebene Schlüsselpräfix und die angegebenen Tags aufweisen. Insbesondere gilt die Regel für Objekte, die das Präfix `Tax/` im Schlüsselnamen und die zwei festgelegten Objekt-Tags aufweisen. Die Priorität gilt nicht, da nur eine Regel vorhanden ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::AcctID:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>

    <Filter>
        <And>
```

```
          <Prefix>Tax/</Prefix>
          <Tag>
             <Tag>
                <Key>tagA</Key>
                <Value>valueA</Value>
             </Tag>
          </Tag>
          <Tag>
             <Tag>
                <Key>tagB</Key>
                <Value>valueB</Value>
             </Tag>
          </Tag>
       </And>

    </Filter>

    <Destination><Bucket>arn:aws:s3:::destinationbucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Sie können eine Speicherklasse für die Objektreplikate wie folgt festlegen.

```
<?xml version="1.0" encoding="UTF-8"?>

<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Destination>
       <Bucket>arn:aws:s3:::destinationbucket</Bucket>
       <StorageClass>storage-class</StorageClass>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Sie können jede Speicherklasse festlegen, die Amazon S3 unterstützt.

Example 2: Replikationskonfiguration mit zwei Regeln

Example

In der folgenden Replikationskonfiguration:

- Jede Regel filtert nach einem anderen Schlüsselpräfix, sodass jede Regel für eine distinkte Objektteilmenge gilt. Amazon S3 repliziert Objekte mit den Schlüsselnamen `Tax/doc1.pdf` und `Project/project1.txt`, nicht aber Objekte mit dem Schlüsselnamen `PersonalDoc/documentA`.
- Die Regelpriorität ist nicht relevant, da die Regeln für zwei unterschiedliche Mengen an Objekten gelten. Das nächste Beispiel zeigt, was geschieht, wenn die Regelpriorität angewendet wird.
- Die zweite Regel gibt eine Speicherklasse für Objektreplikate an. Amazon S3 verwendet die festgelegte Speicherklasse für diese Objektreplikate.
- Beide Regeln geben denselben Ziel-Bucket an. Sie können nur einen Ziel-Bucket angeben, gleich, wie viele Regeln Sie festlegen.

```
<?xml version="1.0" encoding="UTF-8"?>

<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
```

```
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>Tax</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Destination>
      <Bucket>arn:aws:s3:::destinationbucket</Bucket>
    </Destination>
     ...
  </Rule>
 <Rule>
    <Status>Enabled</Status>
    <Priority>2</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>Project</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Destination>
      <Bucket>arn:aws:s3:::destinationbucket</Bucket>
     <StorageClass>STANDARD_IA</StorageClass>
    </Destination>
     ...
  </Rule>


</ReplicationConfiguration>
```

Example 3: Replikationskonfiguration mit zwei Regeln mit einander überlappenden Präfixen

In dieser Konfiguration geben die zwei Regeln Filter mit einander überlappenden Schlüsselpräfixen an – `star/` und `starship`. Beide Regeln gelten für Objekte mit dem Schlüsselnamen `starship-x`. In diesem Fall nutzt Amazon S3 die Regelpriorität, um zu bestimmen, welche Regel angewendet werden soll.

```
<ReplicationConfiguration>

  <Role>arn:aws:iam::AcctID:role/role-name</Role>

  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>star</Prefix>
    </Filter>
    <Destination>
      <Bucket>arn:aws:s3:::destinationbucket</Bucket>
    </Destination>
  </Rule>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
```

```
        <Prefix>starship</Prefix>
    </Filter>
    <Destination>
      <Bucket>arn:aws:s3:::destinationbucket</Bucket>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Example 4: Beispielanleitungen

Beispielanleitungen finden Sie unter Replikations-Walkthroughs (p. 698).

Weitere Informationen zur XML-Struktur der Replikationskonfiguration finden Sie unter PutBucketReplication im Amazon Simple Storage Service API Reference.

Abwärtskompatibilität

Die aktuelle Version der Replikationskonfigurations-XML ist V2. Für die Abwärtskompatibilität unterstützt Amazon S3 auch weiterhin die V1-Konfiguration. Wenn Sie die Replikationskonfigurations-XML V1 verwendet haben, berücksichtigen Sie die folgenden Probleme, die die Abwärtskompatibilität beeinträchtigen:

- Die Replikationskonfigurations-XML V2 enthält das `Filter`-Element für Regeln. Mit dem `Filter`-Element können Sie Objektfilter basierend auf dem Objektschlüsselpräfix, Tags oder einer Kombination aus beidem angeben, um die Objekte festzulegen, für die die Regel gilt. Die in der Replikationskonfiguration XML V1 unterstützte Filterung, die ausschließlich auf dem Schlüsselpräfix basiert. In diesem Fall fügen Sie `Prefix` wie im folgenden Beispiel direkt als untergeordnetes Element des Elements `Rule` hinzu.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::AcctID:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Prefix>key-prefix</Prefix>
    <Destination><Bucket>arn:aws:s3:::destinationbucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

  Für die Abwärtskompatibilität unterstützt `Amazon S3` auch weiterhin die V1-Konfiguration.

- Wenn Sie ein Objekt aus Ihrem Quell-Bucket löschen, ohne eine Objektversions-ID anzugeben, fügt Amazon S3 eine Löschmarkierung hinzu. Wenn Sie V1 der Replikationskonfigurations-XML verwenden, repliziert Amazon S3 Löschmarkierungen, die von Benutzeraktionen herstammen. Anders gesagt – wenn der Benutzer das Objekt gelöscht hat, nicht wenn Amazon S3 es wegen Ablaufs als Teil einer Lebenszyklusaktion gelöscht hat. In V2 repliziert Amazon S3 keine Löschmarkierungen. Deshalb müssen Sie das `DeleteMarkerReplication`-Element auf `Disabled` setzen.

```
...
    <Rule>
        <ID>Rule-1</ID>
        <Status>rule-Enabled-or-Disabled</Status>
        <Priority>integer</Priority>
        <DeleteMarkerReplication>
           <Status>Disabled</Status>
        </DeleteMarkerReplication>
        <Destination>
           <Bucket>arn:aws:s3:::bucket-name</Bucket>
        </Destination>
    </Rule>
```

```
...
```

Einrichten von Berechtigungen für die Replikation

Wenn Sie die Replikation einrichten, müssen Sie die erforderlichen Berechtigungen wie folgt erwerben:

- Erstellen Sie eine IAM-Rolle – Amazon S3 benötigt Berechtigungen, um Objekte in Ihrem Namen zu replizieren. Sie erteilen diese Berechtigungen, indem Sie eine IAM-Rolle erstellen und die Rolle in der Replikationskonfiguration festlegen.
- Wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener AWS-Konten befinden, muss der Eigentümer des Ziel-Buckets dem Quell-Bucket-Eigentümer die Berechtigungen zum Speichern der Replikate erteilen.

Themen

Erstellen einer IAM-Rolle

Standardmäßig sind alle Amazon S3-Ressourcen – Buckets, Objekte und zugehörige Unterressourcen – privat: Nur der Ressourceninhaber kann auf die Ressource zugreifen. Um Objekte aus dem Quell-Bucket zu lesen und im Ziel-Bucket zu replizieren, benötigt Amazon S3 die Berechtigungen zum Durchführen dieser Aufgaben. Sie erteilen diese Berechtigungen, indem Sie eine IAM-Rolle erstellen und die Rolle in der Replikationskonfiguration festlegen.

In diesem Abschnitt werden die Vertrauensrichtlinie und die mindestens erforderliche Berechtigungsrichtlinie erläutert. Diese Beispielanleitungen enthalten Schritt-für-Schritt-Anweisungen zum Erstellen einer IAM-Rolle. Weitere Informationen finden Sie unter Replikations-Walkthroughs (p. 698).

- Im Folgenden wird eine Vertrauensrichtlinie gezeigt, bei der Sie Amazon S3 als den Service-Prinzipal identifizieren, der die Rolle übernehmen kann.

```
{

   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

  Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.
- Im Folgenden wird eine Zugriffsrichtlinie gezeigt, bei der Sie der Rolle die Berechtigungen erteilen, Replikationsaufgaben in Ihrem Namen durchzuführen. Wenn Amazon S3 die Rolle annimmt, verfügt es über die Berechtigungen, die Sie in dieser Richtlinie angeben.

```
{
```

```
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[

            "s3:GetObjectVersion",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"

         ],
         "Resource":[
            "arn:aws:s3:::source-bucket/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags"
         ],
         "Resource":"arn:aws:s3:::destination-bucket/*"
      }
   ]
}
```

Die Zugriffsrichtlinie erteilt Berechtigungen für folgenden Aktionen:

- `s3:GetReplicationConfiguration` und `s3:ListBucket` – Berechtigungen für diese Aktionen für den Quell-Bucket gestatten es Amazon S3, die Replikationskonfiguration abzurufen und die Bucket-Inhalte aufzulisten (das aktuelle Berechtigungsmodell erfordert die `s3:ListBucket`-Berechtigung für den Zugriff auf Löschmarkierungen).
- `s3:GetObjectVersion` und `s3:GetObjectVersionAcl` – Berechtigungen für diese Aktionen, die für alle Objekte erteilt wurden, gestatten es Amazon S3, eine bestimmte Objektversion zu erhalten und auf die mit Objekten verknüpfte Zugriffskontrollliste (ACL) zuzugreifen.
- `s3:ReplicateObject` und `s3:ReplicateDelete` – Berechtigungen für diese Aktionen für Objekte im Ziel-Bucket gestatten es Amazon S3, Objekte oder Löschmarkierungen in den Ziel-Bucket zu replizieren. Informationen zu Löschmarkierungen finden Sie unter Auswirkungen von Löschvorgängen auf die Replikation (p. 672).

  Note

  Berechtigungen für die Aktion `s3:ReplicateObject` für den Ziel-Bucket erlauben auch die Replikation von Objekt-Tags, sodass Sie die Berechtigung für die Aktion `s3:ReplicateTags` nicht explizit erteilen müssen.
- `s3:GetObjectVersionTagging` – Berechtigungen für diese Aktion für Objekte im Quell-Bucket gestatten es Amazon S3, Objekt-Tags für die Replikation zu lesen (siehe Markieren von Objekten (p. 130)). Wenn Amazon S3 nicht über diese Berechtigungen verfügt, repliziert es die Objekte, aber nicht die Objekt-Tags.

Eine Liste der Amazon S3-Aktionen finden Sie unter Amazon S3-Aktionen (p. 380).

Important

Das AWS-Konto, das die IAM-Rolle besitzt, muss über Berechtigungen für die Aktionen verfügen, die es der IAM-Rolle erteilt.
Angenommen, der Quell-Bucket enthält beispielsweise Objekte, die im Besitz eines anderen AWS-Kontos sind. Der Eigentümer der Objekte muss dem AWS-Konto, das die IAM-Rolle besitzt, die erforderlichen Berechtigungen explizit über die Objekt-ACL erteilen. Andernfalls kann Amazon S3 nicht auf die Objekte zugreifen, und die Replikation dieser Objekte schlägt fehl. Weitere Informationen zu ACL-Berechtigungen finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht (p. 480).
Die hier beschriebenen Berechtigungen gehören zur Mindestreplikationskonfiguration. Wenn Sie optionale Replikationskonfigurationen hinzufügen möchten, müssen Sie Amazon S3 zusätzliche Berechtigungen erteilen. Weitere Informationen finden Sie unter Zusätzliche Replikationskonfigurationen (p. 687).

Erteilen von Berechtigungen, wenn sich Quell- und Ziel-Buckets im Besitz verschiedener AWS-Konten befinden

Wenn sich der Quell- und der Ziel-Bucket im Besitz von unterschiedlichen Konten befinden, muss der Eigentümer des Ziel-Buckets auch eine Bucket-Richtlinie hinzufügen, um dem Eigentümer des Quell-Buckets die Berechtigung zum Ausführen von Replikationsaktionen wie folgt zu erteilen.

```
{
   "Version":"2008-10-17",
   "Id":"PolicyForDestinationBucket",
   "Statement":[
      {
         "Sid":"1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"SourceBucket-AcctID"
         },
         "Action":[
            "s3:ReplicateDelete",
            "s3:ReplicateObject"
         ],
         "Resource":"arn:aws:s3:::destinationbucket/*"
      },
      {
         "Sid":"2",
         "Effect":"Allow",
         "Principal":{
            "AWS":"SourceBucket-AcctID"
         },
         "Action": [
            "s3:List*",
            "s3:GetBucketVersioning",
            "s3:PutBucketVersioning"
         ],
         "Resource":"arn:aws:s3:::destinationbucket"
      }
   ]
}
```

Ein Beispiel finden Sie unter Beispiel 2: Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Besitz verschiedener Konten befinden. (p. 708).

Wenn Objekte im Quell-Bucket mit einem Tag versehen sind, beachten Sie Folgendes:

- Wenn der Eigentümer des Quell-Buckets Amazon S3 die Berechtigung für die Aktionen `s3:GetObjectVersionTagging` und `s3:ReplicateTags` zum Replizieren von Objekt-Tags (über die IAM-Rolle) erteilt, repliziert Amazon S3 die Tags zusammen mit den Objekten. Weitere Information zur IAM-Rolle finden Sie unter Erstellen einer IAM-Rolle (p. 684).
- Wenn der Eigentümer des Ziel-Buckets die Tags nicht replizieren will, kann er die folgende Anweisung zur Richtlinie für den Ziel-Bucket hinzufügen, um die Berechtigung für die Aktion `s3:ReplicateTags` explizit zu verweigern.

```
...
   "Statement":[
      {
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-AcctID:root"
         },
         "Action":["s3:ReplicateTags"],
         "Resource":"arn:aws:s3:::destinationbucket/*"
      }
   ]
...
```

Ändern der Replikateigentümerschaft

Wenn verschiedene AWS-Konten Quell- und Ziel-Bucket besitzen, können Sie Amazon S3 anweisen, die Eigentümerschaft des Replikats zu dem AWS-Konto zu ändern, das den Ziel-Bucket besitzt. Dies wird als Eigentümer-Überschreibungs-Option bezeichnet. Weitere Informationen finden Sie unter Ändern des Replikateigentümers (p. 690).

Zusätzliche Replikationskonfigurationen

In diesem Abschnitt werden zusätzliche Optionen zur Replikationskonfiguration beschrieben, die in Amazon S3 verfügbar sind. Weitere Informationen zum Erstellen einer grundlegenden Replikationskonfiguration finden Sie unter Übersicht über die Replikationseinrichtung (p. 674).

Themen

Replizieren von Objekten mit S3 Replication Time Control (S3 RTC)

S3 Replication Time Control (S3 RTC) unterstützt Sie bei der Erfüllung von Compliance- oder geschäftlichen Anforderungen für die Datenreplikation und bietet Einblick in die Amazon S3 Replikationszeiten. S3 RTC repliziert die meisten Objekte, die Sie zu Amazon S3 hochladen, in Sekunden und 99,99 Prozent dieser Objekte innerhalb von 15 Minuten.

Mit Replikationsmetriken können Sie die Gesamtzahl und Größe der Objekte, die zur Replikation ausstehen, sowie die maximale Replikationszeit in die Zielregion überwachen. Replikationsmetriken sind über die Amazon S3-Konsole und Amazon CloudWatch verfügbar. Weitere Informationen finden Sie unter the section called “Amazon S3 CloudWatch-Replikationsmetriken” (p. 743).

Mit S3 RTC benachrichtigen Amazon S3-Ereignisse Sie in dem seltenen Fall davon, dass Objekte nicht innerhalb von 15 Minuten repliziert werden und dass diese Objekte erfolgreich in ihre Zielregion repliziert werden. Amazon S3-Ereignisse sind über Amazon SQS, Amazon SNS oder AWS Lambda verfügbar. Weitere Informationen finden Sie unter Benachrichtigungen (p. 646).

Themen

Aktivieren der S3 Replication Time Control

Sie können mit dem Arbeiten mit S3 Replication Time Control (S3 RTC) mit einer neuen oder vorhandenen Replikationsregel beginnen. Sie können die Replikationsregel auf einen gesamten S3-Bucket oder auf Amazon S3-Objekte mit einem bestimmten Präfix oder Tag anwenden. Wenn Sie S3 RTC aktivieren, werden Replikationsmetriken auch für Ihre Replikationsregel aktiviert.

> **Note**
>
> Replikationsmetriken werden zum selben Tarif in Rechnung gestellt, wie die benutzerdefinierten Amazon CloudWatch-Metriken. Weitere Informationen hierzu finden Sie unter Amazon CloudWatch – Preise.

Sie können Replication Time Control über die Amazon S3-Konsole, die Amazon S3-API, die AWS-SDKs und die AWS-Befehlszeilenschnittstelle (AWS CLI) konfigurieren.

Weitere Informationen finden Sie unter Übersicht über die Replikationskonfiguration (p. 675).

Verwenden von Replikationsmetriken zur Überwachung von Amazon S3 Replikationskonfigurationen

Jede Replikationsregel, für die S3 Replication Time Control (S3 RTC) aktiviert ist, veröffentlicht Replikationsmetriken. Mit Replikationsmetriken können Sie die Gesamtzahl und Größe der Objekte, die zur Replikation ausstehen, sowie die maximale Replikationszeit in die Zielregion überwachen. Anschließend können Sie jedes Dataset, das Sie separat replizieren, überwachen.

Replikationsmetriken sind innerhalb von 15 Minuten nach der Aktivierung von S3 RTC verfügbar. Replikationsmetriken sind über die Amazon S3-Konsole, die Amazon S3-API, die AWS-SDKs, die AWS-Befehlszeilenschnittstelle (AWS CLI) und Amazon CloudWatch verfügbar. Weitere Informationen finden Sie unter the section called “Überwachung von Metriken mit CloudWatch” (p. 739).

Verwenden von Amazon S3-Ereignissen zum Nachverfolgen von S3 Replication Time Control-Objekten

Sie können die Replikationszeit für Objekte nachverfolgen, die nicht innerhalb von 15 Minuten repliziert wurden, indem Sie bestimmte Ereignisbenachrichtigungen überwachen, die von S3 Replication Time Control (S3 RTC) veröffentlicht werden. Diese Ereignisse werden veröffentlicht, wenn ein Objekt, das für die Replikation mit S3 RTC in Frage kam, nicht innerhalb von 15 Minuten repliziert wurde und wenn dieses Objekt in die Zielregion repliziert wird.

Replikationsereignisse sind innerhalb von 15 Minuten nach Aktivierung von S3 RTC verfügbar. Amazon S3-Ereignisse sind über Amazon SQS, Amazon SNS oder AWS Lambda verfügbar. Weitere Informationen finden Sie unter the section called “Ereignisbenachrichtigungstypen und -ziele” (p. 649).

Bewährte Methoden und Richtlinien für die Verwendung der S3 Replication Time Control

Befolgen Sie beim Replizieren von Daten in Amazon S3 mit S3 Replication Time Control (S3 RTC) Verwendung diese Best Practice-Richtlinien, um die Replikationsleistung für Ihre Workloads zu optimieren.

Themen

Leistungsrichtlinien für Amazon S3 Replikation and Anforderungsraten

Wenn Speicherinhalte zu Amazon S3 hochgeladen oder von dort abgerufen werden, können Ihre Anwendungen Tausende von Transaktionen pro Sekunde bei der Anforderungsleistung erhalten. Beispielsweise kann eine Anwendung mindestens 3.500 PUT/COPY/POST/DELETE- oder 5.500 GET/HEAD-Anforderungen pro Sekunde pro Präfix in einem S3-Bucket erreichen, einschließlich der Anforderungen, die die S3 Replication in Ihrem Namen vornimmt. Es gibt keine Einschränkungen für die Anzahl der Präfixe in einem Bucket. Sie können Ihre Lese- und Schreibleistung steigern, indem Sie Lesevorgänge parallelisieren. Wenn Sie beispielsweise 10 Präfixe in einem S3-Bucket für parallele Lesevorgänge einrichten, können Sie damit die Leseleistung auf 55 000 Leseanfragen pro Sekunde skalieren.

Amazon S3 skaliert automatisch als Reaktion auf anhaltende Anforderungsraten oberhalb dieser Richtlinien oder anhaltender Anforderungsraten übereinstimmend mit LIST-Anforderungen. Während Amazon S3 intern für die neue Anforderungsrate optimiert wird, erhalten Sie möglicherweise temporär HTTP 503-Anforderungsantworten, bis die Optimierung abgeschlossen ist. Dies kann mit steigenden Anforderungsraten pro Sekunde oder beim ersten Aktivieren von S3 RTC auftreten. Während dieser Zeiträume kann sich die Replikationslatenz erhöhen. Das S3 RTC Service Level Agreement (SLA) gilt nicht für Zeiträume, in denen Amazon S3-Leistungsrichtlinien für Anforderungen pro Sekunde überschritten werden.

Das S3 RTC SLA gilt auch nicht in Zeiträumen, in denen Ihre Replikationsdatenübertragungsrate das Standardlimit von 1 Gbit/s überschreitet. Wenn Sie erwarten, dass Ihre Replikationsübertragungsrate 1 Gbit/s überschreitet, können Sie AWS Support Center kontaktieren oder über Servicekontingente eine Erhöhung Ihres Limit anzufordern.

Schätzen der Replikationsanforderungsraten

Ihre Gesamtanforderungsrate einschließlich der Anforderungen, die die Amazon S3 Replication in Ihrem Namen vornimmt, sollte den Richtlinien für die Amazon S3-Anforderungsrate sowohl für die Replikationsquelle als auch für die Ziel-Buckets entsprechen. Für jedes replizierte Objekt führt Amazon S3 Replikation bis zu fünf GET/HEAD-Anforderungen und eine PUT-Anforderung an den Quell-Bucket sowie eine PUT-Anforderung an den Ziel-Bucket aus.

Wenn Sie beispielsweise erwarten, 100 Objekte pro Sekunde zu replizieren, führt die Amazon S3 Replikation möglicherweise weitere 100 PUT-Anforderungen in Ihrem Namen für insgesamt 200 PUTs pro Sekunde an den Quell-S3-Bucket aus. Die Amazon S3 Replikation kann auch bis zu 500 GET/HEAD-Anforderungen (5 GET/HEAD-Anforderungen für jedes replizierte Objekt) ausführen.

> Note
>
> Es entstehen Kosten für nur eine PUT-Anforderung pro repliziertem Objekt. Weitere Informationen finden Sie in den Preisinformationen unter Amazon S3 – Häufig gestellte Fragen zur Replikation.

Ergreifen von Maßnahmen beim Überschreiten der Limits für S3 RTC-Datenübertragungsraten

Wenn Sie erwarten, dass die Datenübertragungsrate der S3 Replication Time Control das Standardlimit von 1 Gbit/s überschreitet, wenden Sie sich an AWS Support Center oder verwenden Sie Servicekontingente, um eine Erhöhung Ihres Limit anzufordern.

Replikationsanforderungsraten für mit AWS KMS verschlüsselte Objekte

Wenn Sie Objekte replizieren, die mit serverseitiger Verschlüsselung (SSE-KMS) mit Amazon S3 Replikation verschlüsselt sind, gelten für die AWS Key Management Service (AWS KMS)-Anforderungen pro Sekunde Grenzwerte. AWS KMS lehnt möglicherweise eine ansonsten gültige Anforderung ab, da Ihre Anforderungsrate das Limit für die Anzahl der Anforderungen pro Sekunde überschreitet. Wird eine Anforderung abgelehnt, gibt AWS KMS einen Fehler des Typs `ThrottlingException` zurück. Das AWS KMS-Anforderungsratenlimit gilt für Anforderungen, die Sie direkt ausführen, und für Anforderungen, die von der Amazon S3 Replikation in Ihrem Auftrag durchgeführt werden.

Wenn Sie beispielsweise erwarten, 1.000 Objekte pro Sekunde zu replizieren, können Sie 2.000 Anforderungen von Ihrem AWS KMS-Anforderungsratenlimit subtrahieren. Die resultierende Anforderungsrate pro Sekunde steht für Ihre AWS KMS-Workloads mit Ausnahme der Replikation zur Verfügung. Sie können AWS KMS-Anforderungsmetriken in Amazon CloudWatch verwenden, um die AWS KMS-Gesamtanforderungsrate in Ihrem AWS-Konto zu überwachen.

Ergreifen von Maßnahmen, wenn Sie AWS KMS-Anforderungsratenlimits überschreiten

Wenn Sie glauben, dass Ihre AWS KMS-Anforderungsrate einschließlich Replikationsanforderungen, die von der Amazon S3 Replication in Ihrem Namen vorgenommen wurden, die AWS KMS Anforderungsratengrenzen für Ihr Konto überschreiten, können Sie mithilfe von Servicekontingenten eine Erhöhung des Limits beantragen. Wenn Servicekontingente für AWS KMS in der AWS-Region nicht verfügbar sind, wenden Sie sich an AWS Support Center.

Ändern des Replikateigentümers

Bei der Replikation besitzt der Eigentümer des Quellobjekts standardmäßig auch das Replikat. Wenn sich der Quell- und der Ziel-Bucket im Besitz verschiedener AWS-Konten befinden, können Sie optionale Konfigurationseinstellungen hinzufügen, um die Replikateigentümerschaft zu dem AWS-Konto zu ändern, das den Ziel-Bucket besitzt. Sie können dies z. B. tun, um den Zugriff auf Objektreplikate einzuschränken. Dies wird auch als die Eigentümer-Überschreibungs-Option der Replikationskonfiguration bezeichnet. In diesem Abschnitt werden nur die relevanten zusätzlichen Konfigurationseinstellungen erläutert. Weitere Informationen zum Einrichten der Replikationskonfiguration finden Sie unter Replikation (p. 669).

Um die Eigentümer-Überschreibung zu konfigurieren, gehen Sie wie folgt vor:

- Fügen Sie die Eigentümer-Überschreibungs-Option zur Replikationskonfiguration hinzu, um Amazon S3 anzuweisen, die Replikateigentümerschaft zu ändern.
- Erteilen Sie Amazon S3 die Berechtigungen, die Replikateigentümerschaft zu ändern.
- Fügen Sie die Berechtigung zur Richtlinie für den Ziel-Bucket hinzu, um das Ändern der Replikateigentümerschaft zuzulassen. Damit kann der Eigentümer des Ziel-Buckets die Eigentümerschaft von Objektreplikaten annehmen.

In den folgenden Abschnitten wird beschrieben, wie Sie diese Aufgaben durchführen: Ein funktionierendes Beispiel mit Schritt-für-Schritt-Anleitungen finden Sie unter Beispiel 3: Ändern des Replikateigentümers, wenn sich Quell- und Ziel-Bucket im Besitz unterschiedlicher Konten befinden (p. 709).

Hinzufügen der Eigentümer-Überschreibungs-Option zur Replikationskonfiguration

Warning

Fügen Sie die Eigentümer-Überschreibungs-Option nur dann hinzu, wenn sich Quell- und Ziel-Buckets im Besitz von unterschiedlichen AWS-Konten befinden. Amazon S3 prüft nicht, ob die Buckets im Besitz von gleichen oder unterschiedlichen Konten sind. Falls Sie die Eigentümer-Überschreibung hinzufügen, wenn sich beide Buckets im Besitz desselben AWS-Kontos befinden, wendet Amazon S3 die Eigentümer-Überschreibung an. Sie gewährt dem Ziel-Bucket-Eigentümer vollständige Berechtigungen und repliziert keine nachfolgenden Aktualisierungen der Quellobjekt-Zugriffskontrollliste (ACL). Der Replikateigentümer kann die ACL, die mit einem Replikat mit einer `PUT ACL`-Anforderung verknüpft ist, direkt ändern, aber nicht über eine Replikation.

Um die Eigentümer-Überschreibungs-Option festzulegen, fügen Sie Folgendes zum `Destination`-Element hinzu.

- Das Element `AccessControlTranslation`, das Amazon S3 anweist, die Replikateigentümerschaft zu ändern
- Das Element `Account`, das das AWS-Konto des Ziel-Bucket-Eigentümers angibt

```
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    ...
    <Destination>
      ...
      <AccessControlTranslation>
           <Owner>Destination</Owner>
       </AccessControlTranslation>
      <Account>destination-bucket-owner-account-id</Account>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Die folgende Beispiel-Replikationskonfiguration weist Amazon S3 an, Objekte mit dem Schlüsselpräfix `Tax` in den Ziel-Bucket zu replizieren und die Replikateigentümerschaft zu ändern.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Role>arn:aws:iam::account-id:role/role-name</Role>
   <Rule>
      <ID>Rule-1</ID>
      <Priority>1</Priority>
      <Status>Enabled</Status>
      <Status>Enabled</Status>
      <DeleteMarkerReplication>
         <Status>Disabled</Status>
      </DeleteMarkerReplication>
      <Filter>
         <Prefix>Tax</Prefix>
      </Filter>
      <Destination>
         <Bucket>arn:aws:s3:::destination-bucket</Bucket>
         <Account>destination-bucket-owner-account-id</Account>
         <AccessControlTranslation>
            <Owner>Destination</Owner>
         </AccessControlTranslation>
      </Destination>
   </Rule>
```

```
</ReplicationConfiguration>
```

Erteilen der Berechtigung zum Ändern der Replikateigentümerschaft für Amazon S3

Erteilen Sie Amazon S3 die Berechtigungen zum Ändern der Replikateigentümerschaft, indem Sie die Berechtigung für die Aktion `s3:ObjectOwnerOverrideToBucketOwner` zur Berechtigungsrichtlinie hinzufügen, die mit der IAM-Rolle verknüpft ist. Dies ist die IAM-Rolle, die Sie in der Replikationskonfiguration festgelegt haben und die es Amazon S3 gestattet, Objekte in Ihrem Namen anzunehmen und zu replizieren.

```
...
{
    "Effect":"Allow",
         "Action":[
       "s3:ObjectOwnerOverrideToBucketOwner"
    ],
    "Resource":"arn:aws:s3:::destination-bucket/*"
}
...
```

Hinzufügen der Berechtigung zur Ziel-Bucket-Richtlinie, um das Ändern der Replikateigentümerschaft zuzulassen

Der Eigentümer des Ziel-Buckets muss dem Eigentümer des Quell-Buckets die Berechtigung zum Ändern der Replikateigentümerschaft erteilen. Der Eigentümer des Ziel-Buckets erteilt dem Eigentümer des Quell-Buckets die Berechtigung für die Aktion `s3:ObjectOwnerOverrideToBucketOwner`. Dies ermöglicht dem Eigentümer des Ziel-Buckets, die Eigentümerschaft von Objektreplikaten anzunehmen. Die folgende Beispielanweisung einer Bucket-Richtlinie zeigt, wie dies funktioniert:

```
...
{
    "Sid":"1",
    "Effect":"Allow",
    "Principal":{"AWS":"source-bucket-account-id"},
    "Action":["s3:ObjectOwnerOverrideToBucketOwner"],
    "Resource":"arn:aws:s3:::destination-bucket/*"
}
...
```

Weitere Überlegungen

Wenn Sie die Eigentümer-Überschreibungs-Option konfigurieren, berücksichtigen Sie die folgenden Überlegungen:

- Standardmäßig besitzt der Eigentümer des Quellobjekts auch das Replikat. Amazon S3 repliziert die Objektversion und die damit verknüpfte ACL.

  Wenn Sie die Eigentümer-Überschreibung hinzufügen, repliziert Amazon S3 nur die Objektversion, nicht die ACL. Darüber hinaus repliziert Amazon S3 keine nachfolgenden Änderungen an der ACL des Quellobjekts. Amazon S3 legt die ACL für das Replikat fest, die dem Ziel-Bucket-Eigentümer vollständige Kontrolle gewährt.

- Wenn Sie eine Replikationskonfiguration ändern und die Eigentümerüberschreibung aktivieren oder deaktivieren, geschieht Folgendes:

  - Wenn Sie die Eigentümerüberschreibungs-Option zur Replikationskonfiguration hinzufügen

    Wenn Amazon S3 eine Objektversion repliziert, verwirft es die ACL, die mit dem Quellobjekt verknüpft ist. Es legt stattdessen die ACL für das Replikat fest, sodass der Ziel-Bucket-Eigentümer vollständige Kontrolle erhält. Es repliziert keine nachfolgenden Änderungen an der Quellobjekt-ACL. Diese Änderung der ACL gilt nicht für Objektversionen, die repliziert wurden, bevor Sie die Eigentümer-Überschreibungs-Option festgelegt haben. ACL-Aktualisierungen an den Quellobjekten, die repliziert wurden, bevor die Eigentümer-Überschreibungs-Option festgelegt wurde, werden weiterhin repliziert (da das Objekt und seine Replikate weiterhin denselben Eigentümer haben).

  - Wenn Sie die Eigentümer-Überschreibungs-Option aus der Replikationskonfiguration entfernen

    Amazon S3 repliziert neue Objekte, die im Quell-Bucket erscheinen, und die zugehörigen ACLs in den Ziel-Bucket. Bei Objekten, die repliziert wurden, bevor Sie die Eigentümer-Überschreibung entfernt haben, repliziert Amazon S3 die ACLs nicht, da die Änderung der Objekteigentümerschaft, die Amazon S3 vorgenommen hat, gültig bleibt. Das bedeutet, dass in der Objektversion abgelegte ACLs, die repliziert wurden, als Sie die Eigentümer-Überschreibungs-Option festgelegt hatten, weiterhin nicht repliziert werden.

Replizieren von Objekten, die mit der serverseitigen Verschlüsselung (SSE) unter Verwendung von in AWS KMS gespeicherten Verschlüsselungsschlüsseln erstellt wurden

Standardmäßig repliziert Amazon S3 keine Objekte, die im Ruhezustand gespeichert sind, mit der serverseitigen Verschlüsselung mit in AWS KMS gespeicherten Kundenhauptschlüsseln (CMKs). Dieser Abschnitt beschreibt zusätzliche Konfigurationen, die Sie nutzen können, um Amazon S3 anzuweisen, diese Objekte zu replizieren.

> **Important**
>
> Die Replikation verschlüsselter Daten ist ein serverseitiger Prozess, der vollständig innerhalb von Amazon S3 abläuft. Die Replikation unterstützt keine clientseitige Verschlüsselung.

Ein Beispiel mit Schritt-für-Schritt-Anleitungen finden Sie unter Beispiel 4: Replizieren verschlüsselter Objekte (p. 713). Weitere Informationen zum Erstellen einer Replikationskonfiguration finden Sie unter Replikation (p. 669).

Themen

Angeben zusätzlicher Informationen in der Replikationskonfiguration

In der Replikationskonfiguration machen Sie Folgendes:

- Fügen Sie der `Destination`-Konfiguration den symmetrischen, vom Kunden verwalteten AWS KMS-CMK hinzu, mit dem Amazon S3 Objektreplikate verschlüsseln soll.
- Aktivieren Sie ausdrücklich die Replikation von Objekten, die mit den AWS KMS-CMKs verschlüsselt wurden, indem Sie das `SourceSelectionCriteria`-Element hinzufügen.

```
<ReplicationConfiguration>
   <Rule>
      ...
      <SourceSelectionCriteria>
         <SseKmsEncryptedObjects>
           <Status>Enabled</Status>
         </SseKmsEncryptedObjects>
      </SourceSelectionCriteria>

      <Destination>
          ...
          <EncryptionConfiguration>
             <ReplicaKmsKeyID>AWS KMS key ID for the AWS region of the destination
 bucket.</ReplicaKmsKeyID>
          </EncryptionConfiguration>
       </Destination>
      ...
   </Rule>
</ReplicationConfiguration>
```

> **Important**
>
> Der AWS KMS-CMK muss in derselben AWS-Region erstellt worden sein wie der Ziel-Bucket. Der AWS KMS-CMK muss gültig sein. Die `PUT`-Bucket Replikations-API überprüft nicht die Gültigkeit von AWS KMS-CMKs. Wenn Sie einen ungültigen CMK verwenden, erhalten Sie als Antwort den "200 OK"-Statuscode, aber die Replikation schlägt fehl.

Das folgende Beispiel zeigt eine Replikationskonfiguration, die optionale Konfigurationselemente enthält.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration>
   <Role>arn:aws:iam::account-id:role/role-name</Role>
   <Rule>
      <ID>Rule-1</ID>
      <Priority>1</Priority>
      <Status>Enabled</Status>
      <DeleteMarkerReplication>
         <Status>Disabled</Status>
      </DeleteMarkerReplication>
      <Filter>
         <Prefix>Tax</Prefix>
      </Filter>
      <Destination>
         <Bucket>arn:aws:s3:::destination-bucket</Bucket>
         <EncryptionConfiguration>
            <ReplicaKmsKeyID>The AWS KMS key ID for the AWS region of the destination
 bucket (S3 uses it to encrypt object replicas).</ReplicaKmsKeyID>
         </EncryptionConfiguration>
      </Destination>
      <SourceSelectionCriteria>
```

```
        <SseKmsEncryptedObjects>
           <Status>Enabled</Status>
        </SseKmsEncryptedObjects>
     </SourceSelectionCriteria>
   </Rule>
</ReplicationConfiguration>
```

Diese Replikationskonfiguration weist eine Regel auf. Die Regel gilt für Objekte mit dem Schlüsselpräfix `Tax`. Amazon S3 nutzt die AWS KMS-Schlüssel-ID, um diese Objektreplikate zu verschlüsseln.

Erteilen zusätzlicher Berechtigungen für die IAM-Rolle

Um Objekte zu replizieren, die im Ruhezustand unter AWS Key Management Service (AWS KMS) verschlüsselt sind, vergeben Sie die folgenden zusätzlichen Berechtigungen für die IAM-Rolle, die Sie in der Replikationskonfiguration angeben. Sie erteilen diese Berechtigungen, indem Sie die mit der IAM-Rolle verknüpfte Berechtigungsrichtlinie aktualisieren. Objekte, die mit der serverseitige Verschlüsselung unter Verwendung der vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) erstellt wurden, werden nicht repliziert.

- Berechtigung für die Aktion `s3:GetObjectVersionForReplication` für Quellobjekte. Die Berechtigung für diese Aktion ermöglicht es Amazon S3, sowohl unverschlüsselte Objekte als auch Objekte, die mit serverseitiger Verschlüsselung erstellt wurden, mit von Amazon S3 verwalteten Schlüsseln (SSE-S3) oder CMKs zu replizieren, die in AWS KMS gespeichert sind (SSE-KMS).

  Note

  Wir empfehlen, dass Sie die Aktion `s3:GetObjectVersionForReplication` statt der Aktion `s3:GetObjectVersion` verwenden, da sie Amazon S3 nur die minimalen Berechtigungen bereitstellt, die für eine Replikation nötig sind. Darüber hinaus erlaubt die Berechtigung für die Aktion `s3:GetObjectVersion` die Replikation von unverschlüsselten und über SSE-S3 verschlüsselten Objekten, nicht aber von Objekten, die mit einem in AWS KMS gespeicherten CMK erstellt wurden.
- Berechtigungen für die folgenden AWS KMS-Aktionen:
  - `kms:Decrypt`-Berechtigungen für den AWS KMS-CMK, der zur Verschlüsselung des Quellobjekts verwendet wurde.
  - `kms:Encrypt`-Berechtigungen für den AWS KMS-CMK, der zur Verschlüsselung des replizierten Objekts verwendet wurde.

  Wir empfehlen, dass Sie diese Berechtigungen mithilfe der AWS KMS-Bedingungsschlüssel auf bestimmte Buckets und Objekte beschränken. Dies ist in den folgenden Beispielen für Richtlinienanweisungen dargestellt:

```
{
    "Action": ["kms:Decrypt"],
    "Effect": "Allow",
    "Condition": {
        "StringLike": {
            "kms:ViaService": "s3.source-bucket-region.amazonaws.com",
            "kms:EncryptionContext:aws:s3:arn": [
                "arn:aws:s3:::source-bucket-name/key-prefix1*",
            ]
        }
    },
    "Resource": [
        "List of AWS KMS key IDs used to encrypt source objects.",
    ]
},
{
    "Action": ["kms:Encrypt"],
    "Effect": "Allow",
```

```
    "Condition": {
        "StringLike": {
            "kms:ViaService": "s3.destination-bucket-region.amazonaws.com",
            "kms:EncryptionContext:aws:s3:arn": [
                "arn:aws:s3:::destination-bucket-name/key-prefix1*",
            ]
        }
    },
    "Resource": [
         "AWS KMS key IDs (for the AWS region of the destination bucket). S3 uses it to
 encrypt object replicas",
    ]
}
```

Das AWS-Konto, dem die IAM-Rolle gehört, muss über Berechtigungen für die AWS KMS-Aktionen (`kms:Encrypt` und `kms:Decrypt`) für die in der Richtlinie aufgeführten AWS KMS-CMKs verfügen. Wenn die AWS KMS-CMKs einem anderen AWS-Konto gehören, muss der CMK-Besitzer diese Berechtigungen dem AWS-Konto gewähren, dem die Rolle IAM gehört. Weitere Informationen zur Verwaltung des Zugriffs auf diese CMKs finden Sie unter Verwendung von IAM-Richtlinien mit AWS-KMS in der AWS Key Management Service Developer Guide.

Im Folgenden finden Sie eine vollständige IAM-Richtlinie, die die erforderlichen Berechtigungen zum Replizieren von unverschlüsselten Objekten, Objekten, die mit der serverseitigen Verschlüsselung unter Verwendung von Amazon S3 verwalteten Verschlüsselungsschlüsseln erstellt wurden, und CMKs, die in AWS KMS gespeichert sind enthält.

> **Note**
>
> Objekte, die mit der serverseitige Verschlüsselung unter Verwendung der vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C) erstellt wurden, werden nicht repliziert.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket/key-prefix1*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete"
         ],
         "Resource":"arn:aws:s3:::destination-bucket/key-prefix1*"
      },
```

```
      {
         "Action":[
            "kms:Decrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.source-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::source-bucket-name/key-prefix1*"
               ]
            }
         },
         "Resource":[
           "List of AWS KMS key IDs used to encrypt source objects."
         ]
      },
      {
         "Action":[
            "kms:Encrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.destination-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::destination-bucket-name/prefix1*"
               ]
            }
         },
         "Resource":[
            "AWS KMS key IDs (for the AWS region of the destination bucket) to use for
 encrypting object replicas"
         ]
      }
   ]
}
```

Erteilen von zusätzlichen Berechtigungen für kontenübergreifende Szenarien

In einem kontoübergreifenden Szenario, in dem die *Quelle*- und *Ziel*-Buckets verschiedenen AWS-Konten gehören, können Sie einen kundenverwalteten CMK zum Verschlüsseln von Objektreplikaten verwenden. Der CMK-Besitzer muss dem Besitzer des Quell-Buckets jedoch die Berechtigung erteilen, den CMK zu verwenden.

So erteilen Sie dem Besitzer des Quell-Buckets die Berechtigung zur Verwendung des AWS KMS-CMK (IAM-Konsole):

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die AWS Key Management Service(AWS KMS)-Konsole unter https://console.aws.amazon.com/kms.
2. Um die AWS-Region zu ändern, verwenden Sie die Regionenauswahl in der oberen rechten Ecke der Seite.
3. Zum Anzeigen der Schlüssel in Ihrem Konto, die Sie erstellen und verwalten, wählen Sie im Navigationsbereich Customer managed keys (Vom Kunden verwaltete Schlüssel) aus.
4. Wählen Sie den CMK aus.
5. Wählen Sie unter General configuration (Allgemeine Konfiguration) die Registerkarte Schlüsselrichtlinie aus.
6. Wählen Sie Other AWS Accounts (Weitere AWS-Konten) aus.

7. Wählen Sie Add another AWS Account (Ein weiteres AWS-Konto hinzufügen) aus.
8. Geben Sie unter arn:aws:iam:: die ID des Quell-Buckets ein.
9. Wählen Sie Save Changes.

So erteilen Sie dem Besitzer des Quell-Buckets die Berechtigung zur Verwendung des AWS KMS-CMKs (AWS-CLI):

- Weitere Informationen finden sie unter put-key-policy in der AWS CLI-Befehlsreferenz. Weitere Informationen über die zugrunde liegende API finden Sie unter PutKeyPolicy in der AWS Key Management Service API Reference.

Überlegungen zu Transaktionslimits bei AWS KMS

Wenn Sie nach der CRR-Aktivierung zu viele neue Objekte mit AWS KMS-Verschlüsselung hinzufügen, kann es zu einer Drosselung kommen (Fehler: HTTP 503 Slow Down). Die Drosselung erfolgt, wenn die Anzahl an AWS KMS-Transaktionen pro Sekunde das aktuelle Limit überschreitet. Weitere Informationen finden Sie unter Limits im AWS Key Management Service Developer Guide.

Um eine Erhöhung des Limits zu beantragen, verwenden Sie Servicekontingente. Weitere Informationen finden Sie unter Amazon Web Services-Limits. Wenn Servicekontingente in Ihrer Region nicht unterstützt wird, öffnen Sie einen AWS Support-Fall.

Replikations-Walkthroughs

Das folgende Beispiel zeigt das Konfigurieren der Replikation für häufige Anwendungsfälle. Die Beispiele zeigen die Replikationskonfiguration mit der Amazon S3-Konsole, der AWS Command Line Interface (AWS CLI) und den AWS SDKs (gezeigt werden Java- and .NET-SDK-Beispiele). Weitere Informationen zum Installieren und Konfigurieren der AWS CLI finden Sie in den folgenden Themen im Benutzerhandbuch für AWS Command Line Interface.

- Installieren der AWS-Befehlszeilenschnittstelle
- Konfigurieren der AWS-CLI - Sie müssen mindestens ein Profil einrichten. Richten Sie bei kontoübergreifenden Szenarien zwei Profile ein.

Informationen zu AWS-SDKs finden Sie unter AWS SDK for Java und AWS SDK for .NET.

Themen

Beispiel 1: Einrichten der Replikation, wenn die Quell- und Ziel-Buckets demselben Konto gehören.

In diesem Beispiel richten Sie eine Replikation für Quell- und Ziel-Buckets ein, die demselben AWS-Konto gehören. Es sind Beispiele für die Amazon S3-Konsole, die AWS Command Line Interface (AWS CLI), AWS SDK for Java und AWS SDK für .NET vorhanden.

Konfigurieren der Replikation, wenn die Buckets demselben Konto angehören (Konsole)

Detaillierte Anleitungen finden Sie im Thema über Wie füge ich eine Replikationsregel zu einem S3-Bucket hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service. Dieses Thema enthält Anleitungen für die Replikationskonfiguration, wenn sich die Buckets im Besitz derselben und unterschiedlicher AWS-Konten befinden.

Konfigurieren der Replikation, wenn die Buckets demselben Konto angehören (AWS-CLI)

Um die Replikation einzurichten, wenn Quell- und Ziel-Bucket demselben AWS-Konto gehören, erstellen Sie mit der AWS CLI Quell- und Ziel-Buckets, aktivieren das Versioning für die Buckets, erstellen eine IAM-Rolle, die Amazon S3 die Berechtigung zum Replizieren von Objekten gewährt, und fügen dem Quell-Bucket die Replikationskonfiguration hinzu. Um die Einrichtung zu prüfen, testen Sie sie.

Einrichten der Replikation, wenn sich Quell- und Ziel-Bucket im Besitz desselben AWS-Kontos befinden

1. Richten Sie das Anmeldeinformationsprofil für die AWS CLI ein. In diesem Beispiel verwenden wir den Profilnamen `acctA`. Weitere Informationen zum Einrichten der Anmeldeinformationsprofile finden Sie unter Benannte Profile im Benutzerhandbuch für AWS Command Line Interface.

   Important

   Das Profil, das Sie für diese Übung verwenden, muss über die nötigen Berechtigungen verfügen. Beispielsweise legen Sie in der Replikationskonfiguration die IAM-Rolle fest, die Amazon S3 annehmen kann. Dies können Sie nur tun, wenn das verwendete Profil über die `iam:PassRole`-Berechtigung verfügt. Weitere Informationen finden Sie unter Gewähren von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch. Wenn Sie zur Erstellung des benannten Profils die Benutzeranmeldeinformationen eines Administrators verwenden, können Sie alle Aufgaben durchführen.

2. Erstellen Sie einen *`Quell`*-Bucket und aktivieren Sie das Versioning für ihn. Der folgende Code erstellt einen *`Quell`*-Bucket in der Region USA Ost (Nord-Virginia) (us-east-1).

```
aws s3api create-bucket \
--bucket source \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket source \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Erstellen Sie einen *`Ziel`*-Bucket und aktivieren Sie das Versioning für ihn. Der folgende Code erstellt einen *`Ziel`*-Bucket in der Region USA West (Oregon) (us-west-2).

Note

Um die Replikationskonfiguration einzurichten, wenn sich Quell- und Ziel-Bucket im selben AWS-Konto befinden, verwenden Sie dasselbe Profil. Dieses Beispiel verwendet `acctA`. Um die Replikationskonfiguration zu testen, wenn sich die Buckets im Besitz unterschiedlicher AWS-Konten befinden, legen Sie verschiedene Profile für die Buckets fest. In diesem Beispiel verwenden wir das Profil `acctB` für den Ziel-Bucket.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctA
```

4. Erstellen Sie eine IAM-Rolle. Sie legen diese Rolle in der Replikationskonfiguration fest, die Sie später zum *Quell*-Bucket hinzufügen. Amazon S3 nimmt diese Rolle an, um in Ihrem Namen Objekte zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   - Erstellen Sie eine Rolle.
   - Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

   a. Erstellen Sie die IAM-Rolle.

      i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `S3-role-trust-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie gewährt Amazon S3 Service-Prinzipal-Berechtigungen, um die Rolle anzunehmen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

      ii. Führen Sie den folgenden Befehl aus, um eine Rolle zu erstellen.

```
$ aws iam create-role \
--role-name replicationRole \
--assume-role-policy-document file://s3-role-trust-policy.json  \
--profile acctA
```

   b. Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

      i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `S3-role-permissions-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen

Computer. Diese Zugriffsrichtlinie erteilt Berechtigungen für verschiedene Amazon S3-Bucket- und -Objektaktionen.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags",
            "s3:GetObjectVersionTagging"

         ],
         "Resource":"arn:aws:s3:::destination-bucket/*"
      }
   ]
}
```

ii. Führen Sie den folgenden Befehl aus, um eine Richtlinie zu erstellen und sie der Rolle anzufügen.

```
$ aws iam put-role-policy \
--role-name replicationRole \
--policy-document file://s3-role-permissions-policy.json \
--policy-name replicationRolePolicy \
--profile acctA
```

5. Fügen Sie eine Replikationskonfiguration zum *Quell*-Bucket hinzu.

   a. Zwar fordert die Amazon S3-API die Replikationskonfiguration als XML, aber die AWS CLI verlangt, dass Sie die Replikationskonfiguration als JSON angeben. Speichern Sie den folgenden JSON-Code in einer Datei mit dem Namen `replication.json` im lokalen Verzeichnis auf Ihrem Computer.

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
```

```
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket": "arn:aws:s3:::destination-bucket"
      }
    }
  ]
}
```

b. Aktualisieren Sie den JSON-Code, indem Sie Werte für *destination-bucket* und *IAM-role-ARN* angeben. Speichern Sie die Änderungen.

c. Führen Sie den folgenden Befehl aus, um die Replikationskonfiguration zu Ihrem Quell-Bucket hinzuzufügen. Stellen Sie sicher, dass Sie den Namen für den *Quell*-Bucket angeben.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket source \
--profile acctA
```

Um die Replikationskonfiguration abzurufen, verwenden Sie den Befehl `get-bucket-replication`.

```
$ aws s3api get-bucket-replication \
--bucket source \
--profile acctA
```

6. Testen Sie die Einrichtung in der Amazon S3-Konsole:

a. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3.

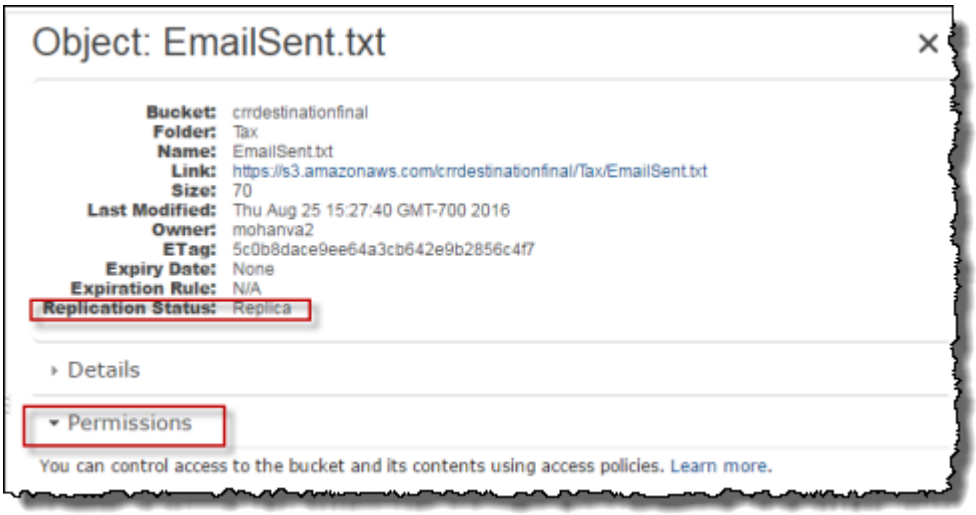
b. Erstellen Sie im *Quell*-Bucket einen Ordner mit dem Namen `Tax`.

c. Fügen Sie Beispielobjekte zum `Tax`-Ordner im *Quell*-Bucket hinzu.

> **Note**
>
> Die von Amazon S3 für die Replikation eines Objekts benötigte Zeit hängt von der Größe des Objekts ab. Weitere Informationen zum Anzeigen des Replikationsstatus finden Sie unter Informationen zum Replikationsstatus (p. 721).

Überprüfen Sie im *Ziel*-Bucket Folgendes:

- Dass Amazon S3 die Objekte repliziert hat.
- In den properties (Eigenschaften) des Objekts, dass der Replication Status (Replikationsstatus) auf `Replica` festgelegt ist (was es als Replikatobjekt kennzeichnet).
- In den properties (Eigenschaften) des Objekts, dass im Berechtigungsabschnitt keine Berechtigungen aufgeführt sind. Dies bedeutet, dass sich das Replikat noch im Besitz des Eigentümers des *Quell*-Buckets befindet und der Eigentümer des *Ziel*-Ordners über keine Berechtigung auf dem Objektreplikat verfügt. Sie können eine optionale Konfiguration hinzufügen, um Amazon S3 anzuweisen, die Replikateigentümerschaft zu ändern. Ein Beispiel finden Sie unter Beispiel 3: Ändern des Replikateigentümers, wenn sich Quell- und Ziel-Bucket im Besitz unterschiedlicher Konten befinden (p. 709).



d. Aktualisieren Sie die ACL eines Objekts im *Quell*-Bucket und stellen Sie sicher, dass die Änderungen im *Ziel*-Bucket angezeigt werden.

   Anleitungen finden Sie unter Wie lege ich Berechtigungen für ein Objekt fest? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Konfigurieren der Replikation, wenn die Buckets demselben Konto angehören (AWS-SDK)

Verwenden Sie die folgenden Code-Beispiele, um eine Replikationskonfiguration mit AWS SDK for Java bzw. AWS SDK für .NET zu einem Bucket hinzuzufügen.

Java

Das folgende Beispiel fügt eine Replikationskonfiguration einem Bucket hinzu und ruft die Konfiguration anschließend ab und überprüft sie. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Testen der Amazon S3-Java-Codebeispiele (p. 810).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.identitymanagement.AmazonIdentityManagement;
import com.amazonaws.services.identitymanagement.AmazonIdentityManagementClientBuilder;
import com.amazonaws.services.identitymanagement.model.CreateRoleRequest;
import com.amazonaws.services.identitymanagement.model.PutRolePolicyRequest;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.BucketReplicationConfiguration;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.CreateBucketRequest;
import com.amazonaws.services.s3.model.DeleteMarkerReplication;
import com.amazonaws.services.s3.model.DeleteMarkerReplicationStatus;
import com.amazonaws.services.s3.model.ReplicationDestinationConfig;
import com.amazonaws.services.s3.model.ReplicationRule;
import com.amazonaws.services.s3.model.ReplicationRuleStatus;
import com.amazonaws.services.s3.model.SetBucketVersioningConfigurationRequest;
import com.amazonaws.services.s3.model.StorageClass;
import com.amazonaws.services.s3.model.replication.ReplicationFilter;
import com.amazonaws.services.s3.model.replication.ReplicationFilterPredicate;
```

```
import com.amazonaws.services.s3.model.replication.ReplicationPrefixPredicate;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class CrossRegionReplication {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String accountId = "*** Account ID ***";
        String roleName = "*** Role name ***";
        String sourceBucketName = "*** Source bucket name ***";
        String destBucketName = "*** Destination bucket name ***";
        String prefix = "Tax/";

        String roleARN = String.format("arn:aws:iam::%s:role/%s", accountId, roleName);
        String destinationBucketARN = "arn:aws:s3:::" + destBucketName;

        AmazonS3 s3Client = AmazonS3Client.builder()
            .withCredentials(new ProfileCredentialsProvider())
            .withRegion(clientRegion)
            .build();

        createBucket(s3Client, clientRegion, sourceBucketName);
        createBucket(s3Client, clientRegion, destBucketName);
        assignRole(roleName, clientRegion, sourceBucketName, destBucketName);


        try {


            // Create the replication rule.
            List<ReplicationFilterPredicate> andOperands = new
 ArrayList<ReplicationFilterPredicate>();
            andOperands.add(new ReplicationPrefixPredicate(prefix));


            Map<String, ReplicationRule> replicationRules = new HashMap<String,
 ReplicationRule>();
            replicationRules.put("ReplicationRule1",
                new ReplicationRule()
                    .withPriority(0)
                    .withStatus(ReplicationRuleStatus.Enabled)
                    .withDeleteMarkerReplication(new
 DeleteMarkerReplication().withStatus(DeleteMarkerReplicationStatus.DISABLED))
                    .withFilter(new ReplicationFilter().withPredicate(new
 ReplicationPrefixPredicate(prefix)))
                    .withDestinationConfig(new ReplicationDestinationConfig()
                        .withBucketARN(destinationBucketARN)
                        .withStorageClass(StorageClass.Standard)));

            // Save the replication rule to the source bucket.
            s3Client.setBucketReplicationConfiguration(sourceBucketName,
                new BucketReplicationConfiguration()
                    .withRoleARN(roleARN)
                    .withRules(replicationRules));

            // Retrieve the replication configuration and verify that the configuration
            // matches the rule we just set.
            BucketReplicationConfiguration replicationConfig =
 s3Client.getBucketReplicationConfiguration(sourceBucketName);
            ReplicationRule rule = replicationConfig.getRule("ReplicationRule1");
```

```
            System.out.println("Retrieved destination bucket ARN: " +
 rule.getDestinationConfig().getBucketARN());
            System.out.println("Retrieved priority: " + rule.getPriority());
            System.out.println("Retrieved source-bucket replication rule status: " +
 rule.getStatus());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void createBucket(AmazonS3 s3Client, Regions region, String
 bucketName) {
        CreateBucketRequest request = new CreateBucketRequest(bucketName,
 region.getName());
        s3Client.createBucket(request);
        BucketVersioningConfiguration configuration = new
 BucketVersioningConfiguration().withStatus(BucketVersioningConfiguration.ENABLED);

        SetBucketVersioningConfigurationRequest enableVersioningRequest = new
 SetBucketVersioningConfigurationRequest(bucketName, configuration);
        s3Client.setBucketVersioningConfiguration(enableVersioningRequest);


    }

    private static void assignRole(String roleName, Regions region, String
 sourceBucket, String destinationBucket) {
        AmazonIdentityManagement iamClient =
 AmazonIdentityManagementClientBuilder.standard()
            .withRegion(region)
            .withCredentials(new ProfileCredentialsProvider())
            .build();
        StringBuilder trustPolicy = new StringBuilder();
        trustPolicy.append("{\\r\\n   ");
        trustPolicy.append("\\\"Version\\\":\\\"2012-10-17\\\",\\r\\n   ");
        trustPolicy.append("\\\"Statement\\\":[\\r\\n      {\\r\\n         ");
        trustPolicy.append("\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\\"Principal\\
\":{\\r\\n            ");
        trustPolicy.append("\\\"Service\\\":\\\"s3.amazonaws.com\\\"\\r\\n         },\
\r\\n         ");
        trustPolicy.append("\\\"Action\\\":\\\"sts:AssumeRole\\\"\\r\\n      }\\r\\n
  ]\\r\\n}");

        CreateRoleRequest createRoleRequest = new CreateRoleRequest()
            .withRoleName(roleName)
            .withAssumeRolePolicyDocument(trustPolicy.toString());

        iamClient.createRole(createRoleRequest);

        StringBuilder permissionPolicy = new StringBuilder();
        permissionPolicy.append("{\\r\\n   \\\"Version\\\":\\\"2012-10-17\\\",\\r\\n
 \\\"Statement\\\":[\\r\\n      {\\r\\n         ");
        permissionPolicy.append("\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\\"Action
\\\":[\\r\\n            ");
        permissionPolicy.append("\\\"s3:GetObjectVersionForReplication\\\",\\r\\n
      ");
        permissionPolicy.append("\\\"s3:GetObjectVersionAcl\\\"\\r\\n         ],\\r\\n
        \\\"Resource\\\":[\\r\\n            ");
        permissionPolicy.append("\\\"arn:aws:s3:::");
        permissionPolicy.append(sourceBucket);
```

```
        permissionPolicy.append("/*\\\"\\r\\n         ]\\r\\n      },\\r\\n      {\\r\
\n         ");
        permissionPolicy.append("\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\\"Action
\\\":[\\r\\n            ");
        permissionPolicy.append("\\\"s3:ListBucket\\\",\\r\\n            \\
\"s3:GetReplicationConfiguration\\\"\\r\\n         ");
        permissionPolicy.append("],\\r\\n         \\\"Resource\\\":[\\r\\n            \
\\"arn:aws:s3:::");
        permissionPolicy.append(sourceBucket);
        permissionPolicy.append("\\r\\n         ");
        permissionPolicy.append("]\\r\\n      },\\r\\n      {\\r\\n         \\\"Effect\
\\":\\\"Allow\\\",\\r\\n         ");
        permissionPolicy.append("\\\"Action\\\":[\\r\\n            \\
\"s3:ReplicateObject\\\",\\r\\n            ");
        permissionPolicy.append("\\\"s3:ReplicateDelete\\\",\\r\\n            \\
\"s3:ReplicateTags\\\",\\r\\n            ");
        permissionPolicy.append("\\\"s3:GetObjectVersionTagging\\\"\\r\\n\\r\\n
  ],\\r\\n         ");
        permissionPolicy.append("\\\"Resource\\\":\\\"arn:aws:s3:::");
        permissionPolicy.append(destinationBucket);
        permissionPolicy.append("/*\\\"\\r\\n      }\\r\\n   ]\\r\\n}");

        PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withRoleName(roleName)
            .withPolicyDocument(permissionPolicy.toString())
            .withPolicyName("crrRolePolicy");

        iamClient.putRolePolicy(putRolePolicyRequest);


    }
}
```

C#

Das folgende AWS SDK für .NET-Codebeispiel fügt zuerst eine Replikationskonfiguration zu einem Bucket hinzu und ruft ihn anschließend ab. Um diesen Code zu verwenden, geben Sie die Namen für die Buckets und den Amazon-Ressourcennamen (ARN) für die IAM-Rolle an. Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CrossRegionReplicationTest
    {
        private const string sourceBucket = "*** source bucket ***";
        // Bucket ARN example - arn:aws:s3:::destinationbucket
        private const string destinationBucketArn = "*** destination bucket ARN ***";
        private const string roleArn = "*** IAM Role ARN ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint sourceBucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(sourceBucketRegion);
            EnableReplicationAsync().Wait();
```

```
        }
        static async Task EnableReplicationAsync()
        {
            try
            {
                ReplicationConfiguration replConfig = new ReplicationConfiguration
                {
                    Role = roleArn,
                    Rules =
                        {
                            new ReplicationRule
                            {
                                Prefix = "Tax",
                                Status = ReplicationRuleStatus.Enabled,
                                Destination = new ReplicationDestination
                                {
                                    BucketArn = destinationBucketArn
                                }
                            }
                        }
                };

                PutBucketReplicationRequest putRequest = new
PutBucketReplicationRequest
                {
                    BucketName = sourceBucket,
                    Configuration = replConfig
                };

                PutBucketReplicationResponse putResponse = await
s3Client.PutBucketReplicationAsync(putRequest);

                // Verify configuration by retrieving it.
                await RetrieveReplicationConfigurationAsync(s3Client);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }
        private static async Task RetrieveReplicationConfigurationAsync(IAmazonS3
client)
        {
            // Retrieve the configuration.
            GetBucketReplicationRequest getRequest = new GetBucketReplicationRequest
            {
                BucketName = sourceBucket
            };
            GetBucketReplicationResponse getResponse = await
client.GetBucketReplicationAsync(getRequest);
            // Print.
            Console.WriteLine("Printing replication configuration information...");
            Console.WriteLine("Role ARN: {0}", getResponse.Configuration.Role);
            foreach (var rule in getResponse.Configuration.Rules)
            {
                Console.WriteLine("ID: {0}", rule.Id);
                Console.WriteLine("Prefix: {0}", rule.Prefix);
                Console.WriteLine("Status: {0}", rule.Status);
            }
        }
```

```
    }
}
```

Beispiel 2: Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Besitz verschiedener Konten befinden.

Das Einrichten der Replikation, wenn sich *Quell*- und *Ziel*-Bucket im Besitz verschiedener AWS-Konten befinden, ähnelt der Einrichtung der Replikation, wenn beide Buckets zum selben Konto gehören. Der einzige Unterschied besteht darin, dass der Eigentümer des *Ziel*-Buckets dem Eigentümer des *Quell*-Buckets die Berechtigung zum Replizieren von Objekten gewährt, indem er eine Bucket-Richtlinie hinzufügt.

So richten Sie die Konfiguration ein, wenn sich Quell- und Ziel-Bucket im Besitz verschiedener AWS-Konten befinden

1. In diesem Beispiel erstellen Sie *Quell*- und *Ziel*-Bucket in zwei unterschiedlichen AWS-Konten. Sie müssen zwei Anmeldeinformationsprofile für die AWS-CLI eingerichtet haben (in diesem Beispiel haben wir `acctA` und `acctB` als Profilnamen verwendet). Weitere Informationen zum Einrichten der Anmeldeinformationsprofile finden Sie unter Benannte Profile im Benutzerhandbuch für AWS Command Line Interface.
2. Befolgen Sie die Schritt-für-Schritt-Anleitung in Beispiel 1: Konfigurieren für Buckets im selben Konto (p. 699) mit den folgenden Änderungen:

   - Verwenden Sie für alle AWS CLI-Befehle im Zusammenhang mit Aktivitäten des *Quell*-Buckets (Erstellen des *Quell*-Buckets, Aktivieren des Versionings und Erstellen der IAM-Rolle) das Profil `acctA`. Verwenden Sie das Profil `acctB`, um den *Ziel*-Bucket zu erstellen.
   - Stellen Sie sicher, dass die Berechtigungsrichtlinie den *Quell*- und den *Ziel*-Bucket angibt, die Sie für dieses Beispiel erstellt haben.
3. Fügen Sie in der Konsole die folgende Bucket-Richtlinie für den *Ziel*-Bucket hinzu, um dem Eigentümer des *Quell*-Buckets die Berechtigung zum Replizieren von Objekten zu erteilen. Stellen Sie sicher, dass Sie die Richtlinie bearbeiten, indem Sie die AWS-Konto-ID des Eigentümers des *Quell*-Buckets und den Namen des *Ziel*-Buckets angeben.

```
{
   "Version":"2008-10-17",
   "Id":"",
   "Statement":[
      {
         "Sid":"1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::source-bucket-acct-ID:source-acct-IAM-role"
         },
         "Action":["s3:ReplicateObject", "s3:ReplicateDelete"],
         "Resource":"arn:aws:s3:::destination/*"
      },
      {
         "Sid":"2",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::source-bucket-acct-ID:source-acct-IAM-role"
         },
         "Action":["s3:GetBucketVersioning", "s3:PutBucketVersioning"],
         "Resource":"arn:aws:s3:::destination"
      }
```

```
    ]
}
```

Wählen Sie den Bucket aus und fügen Sie die Bucket-Richtlinie hinzu. Weitere Informationen finden Sie unter Wie füge ich eine S3-Bucket-Richtlinie hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Beispiel 3: Ändern des Replikateigentümers, wenn sich Quell- und Ziel-Bucket im Besitz unterschiedlicher Konten befinden

Wenn sich der *Quell*- und der *Ziel*-Bucket bei einer Replikation im Besitz verschiedener AWS-Konten befinden, können Sie Amazon S3 anweisen, die Replikateigentümerschaft zu dem AWS-Konto zu ändern, das den *Ziel*-Bucket besitzt. Dieses Beispiel erläutert, wie Sie mit der Amazon S3-Konsole und der AWS CLI die Replikatseigentümerschaft ändern. Weitere Informationen finden Sie unter Ändern des Replikateigentümers (p. 690).

Ändern des Replikateigentümers, wenn sich die Buckets im Besitz unterschiedlicher Konten befinden (Konsole)

Schritt-für-Schritt-Anleitungen finden Sie im Thema über Konfigurieren einer Replikationsregel, wenn sich der Ziel-Bucket in einem anderen AWS-Konto befindet im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Ändern des Replikateigentümers, wenn sich die Buckets im Besitz unterschiedlicher AWS-Konten (AWS-CLI) befinden

Um die Replikateigentümerschaft mit der AWS CLI zu ändern, erstellen Sie Buckets, aktivieren das Versioning für die Buckets, erstellen eine IAM-Rolle, die Amazon S3 die Berechtigung zum Replizieren von Objekten gibt, und fügen die Replikationskonfiguration zum Quell-Bucket hinzu. In der Replikationskonfiguration weisen Sie Amazon S3 an, den Replikatseigentümer zu ändern. Sie testen außerdem die Einrichtung.

Ändern der Replikatseigentümerschaft, wenn sich Quell- und Ziel-Bucket im Besitz unterschiedlicher AWS-Konten (AWS-CLI) befinden

1. In diesem Beispiel erstellen Sie *Quell*- und *Ziel*-Buckets in zwei unterschiedlichen AWS-Konten. Konfigurieren Sie die AWS-CLI mit zwei benannten Profilen. Bei diesem Beispiel werden die Profile `acctA` bzw. `acctB`verwendet. Weitere Informationen zum Einrichten der Anmeldeinformationsprofile finden Sie unter Benannte Profile im Benutzerhandbuch für AWS Command Line Interface.

   Important

   Die Profile, die Sie für diese Übung verwenden, müssen über die nötigen Berechtigungen verfügen. Beispielsweise legen Sie in der Replikationskonfiguration die IAM-Rolle fest, die Amazon S3 annehmen kann. Dies können Sie nur tun, wenn das verwendete Profil über die `iam:PassRole`-Berechtigung verfügt. Wenn Sie zur Erstellung eines benannten Profils die Benutzeranmeldeinformationen eines Administrators verwenden, können Sie alle Aufgaben durchführen. Weitere Informationen finden Sie unter Gewähren von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch.

   Sie müssen sicherstellen, dass diese Profile über die nötigen Berechtigungen verfügen. Beispielsweise enthält die Replikationskonfiguration eine IAM-Rolle, die Amazon S3 annehmen kann. Das benannte Profil, mit dem Sie eine solche Konfiguration einem Bucket anfügen, kann dies nur tun, wenn es über

die `iam:PassRole`-Berechtigung verfügt. Wenn Sie zur Erstellung dieser benannten Profile die Benutzeranmeldeinformationen eines Administrators verwenden, haben diese alle Berechtigungen. Weitere Informationen finden Sie unter Gewähren von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch.

2. Erstellen Sie den *Quell*-Bucket und aktivieren Sie das Versioning für ihn. In diesem Beispiel erstellen wir den *Quell*-Bucket in der Region USA Ost (Nord-Virginia) (us-east-1).

```
aws s3api create-bucket \
--bucket source \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket source \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Erstellen Sie einen *Ziel*-Bucket und aktivieren Sie das Versioning für ihn. In diesem Beispiel erstellen wir den *Ziel*-Bucket in der Region USA West (Oregon) (us-west-2). Verwenden Sie ein anderes AWS-Kontoprofil als das, das Sie für den *Quell*-Bucket verwendet haben.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctB
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctB
```

4. Sie müssen Ihrer *Ziel*-Bucket-Richtlinie Berechtigungen hinzufügen, um eine Änderung der Replikateigentümerschaft zuzulassen.

   a. Speichern Sie die folgende Richtlinie in *destination-bucket-policy*`.json`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "destination_bucket_policy_sid",
      "Principal": {
        "AWS": "source-bucket-owner-account-id"
      },
      "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateDelete"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::destination/*"
      ]
    }
  ]
}
```

   b. Legen Sie die obige Richtlinie im *Ziel*-Bucket ab:

```
aws s3api put-bucket-policy --region $ {destination_region} --
bucket $ {destination} --policy file://destination_bucket_policy.json
```

5. Erstellen Sie eine IAM-Rolle. Sie legen diese Rolle in der Replikationskonfiguration fest, die Sie später zum *Quell*-Bucket hinzufügen. Amazon S3 nimmt diese Rolle an, um in Ihrem Namen Objekte zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   - Erstellen Sie eine Rolle.
   - Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

   a. Erstellen Sie eine IAM-Rolle.

      i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `S3-role-trust-policy.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie erteilt Amazon S3 Berechtigungen für die Übernahme der Rolle.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

      ii. Führen Sie den folgenden AWS CLI-Befehl aus, um eine Rolle zu erstellen:

```
$ aws iam create-role \
--role-name replicationRole \
--assume-role-policy-document file://s3-role-trust-policy.json  \
--profile acctA
```

   b. Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

      i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-perm-pol-changeowner.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Zugriffsrichtlinie erteilt Berechtigungen für verschiedene Amazon S3-Bucket- und -Objektaktionen. In den folgenden Schritten erstellen Sie eine IAM-Rolle und fügen diese Richtlinie der Rolle an.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::source/*"
         ]
      },
      {
         "Effect":"Allow",
```

```
            "Action":[
               "s3:ListBucket",
               "s3:GetReplicationConfiguration"
            ],
            "Resource":[
               "arn:aws:s3:::source"
            ]
         },
         {
            "Effect":"Allow",
            "Action":[
               "s3:ReplicateObject",
               "s3:ReplicateDelete",
               "s3:ObjectOwnerOverrideToBucketOwner",
               "s3:ReplicateTags",
               "s3:GetObjectVersionTagging"
            ],
            "Resource":"arn:aws:s3:::destination/*"
         }
      ]
}
```

ii. Um eine Richtlinie zu erstellen und sie an die Rolle anzufügen, führen Sie den folgenden Befehl aus:

```
$ aws iam put-role-policy \
--role-name replicationRole \
--policy-document file://s3-role-perm-pol-changeowner.json \
--policy-name replicationRolechangeownerPolicy \
--profile acctA
```

6. Fügen Sie Ihrem Quell-Bucket eine Replikationskonfiguration hinzu.

   a. Die AWS CLI erfordert, dass Sie die Replikationskonfiguration als JSON festlegen. Speichern Sie den folgenden JSON-Code in einer Datei mit dem Namen `replication.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Fügen Sie in der Konfiguration `AccessControlTranslation` hinzu, um eine Änderung in der Replikateigentümerschaft anzugeben.

```
{
   "Role":"IAM-role-ARN",
   "Rules":[
      {
         "Status":"Enabled",
         "Priority":"1",
         "DeleteMarkerReplication":{
            "Status":"Disabled"
         },
         "Filter":{
            "Prefix":"Tax"
         },
         "Status":"Enabled",
         "Destination":{
            "Bucket":"arn:aws:s3:::destination",
            "Account":"destination-bucket-owner-account-id",
            "AccessControlTranslation":{
               "Owner":"Destination"
            }
         }
      }
   ]
}
```

b. Bearbeiten Sie den JSON-Code, indem Sie Werte für die Konto-ID des Eigentümers des *Ziel*-Buckets und für *IAM-role-ARN* angeben. Speichern Sie die Änderungen.

c. Um die Replikationskonfiguration zum Quell-Bucket hinzuzufügen, führen Sie den folgenden Befehl aus. Geben Sie den Namen für den *Quell*-Bucket an.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication-changeowner.json \
--bucket source \
--profile acctA
```

7. Überprüfen Sie die Replikatseigentümerschaft in der Amazon S3-Konsole.

a. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3.

b. Erstellen Sie im *Quell*-Bucket einen Ordner mit dem Namen `Tax`.

c. Fügen Sie dem Ordner im *Quell*-Bucket Objekte hinzu. Überprüfen Sie, ob der *Ziel*-Bucket die Objektreplikate enthält und ob die Eigentümerschaft der Replikate zu dem AWS-Konto geändert wurde, das den *Ziel*-Bucket besitzt.

Ändern des Replikateigentümers, wenn sich die Buckets im Besitz unterschiedlicher Konten (AWS-SDK) befinden

Ein Code-Beispiel zum Hinzufügen einer Replikationskonfiguration finden Sie unter Konfigurieren der Replikation, wenn die Buckets demselben Konto angehören (AWS-SDK) (p. 703). Sie müssen die Replikationskonfiguration entsprechend ändern. Weitere konzeptuelle Informationen finden Sie unter Ändern des Replikateigentümers (p. 690).

Beispiel 4: Replizieren verschlüsselter Objekte

Standardmäßig repliziert Amazon S3 keine Objekte, die sich im Ruhezustand befinden, mit der serverseitiger Verschlüsselung mit AWS Key Management Service (AWS KMS) Kundenhauptschlüsseln (CMKs). Um verschlüsselte Objekte zu replizieren, ändern Sie die Bucket-Replikationskonfiguration und weisen damit Amazon S3 an, diese Objekte zu replizieren. Dieses Beispiel erläutert, wie Sie mit der Amazon S3-Konsole und der AWS Command Line Interface (AWS CLI) die Bucket-Replikationskonfiguration ändern, um die Replikation verschlüsselter Objekte zu aktivieren. Weitere Informationen finden Sie unter Replizieren von Objekten, die mit der serverseitigen Verschlüsselung (SSE) unter Verwendung von in AWS KMS gespeicherten Verschlüsselungsschlüsseln erstellt wurden (p. 693).

Replizieren verschlüsselter Objekte (Konsole)

Detaillierte Anleitungen finden Sie im Thema über Wie füge ich eine Replikationsregel zu einem S3-Bucket hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service. Dieses Thema enthält Anleitungen für die Replikationskonfiguration, wenn sich die Buckets im Besitz derselben und unterschiedlicher AWS-Konten befinden.

Replizieren verschlüsselter Objekte (AWS-CLI)

Um verschlüsselte Objekte mit der AWS CLI zu replizieren, erstellen Sie Buckets, aktivieren das Versioning für die Buckets, erstellen eine IAM-Rolle, die Amazon S3 die Berechtigung zum Replizieren von Objekten gibt, und fügen die Replikationskonfiguration zum Quell-Bucket hinzu. Die Replikationskonfiguration stellt Informationen zur Replizierung von Objekten bereit, die mit KMS-Schlüsseln verschlüsselt wurden. Zu den Berechtigungen für die IAM-Rolle gehören die, die zum Replizieren der verschlüsselten Objekte notwendig sind. Sie testen außerdem die Einrichtung.

So replizieren Sie serverseitig verschlüsselte Objekte (AWS-CLI):

1. In diesem Beispiel erstellen wir *Quell*- und *Ziel*-Bucket im selben AWS-Konto. Richten Sie das Anmeldeinformationsprofil für die AWS CLI ein. In diesem Beispiel verwenden wir den Profilnamen `acctA`. Weitere Informationen zum Einrichten der Anmeldeinformationsprofile finden Sie unter Benannte Profile im Benutzerhandbuch für AWS Command Line Interface.
2. Erstellen Sie den *Quell*-Bucket und aktivieren Sie das Versioning für ihn. In diesem Beispiel erstellen wir den *Quell*-Bucket in der Region USA Ost (Nord-Virginia) (us-east-1).

```
aws s3api create-bucket \
--bucket source \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket source \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Erstellen Sie den *Ziel*-Bucket und aktivieren Sie das Versioning für ihn. In diesem Beispiel erstellen wir den *Ziel*-Bucket in der Region USA West (Oregon) (us-west-2).

   Note

   Um die Replikationskonfiguration einzurichten, wenn sich *Quell*- und *Ziel*-Bucket im selben AWS-Konto befinden, verwenden Sie dasselbe Profil. In diesem Beispiel verwenden wir `acctA`. Um die Replikationskonfiguration zu testen, wenn sich die Buckets im Besitz unterschiedlicher AWS-Konten befinden, legen Sie verschiedene Profile für die Buckets fest.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctA
```

4. Erstellen Sie eine IAM-Rolle. Sie legen diese Rolle in der Replikationskonfiguration fest, die Sie später zum *Quell*-Bucket hinzufügen. Amazon S3 nimmt diese Rolle an, um in Ihrem Namen Objekte zu replizieren. Sie erstellen eine IAM-Rolle in zwei Schritten:

   - Erstellen einer Rolle
   - Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu.

   a. Erstellen Sie eine IAM-Rolle.

      i. Kopieren Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-trust-policy-kmsobj.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Diese Richtlinie weist dem Amazon S3-Service-Prinzipal die Berechtigungen zum Annehmen der Rolle zu, damit Amazon S3 Aufgaben in Ihrem Namen durchführen kann.

```
{
   "Version":"2012-10-17",
```

```
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

ii. Erstellen Sie eine Rolle.

```
$ aws iam create-role \
--role-name replicationRolekmsobj \
--assume-role-policy-document file://s3-role-trust-policy-kmsobj.json  \
--profile acctA
```

b. Fügen Sie eine Berechtigungsrichtlinie zur Rolle hinzu. Diese Zugriffsrichtlinie erteilt Berechtigungen für verschiedene Amazon S3-Bucket- und -Objektaktionen.

i. Kopieren Sie die folgende Berechtigungsrichtlinie und speichern Sie sie in einer Datei mit dem Namen `s3-role-permissions-policykmsobj.json` im aktuellen Verzeichnis auf Ihrem lokalen Computer. Sie erstellen eine IAM-Rolle und fügen ihr später die Richtlinie an.

> **Important**
>
> In der Berechtigungsrichtlinie legen Sie die KMS-Schlüssel-IDs für AWS fest, die zur Verschlüsselung der Quell- und `destination`-Buckets verwendet werden. Sie müssen zwei separate AWS KMS-CMKs für die Buckets `source` und `destination` erstellen. AWS KMS-CMKs werden niemals außerhalb der AWS-Region, in der sie erstellt wurden, freigegeben.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration",
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::source",
            "arn:aws:s3:::source/*"
         ]
      },
      {
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags",
            "s3:GetObjectVersionTagging"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLikeIfExists":{
               "s3:x-amz-server-side-encryption":[
                  "aws:kms",
                  "AES256"
               ],
```

```
                "s3:x-amz-server-side-encryption-aws-kms-key-id":[
                   "AWS KMS key IDs(in ARN format) to use for encrypting object
 replicas"
                ]
             }
          },
          "Resource":"arn:aws:s3:::destination/*"
       },
       {
          "Action":[
             "kms:Decrypt"
          ],
          "Effect":"Allow",
          "Condition":{
             "StringLike":{
                "kms:ViaService":"s3.us-east-1.amazonaws.com",
                "kms:EncryptionContext:aws:s3:arn":[
                   "arn:aws:s3:::source/*"
                ]
             }
          },
          "Resource":[
             "AWS KMS key IDs(in ARN format) used to encrypt source objects."
          ]
       },
       {
          "Action":[
             "kms:Encrypt"
          ],
          "Effect":"Allow",
          "Condition":{
             "StringLike":{
                "kms:ViaService":"s3.us-west-2.amazonaws.com",
                "kms:EncryptionContext:aws:s3:arn":[
                   "arn:aws:s3:::destination/*"
                ]
             }
          },
          "Resource":[
             "AWS KMS key IDs(in ARN format) to use for encrypting object
 replicas"
          ]
       }
    ]
}
```

ii. Erstellen Sie eine Richtlinie und hängen Sie sie an die Rolle an.

```
$ aws iam put-role-policy \
--role-name replicationRolekmsobj \
--policy-document file://s3-role-permissions-policykmsobj.json \
--policy-name replicationRolechangeownerPolicy \
--profile acctA
```

5. Fügen Sie die folgende Replikationskonfiguration zum *Quell*-Bucket hinzu. Sie weist Amazon S3 an, Objekte mit dem Präfix `Tax/` in den angegebenen *Ziel*-Bucket zu replizieren.

   Important

   In der Replikationskonfiguration legen Sie die IAM-Rolle fest, die Amazon S3 annehmen kann. Dies können Sie nur tun, wenn Sie über die `iam:PassRole`-Berechtigung verfügen. Das Profil, das Sie im CLI-Befehl angeben, muss über die Berechtigung verfügen. Weitere Informationen finden Sie unter Gewähren von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch.

```
 <ReplicationConfiguration>
  <Role>IAM-Role-ARN</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>Disabled</Status>
    </DeleteMarkerReplication>
    <Filter>
       <Prefix>Tax</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <SourceSelectionCriteria>
      <SseKmsEncryptedObjects>
        <Status>Enabled</Status>
      </SseKmsEncryptedObjects>
    </SourceSelectionCriteria>
    <Destination>
      <Bucket>arn:aws:s3:::dest-bucket-name</Bucket>
      <EncryptionConfiguration>
        <ReplicaKmsKeyID>AWS KMS key IDs to use for encrypting object replicas</
ReplicaKmsKeyID>
      </EncryptionConfiguration>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Um die Replikationskonfiguration zum *Quell*-Bucket hinzuzufügen, gehen Sie wie folgt vor:

a. Die AWS CLI erfordert, dass Sie die Replikationskonfiguration als JSON festlegen. Speichern Sie den folgenden JSON-Code in einer Datei (`replication.json`) im aktuellen Verzeichnis auf Ihrem lokalen Computer.

```
{
   "Role":"IAM-Role-ARN",
   "Rules":[
      {
         "Status":"Enabled",
         "Priority":1,
         "DeleteMarkerReplication":{
            "Status":"Disabled"
         },
         "Filter":{
            "Prefix":"Tax"
         },
         "Destination":{
            "Bucket":"arn:aws:s3:::destination",
            "EncryptionConfiguration":{
               "ReplicaKmsKeyID":"AWS KMS key IDs(in ARN format)  to use for
 encrypting object replicas"
            }
         },
         "SourceSelectionCriteria":{
            "SseKmsEncryptedObjects":{
               "Status":"Enabled"
            }
         }
      }
   ]
}
```

b. Bearbeiten Sie den JSON-Code, indem Sie Werte für den *Ziel*-Bucket, den *KMS-ID-ARN* und den *IAM-role-ARN* angeben. Speichern Sie die Änderungen.

c. Fügen Sie Ihrem *Quell*-Bucket diese Replikationskonfiguration hinzu. Stellen Sie sicher, dass Sie den Namen für den *Quell*-Bucket angeben.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket source \
--profile acctA
```

6. Testen Sie die Einrichtung, um zu überprüfen, ob die verschlüsselten Objekte repliziert werden. In der Amazon S3-Konsole:

a. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3.

b. Erstellen Sie im *Quell*-Bucket einen Ordner mit dem Namen `Tax`.

c. Fügen Sie Beispielobjekte zum Ordner hinzu. Stellen Sie sicher, dass Sie die Verschlüsselungsoption wählen und Ihren AWS KMS-CMK angeben, um die Objekte zu verschlüsseln.

d. Vergewissern Sie sich, dass der *Ziel*-Bucket die Objektreplikate enthält und dass sie mit dem AWS KMS-CMK verschlüsselt sind, den Sie in der Konfiguration angegeben haben.

Replizieren verschlüsselter Objekte (AWS-SDK)

Ein Code-Beispiel zum Hinzufügen einer Replikationskonfiguration finden Sie unter Konfigurieren der Replikation, wenn die Buckets demselben Konto angehören (AWS-SDK) (p. 703). Sie müssen die Replikationskonfiguration entsprechend ändern.

Weitere konzeptuelle Informationen finden Sie unter Replizieren von Objekten, die mit der serverseitigen Verschlüsselung (SSE) unter Verwendung von in AWS KMS gespeicherten Verschlüsselungsschlüsseln erstellt wurden (p. 693).

Beispiel 5: S3 Replication Time Control (S3 RTC)-Konfiguration

S3 Replication Time Control (S3 RTC) unterstützt Sie bei der Erfüllung von Compliance- oder geschäftlichen Anforderungen für die Datenreplikation und bietet Einblick in die Amazon S3 Replikationszeiten. S3 RTC repliziert die meisten Objekte, die Sie zu Amazon S3 hochladen, in Sekunden und 99,99 Prozent dieser Objekte innerhalb von 15 Minuten.

Mit S3 RTC können Sie die Gesamtzahl und Größe der Objekte, die zur Replikation ausstehen, sowie die maximale Replikationszeit in die Zielregion überwachen. Replikationsmetriken sind über die AWS Management Console und Amazon CloudWatch-Benutzerhandbuch verfügbar. Weitere Informationen finden Sie unter the section called “Amazon S3 CloudWatch-Replikationsmetriken” (p. 743)

Aktivieren von S3 RTC in der Amazon S3-Konsole

Detaillierte Anleitungen finden Sie im Thema über Wie füge ich eine Replikationsregel zu einem S3-Bucket hinzu? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service. Dieses Thema enthält Anleitungen zum Aktivieren von S3 RTC in Ihrer Replikationskonfiguration, wenn sich die Buckets im Besitz derselben und unterschiedlicher AWS-Konten befinden.

Replikation von Objekten mit Amazon S3 Replication Time Control (AWS CLI)

Um Objekte, auf denen S3 RTC aktiviert ist, mit der AWS CLI zu replizieren, erstellen Sie Buckets, aktivieren das Versioning für die Buckets, erstellen eine IAM-Rolle, die Amazon S3 die Berechtigung zum

Replizieren von Objekten gibt, und fügen die Replikationskonfiguration zum Quell-Bucket hinzu. Für die Replikationskonfiguration muss S3 Replication Time Control (S3 RTC) aktiviert sein.

So replizieren Sie mit aktiviertem S3 RTC (AWS CLI)

- In diesem Beispiel legen wir `ReplicationTime` und `Metric` fest und fügen die Replikationskonfiguration zum Quell-Bucket hinzu.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "DeleteMarkerReplication": {
                "Status": "Disabled"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::destination",
                "Metrics": {
                    "Status": "Enabled",
                    "EventThreshold": {
                        "Minutes": 15
                    }
                },
                "ReplicationTime": {
                    "Status": "Enabled",
                    "Time": {
                        "Minutes": 15
                    }
                }
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

Important

Für `Metrics:EventThreshold:Minutes` und `ReplicationTime:Time:Minutes` ist als gültiger Wert nur „15“ zulässig.

Replikation von Objekten mit Amazon S3 Replication Time Control (AWS SDK)

Es folgt ein Java-Beispiel, um die Replikationskonfiguration mit S3 Replication Time Control (S3 RTC) hinzuzufügen:

```
import software.amazon.awssdk.auth.credentials.AwsBasicCredentials;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.model.DeleteMarkerReplication;
import software.amazon.awssdk.services.s3.model.Destination;
import software.amazon.awssdk.services.s3.model.Metrics;
import software.amazon.awssdk.services.s3.model.MetricsStatus;
import software.amazon.awssdk.services.s3.model.PutBucketReplicationRequest;
import software.amazon.awssdk.services.s3.model.ReplicationConfiguration;
import software.amazon.awssdk.services.s3.model.ReplicationRule;
import software.amazon.awssdk.services.s3.model.ReplicationRuleFilter;
import software.amazon.awssdk.services.s3.model.ReplicationTime;
import software.amazon.awssdk.services.s3.model.ReplicationTimeStatus;
```

```
import software.amazon.awssdk.services.s3.model.ReplicationTimeValue;

public class Main {

  public static void main(String[] args) {
    S3Client s3 = S3Client.builder()
      .region(Region.US_EAST_1)
      .credentialsProvider(() -> AwsBasicCredentials.create(
          "AWS_ACCESS_KEY_ID",
          "AWS_SECRET_ACCESS_KEY")
      )
      .build();

    ReplicationConfiguration replicationConfig = ReplicationConfiguration
      .builder()
      .rules(
          ReplicationRule
            .builder()
            .status("Enabled")
            .priority(1)
            .deleteMarkerReplication(
                DeleteMarkerReplication
                    .builder()
                    .status("Disabled")
                    .build()
            )
            .destination(
                Destination
                    .builder()
                    .bucket("destination_bucket_arn")
                    .replicationTime(
                        ReplicationTime.builder().time(
                            ReplicationTimeValue.builder().minutes(15).build()
                        ).status(
                            ReplicationTimeStatus.ENABLED
                        ).build()
                    )
                    .metrics(
                        Metrics.builder().eventThreshold(
                            ReplicationTimeValue.builder().minutes(15).build()
                        ).status(
                            MetricsStatus.ENABLED
                        ).build()
                    )
                    .build()
            )
            .filter(
                ReplicationRuleFilter
                    .builder()
                    .prefix("testtest")
                    .build()
            )
        .build())
        .role("role_arn")
        .build();

    // Put replication configuration
    PutBucketReplicationRequest putBucketReplicationRequest = PutBucketReplicationRequest
      .builder()
      .bucket("source_bucket")
      .replicationConfiguration(replicationConfig)
      .build();

    s3.putBucketReplication(putBucketReplicationRequest);
  }
}
```

Weitere Informationen finden Sie unter Replizieren von Objekten mit S3 Replication Time Control (S3 RTC) (p. 687).

Informationen zum Replikationsstatus

Um den Status der Replikation von Objekten in einem Bucket abzurufen, verwenden Sie das Amazon S3-Bestandstool. Amazon S3 sendet eine CSV-Datei an den Ziel-Bucket, den Sie in der Bestandskonfiguration angeben. Sie können auch mit Amazon Athena den Replikationsstatus im Bestandsbericht abfragen. Weitere Informationen zum Amazon S3-Bestand finden Sie unter Amazon S3-Bestand (p. 503).

Bei der Replikation haben Sie einen Quell-Bucket, auf dem die Replikation konfiguriert wird, und einen Ziel-Bucket, in den Amazon S3 Objekte repliziert. Wenn Sie ein Objekt (mit `GET`-Objekt) oder Objektmetadaten (mit `HEAD`-Objekt) von diesen Buckets anfordern, gibt Amazon S3 den Header `x-amz-replication-status` wie folgt in der Antwort zurück:

- Wenn Sie ein Objekt aus dem Quell-Bucket anfordern, gibt Amazon S3 den Header `x-amz-replication-status` zurück, wenn das Objekt in der Anforderung für die Replikation geeignet ist.

  Nehmen wir beispielsweise an, dass Sie in Ihrer Replikationskonfiguration das Objektpräfix `TaxDocs` angeben, um Amazon S3 anzuweisen, nur Objekte mit dem Schlüsselnamenpräfix `TaxDocs` zu replizieren. Alle Objekte mit diesem Schlüsselnamenpräfix, die Sie hochladen, z. B. `TaxDocs/document1.pdf`, werden repliziert. Für Objektanforderungen mit diesem Schlüsselnamenpräfix gibt Amazon S3 den Header `x-amz-replication-status` mit einem der folgenden Werte für den Replikationsstatus des Objekts zurück: `PENDING`, `COMPLETED` oder `FAILED`.

  > **Note**
  >
  > Wenn nach dem Hochladen eines Objekts die Objektreplikation fehlschlägt, können Sie die fehlgeschlagene Replikation nicht erneut durchzuführen versuchen. Sie müssen das Objekt erneut hochladen.

- Wenn Sie ein Objekt aus dem Ziel-Bucket hochladen und es sich bei dem Objekt Ihrer Anforderung um ein Replikat handelt, das Amazon S3 erstellt hat, gibt Amazon S3 den Header `x-amz-replication-status` mit dem Wert `REPLICA` zurück.

Sie finden den Objektreplikationsstatus mit der Konsole, der AWS Command Line Interface (AWS CLI) oder dem AWS-SDK heraus.

- Konsole: Wählen Sie das Objekt und dann Properties (Eigenschaften) aus, um die Objekteigenschaften, einschließlich des Replikationsstatus, anzuzeigen.
- AWS CLI: Verwenden Sie den `head-object` AWS CLI-Befehl, um die Objektmetadaten abzurufen.

```
aws s3api head-object --bucket source-bucket --key object-key --version-id object-version-id
```

Der Befehl gibt Informationen zu Objektmetadaten, einschließlich des `ReplicationStatus`, zurück, wie in der folgenden Beispielantwort dargestellt:

```
{
   "AcceptRanges":"bytes",
   "ContentType":"image/jpeg",
   "LastModified":"Mon, 23 Mar 2015 21:02:29 GMT",
   "ContentLength":3191,
   "ReplicationStatus":"COMPLETED",
   "VersionId":"jfnW.HIMOfYiD_9rGbSkmroXsFj3fqZ.",
```

```
   "ETag":"\"6805f2cfc46c0f04559748bb039d69ae\"",
   "Metadata":{

   }
}
```

- AWS-SDKs: Mit den folgenden Code-Fragmenten können Sie den Replikationsstatus mit AWS SDK for Java bzw. AWS SDK für .NET abrufen.
  - AWS SDK for Java

```
GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest(bucketName,
 key);
ObjectMetadata metadata = s3Client.getObjectMetadata(metadataRequest);

System.out.println("Replication Status : " +
 metadata.getRawMetadataValue(Headers.OBJECT_REPLICATION_STATUS));
```

  - AWS SDK für .NET

```
GetObjectMetadataRequest getmetadataRequest = new GetObjectMetadataRequest
    {
         BucketName = sourceBucket,
         Key        = objectKey
    };

GetObjectMetadataResponse getmetadataResponse =
 client.GetObjectMetadata(getmetadataRequest);
Console.WriteLine("Object replication status: {0}",
 getmetadataResponse.ReplicationStatus);
```

Note

Bevor Sie ein Objekt aus einem Quell-Bucket löschen, bei dem die Replikation aktiviert ist, sollten Sie den Replikationsstatus des Objekts überprüfen, um sicherzustellen, dass das Objekt repliziert wurde.
Wenn die Lebenszykluskonfiguration auf dem Quell-Bucket aktiviert ist, setzt Amazon S3 alle Lebenszyklusaktionen so lange aus, bis der Status des Objekts als `COMPLETED` oder `FAILED` gekennzeichnet wird.

Verwandte Themen

Replikation (p. 669)

Fehlerbehebung bei einer Replikation

Wenn Objektreplikate nicht im Ziel-Bucket erscheinen, nachdem Sie die Replikation konfiguriert haben, können Sie mit diesen Tipps zur Fehlerbehebung Probleme identifizieren und beheben.

- Die meisten Objekte werden innerhalb von 15 Minuten repliziert, manchmal kann dies jedoch auch einige Stunden dauern. In seltenen Fällen kann die Replikation noch mehr Zeit in Anspruch nehmen. Die von Amazon S3 für die Replikation eines Objekts benötigte Zeit hängt von verschiedenen Faktoren ab, einschließlich des Quell- und Ziel-Regionspaars sowie der Größe des Objekts. Bei großen Objekten kann die Replikation mehrere Stunden dauern. Wenn das Objekt, das repliziert wird, groß ist, warten Sie eine Weile bevor Sie überprüfen, ob es im Ziel-Bucket angezeigt wird.
  Sie können auch den Replikationsstatus des Quellobjekts überprüfen. Wenn der Replikationsstatus des Objekts `pending` (ausstehend) lautet, wissen Sie, dass Amazon S3 die Replikation noch nicht

abgeschlossen hat. Wenn der Replikationsstatus des Objekts `failed` (fehlgeschlagen) lautet, sollten Sie die Replikationskonfiguration des Quell-Buckets überprüfen.

- Überprüfen Sie in der Replikationskonfiguration des Quell-Buckets Folgendes:
  - ob der Amazon-Ressourcennamen (ARN) des Ziel-Buckets korrekt ist.
  - ob das Schlüsselnamenpräfix korrekt ist. Wenn Sie beispielsweise die Konfiguration so einrichten, dass nur Objekte mit dem Präfix `Tax` repliziert werden, werden nur Objekte mit Schlüsselnamen wie beispielsweise `Tax/document1` oder `Tax/document2` repliziert. Ein Objekt mit dem Schlüsselnamen `document3` wird nicht repliziert.
  - ob der Status `enabled` lautet.
- Wenn sich der Ziel-Bucket im Besitz eines anderen AWS-Kontos befindet, stellen Sie sicher, dass der Bucket-Eigentümer eine Bucket-Richtlinie für den Ziel-Bucket eingerichtet hat, die dem Eigentümer des Quell-Buckets die Replikation von Objekten gestattet. Ein Beispiel finden Sie unter Beispiel 2: Konfigurieren der Replikation, wenn sich Quell- und Ziel-Buckets im Besitz verschiedener Konten befinden. (p. 708)
- Wenn ein Objektreplikat nicht im Ziel-Bucket angezeigt wird, könnte Folgendes die Replikation verhindert haben:
  - Amazon S3 repliziert keine Objekte in einem Quell-Bucket, der ein Replikat ist, das mit einer anderen Replikationskonfiguration erstellt wurde. Wenn Sie beispielsweise die Replikationskonfiguration von Bucket A zu Bucket B zu Bucket C festlegen, repliziert Amazon S3 keine Objektreplikate von Bucket B in Bucket C.
  - Ein Quell-Bucket-Eigentümer kann anderen AWS-Konten Berechtigungen für das Hochladen von Objekten erteilen. Standardmäßig besitzt der Quell-Bucket-Eigentümer keine Berechtigungen für die Objekte, die von anderen Konten erstellt wurden. Die Replikationskonfiguration repliziert nur die Objekte, für die der Quell-Bucket-Eigentümer über Zugriffsberechtigungen verfügt. Der Bucket-Eigentümer kann anderen AWS-Konten Berechtigungen zum bedingten Erstellen von Objekten erteilen. Dabei sind explizite Zugriffsberechtigungen für diese Objekte erforderlich. Eine Beispielrichtlinie finden Sie unter Erteilung von kontenübergreifenden Berechtigungen für das Hochladen von Objekten, wobei sichergestellt wird, dass der Bucket-Eigentümer volle Kontrolle besitzt (p. 453).
- Angenommen, Sie fügen einer Replikationskonfiguration eine Regel hinzu, um eine Teilmenge an Objekten mit einem bestimmten Tag zu replizieren. In diesem Fall müssen Sie den spezifischen Tag-Schlüssel und -Wert zum Zeitpunkt der Objekterstellung für Amazon S3 zuweisen, um das Objekt zu replizieren. Wenn Sie zuerst ein Objekt erstellen und dann das Tag zum vorhandenen Objekt hinzufügen, repliziert Amazon S3 das Objekt nicht.

Verwandte Themen

Replikation (p. 669)

Weitere Überlegungen zur Replikation

Amazon S3 unterstützt auch Bucket-Konfigurationen für Folgendes:

- Versioning — Weitere Informationen finden Sie unter Verwenden von Versioning (p. 514).
- Website-Hosting — Weitere Informationen finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).
- Bucket-Zugriff über eine Bucket-Richtlinie oder Zugriffskontrollliste (ACL) — Weitere Informationen finden Sie unter Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375) und Zugriffsverwaltung mit ACLs (p. 479).
- Protokollspeicher — Weitere Informationen finde Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).

- Lebenszyklusverwaltung für Objekte in einem Bucket — Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

In diesem Thema wird erläutert, wie die Bucket-Replikationskonfiguration das Verhalten dieser Bucket-Konfigurationen beeinflusst.

Themen

Lebenszykluskonfiguration und Objektreplikate

Die von Amazon S3 für die Replikation eines Objekts benötigte Zeit hängt von der Größe des Objekts ab. Bei großen Objekten kann dies mehrere Stunden dauern. Auch wenn es möglicherweise etwas dauert, bis ein Replikat im Ziel-Bucket verfügbar ist, entspricht die Erstellungszeit des Replikats der Erstellungszeit des entsprechenden Objekts im Quell-Bucket. Wenn Sie eine Lebenszyklusrichtlinie auf dem Ziel-Bucket eingerichtet haben, berücksichtigen die Lebenszyklusregeln die ursprüngliche Erstellungszeit des Objekts, nicht den Zeitpunkt, zu dem das Replikat im Ziel-Bucket verfügbar wurde.

Die Replikationskonfiguration erfordert einen Bucket mit aktiviertem Versioning. Berücksichtigen Sie bei der Aktivierung des Versioning in einem Bucket Folgendes:

- Wenn eine Lebenszyklus-Richtlinie für den Ablauf von Objekten vorhanden ist, nachdem Sie das Versioning aktiviert haben, fügen Sie eine `NonCurrentVersionExpiration`-Richtlinie hinzu, damit dasselbe permanente Löschverhalten beibehalten wird wie vor der Aktivierung des Versionings.
- Wenn eine Übertragungs-Lebenszyklus-Richtlinie vorhanden ist, nachdem Sie das Versioning aktiviert haben, sollten Sie in Betracht ziehen, eine `NonCurrentVersionTransition`-Richtlinie hinzuzufügen.

Versioning-Konfiguration und Replikationskonfiguration

Wenn Sie die Replikation auf einem Bucket konfigurieren, muss sowohl für den Quell- als auch für den Ziel-Bucket das Versioning aktiviert sein. Nachdem Sie sowohl im Quell- als auch im Ziel-Bucket das Versioning aktiviert und die Replikation im Quell-Bucket konfiguriert haben, werden Sie folgende Schwierigkeiten beobachten:

- Wenn Sie versuchen, das Versioning auf dem Quell-Bucket zu deaktivieren, gibt Amazon S3 einen Fehler zurück. Sie müssen die Replikationskonfiguration entfernen, damit Sie das Versioning auf dem Quell-Bucket deaktivieren können.
- Wenn Sie das Versioning auf dem Ziel-Bucket deaktivieren, schlägt die Replikation fehl. Das Quellobjekt weist den Replikationsstatus `Failed` auf.

Protokollierungskonfiguration und Replikationskonfiguration

Wenn Amazon S3 Protokolle an einen Bucket übermittelt, für den die Replikation aktiviert ist, repliziert der Service die Protokollobjekte.

Wenn Sie Server-Zugriffsprotokolle (Amazon S3-Serverzugriffsprotokollierung (p. 777)) oder AWS CloudTrail-Protokolle (Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail (p. 750)) im Quell- oder Ziel-Bucket aktiviert haben, schließt Amazon S3 die replikationsbezogenen Anforderungen in die Protokolle mit ein. Beispiel: Amazon S3 protokolliert jedes Objekt, das es repliziert.

CRR und die Zielregion

In der CRR-Konfiguration muss sich der Quell- und der Ziel-Bucket in verschiedenen AWS-Regionen befinden. Sie können die Region für Ihren Ziel-Bucket entweder auf der Basis Ihrer geschäftlichen Anforderungen oder von Kostenaspekten auswählen. Beispielsweise variieren die Gebühren der Datenübertragung zwischen Regionen je nach ausgewählten Regionen. Angenommen, Sie wählen USA Ost (Nord-Virginia) (us-east-1) als Region für Ihren Quell-Bucket aus. Wenn Sie USA West (Oregon) (us-west-2) als Region für Ihren Ziel-Bucket auswählen, zahlen Sie mehr, als wenn Sie die Region USA Ost (Ohio) (us-east-2) auswählen. Weitere Informationen zu Preisen finden Sie im Abschnitt "Datenübertragungspreise" unter Amazon S3 – Preise. Für die Replikation innerhalb derselben Region (SRR) fallen keine Datenübertragungskosten an.

Pausieren einer Replikation

Um mit der Replikation vorübergehend zu pausieren, deaktivieren Sie die entsprechende Regel in der Replikationskonfiguration.

Wenn die Replikation aktiviert ist und Sie die IAM-Rolle entfernen, die Amazon S3 die erforderlichen Berechtigungen gewährt, schlägt die Replikation fehl. Amazon S3 gibt den Replikationsstatus für die betroffenen Objekte als `Failed` an.

Verwandte Themen

Replikation (p. 669)

Weiterleitung von Anforderungen

Themen

Programme, die Anfragen für Buckets machen, die mit der <CreateBucketConfiguration>-API erstellt wurden, müssen Umleitungen unterstützen. Darüber hinaus entstehen möglicherweise für einige Clients Probleme, die DNS TTLs nicht unterstützen.

Dieser Abschnitt beschreibt Weiterleitungs- und DNS-Aspekte, die Sie beim Entwurf Ihres Service oder Ihrer Anwendung für Amazon S3 berücksichtigen sollten.

Anforderungsumleitung und die REST API

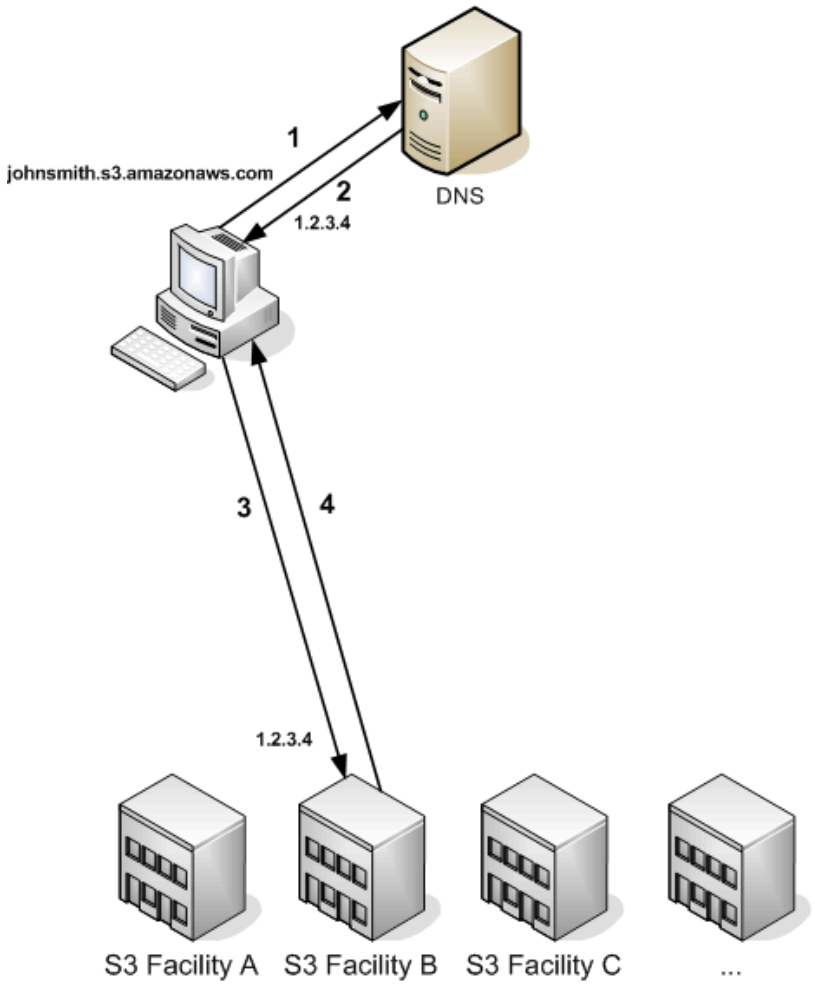
Amazon S3 verwendet das DNS (Domain Name System), um Anfragen an Einrichtungen weiterzuleiten, die sie verarbeiten können. Dieses System arbeitet effektiv, aber es können temporäre Routing-Fehler auftreten. Trifft eine Anfrage am falschen Amazon S3-Standort ein, reagiert Amazon S3 mit einer temporären Umleitung, die den Auftraggeber anweisen, die Anfrage an einen neuen Endpunkt zu senden. Ist eine Anfrage fehlerhaft aufgebaut, verwendet Amazon S3 permanente Umleitungen, um dem Auftraggeber mitzuteilen, wie die Anfrage korrekt ausgeführt wird.

> **Important**
>
> Um diese Funktion nutzen zu können, benötigen Sie eine Anwendung, die Amazon S3-Umleitungsantworten verarbeiten kann. Die einzige Ausnahme bilden Anwendungen, die ausschließlich mit Buckets arbeiten, die ohne `<CreateBucketConfiguration>` erstellt wurden. Weitere Informationen über Standorteinschränkungen finden Sie unter Zugriff auf einen Bucket (p. 60).
> Für alle Regionen, die nach dem 20. März 2019 gestartet wurden, gibt Amazon S3 den Fehler „HTTP 400 Bad Request“ zurück, wenn eine Anforderung an einer falschen Amazon S3-Position empfangen wird.
> Weitere Informationen zum Aktivieren oder Deaktivieren einer AWS-Region siehe AWS-Regionen und -Endpunkte im AWS General Reference.

Themen

DNS-Weiterleitung

Die DNS-Weiterleitung leitet Anfragen an geeignete Amazon S3-Einrichtungen weiter. Die folgende Abbildung und Vorgehensweise zeigen ein Beispiel für DNS-Routing.



DNS-Routing-Anforderungsschritte

1. Der Client stellt eine DNS-Anfrage, um ein Objekt in Amazon S3 speichern zu lassen.
2. Der Client erhält eine oder mehrere IP-Adressen für Einrichtungen, die die Anfrage verarbeiten können. In diesem Beispiel ist die IP-Adresse für die Einrichtung B.
3. Der Client stellt eine Anforderung an Amazon S3-Einrichtung B.
4. Einrichtung B gibt eine Kopie des Objekts an den Kunden zurück.

Temporäre Anforderungsumleitung

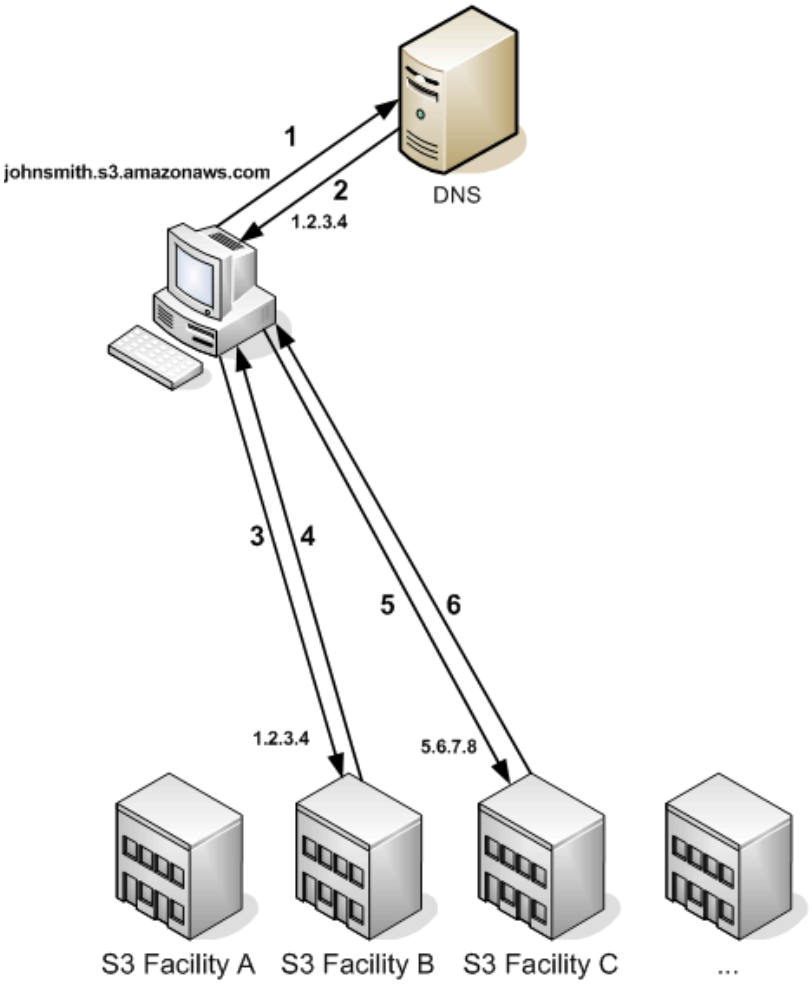
Eine temporäre Umleitung ist eine Art Fehlermeldung, die dem Auftraggeber signalisiert, dass er die Anfrage an einen anderen Endpunkt senden sollte. Aufgrund der verteilten Natur von Amazon S3 können Anfragen temporär in die falsche Einrichtung weitergeleitet werden. Dies tritt am wahrscheinlichsten auf, unmittelbar nachdem Buckets erstellt oder gelöscht wurden.

Wenn Sie beispielsweise einen neuen Bucket erstellen und sofort eine Anfrage für den Bucket machen, werden Sie möglicherweise temporär umgeleitet, abhängig von der Standorteinschränkung des Buckets. Wenn Sie den Bucket in der AWS Region USA Ost (Nord-Virginia) erstellt haben, wird die Umleitung nicht angezeigt, da dies auch der Standard-Amazon S3-Endpunkt ist.

Wenn der Bucket jedoch in einer anderen Region erstellt wird, gehen alle Anfragen für den Bucket an den Standard-Endpunkt, während der DNS-Eintrag des Buckets propagiert wird. Der Standard-Endpunkt leitet die Anfrage mit einer HTTP 302-Antwort an den richtigen Endpunkt weiter. Temporäre Umleitungen enthalten eine URI zur richtigen Einrichtung, die Sie verwenden können, um die Anfrage sofort erneut zu senden.

> **Important**
>
> Verwenden Sie keinen Endpunkt wieder, der aus einer vorherigen Umleitungsantwort stammt. Das scheint zu funktionieren (manchmal sogar sehr lange), kann aber unvorhersehbare Ergebnisse erzeugen und schlägt irgendwann ohne weitere Mitteilung fehl.

Die folgende Abbildung und Vorgehensweise zeigen ein Beispiel für eine temporäre Umleitung.



Temporäre Umleitungsschritte für Anfragen

1. Der Client stellt eine DNS-Anfrage, um ein Objekt in Amazon S3 speichern zu lassen.
2. Der Client erhält eine oder mehrere IP-Adressen für Einrichtungen, die die Anfrage verarbeiten können.
3. Der Client stellt eine Anforderung an Amazon S3-Einrichtung B.
4. Einrichtung B gibt eine Umleitung zurück, die angibt, dass das Objekt an Standort C zur Verfügung steht.
5. Der Client sendet Anfrage noch einmal an Einrichtung C.

6. Einrichtung C gibt eine Kopie des Objekts zurück.

Permanente Anforderungsumleitung

Eine permanente Anfrageumleitung zeigt an, dass Ihre Anfrage eine fehlerhafte Adresse für eine Ressource angegeben hat. Permanente Umleitungen treten beispielsweise auf, wenn Sie eine Anfrage mit Angabe eines Pfads für den Zugriff auf einen Bucket verwenden, der mit `<CreateBucketConfiguration>` erstellt wurde. Weitere Informationen finden Sie unter Zugriff auf einen Bucket (p. 60).

Um diese Fehler bei der Entwicklung zu erkennen, enthält diese Art Umleitung keinen HTTP Location-Header, der Ihnen ermöglicht, die Anfrage automatisch an den richtigen Standort zu verfolgen. Weitere Informationen über die Verwendung des richtigen Amazon S3-Endpunkts finden Sie im resultierenden XML-Fehlerdokument.

Beispiele für Anforderungsumleitung

Im Folgenden finden Sie Beispiele für temporäre Umleitungsantworten.

Temporäre Umleitungsantwort der REST API

```
HTTP/1.1 307 Temporary Redirect
Location: http://awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com/photos/puppy.jpg?rk=e2c69a31
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Fri, 12 Oct 2007 01:12:56 GMT
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the specified temporary endpoint.
  Continue to use the original request endpoint for future requests.</Message>
  <Endpoint>awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com</Endpoint>
</Error>
```

Temporäre Umleitungsantwort der SOAP API

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.TemporaryRedirect</Faultcode>
    <Faultstring>Please re-send this request to the specified temporary endpoint.
    Continue to use the original request endpoint for future requests.</Faultstring>
    <Detail>
      <Bucket>images</Bucket>
      <Endpoint>s3-gztb4pa9sq.amazonaws.com</Endpoint>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

Überlegungen zu DNS

Eine der Entwurfsanforderungen von Amazon S3 ist eine extrem hohe Verfügbarkeit. Unter anderem erfüllen wir diese Anforderung, indem wir die dem Amazon S3-Endpunkt in DNS zugeordnete IP-Adressen nach Bedarf aktualisieren. Diese Änderungen werden in kurzlebigen Clients automatisch reflektiert, nicht jedoch in einigen langlebigen Clients. Für langlebige Clients ist eine spezielle Maßnahme erforderlich, um den Amazon S3-Endpunkt regelmäßig wieder aufzulösen, um von diesen Änderungen zu profitieren. Weitere Informationen zu virtuellen Maschinen (VMs) finden Sie hier:

- Für Java speichert die JVM von Sun die DNS-Suchen standardmäßig auf unbegrenzte Zeit. Informationen zum Ändern dieses Verhaltens finden Sie im Abschnitt „InetAddress Caching“ in der InetAddress-Dokumentation.
- Für PHP speichert die persistente PHP VM, die in den gebräuchlichsten Bereitstellungskonfigurationen ausgeführt wird, DNS-Suchen, bis die VM neu gestartet wird. Weitere Informationen finden Sie in den getHostByName-PHP-Dokumenten.

bewährte Verfahren für Designmuster: Optimierung der Amazon S3-Leistung

Ihre Anwendungen erreichen schnell Tausende von Transaktionen pro Sekunde bei der Anfrageleistung, wenn Speicherinhalte auf Amazon S3 hochgeladen oder von dort abgerufen werden. Amazon S3 wird automatisch auf hohe Anfrageraten skaliert. Ihre Anwendung kann beispielsweise mindestens 3 500 PUT/COPY/POST/DELETE- oder 5 500 GET/HEAD-Anforderungen pro Sekunde pro Präfix in einem Bucket erreichen. Es gibt keine Einschränkungen für die Anzahl der Präfixe in einem Bucket. Sie können Ihre Lese- und Schreibleistung steigern, indem Sie Lesevorgänge parallelisieren. Wenn Sie beispielsweise 10 Präfixe in einem Amazon S3-Bucket für parallele Lesevorgänge einrichten, können Sie damit die Leseleistung auf 55 000 Leseanfragen pro Sekunde skalieren.

So prüfen etwa manche Data-Lake-Anwendungen in Amazon S3 Millionen oder Milliarden von Objekten auf Anfragen, die über Petabytes von Daten ausgeführt werden. Solche Data Lake-Anwendungen erreichen Single-Instance-Übertragungsraten, die die Netzwerkschnittstellennutzung für ihre Amazon EC2-Instance maximieren und bis zu 100 GB/s auf einer einzelnen Instance erreichen können. Anschließend aggregieren diese Anwendungen den Durchsatz über mehrere Instances hinweg, um mehrere Terabit pro Sekunde zu erreichen.

Andere Anwendungen sind latenzempfindlich, wie etwa Social-Media-Messaginganwendungen. Diese Anwendungen können konsistente Small Objekt-Latenzen (und First-byte-out-Latenzen für große Objekte) von etwa 100–200 Millisekunden erreichen.

Andere AWS-Services können auch dabei helfen, die Leistung für unterschiedliche Anwendungsarchitekturen zu beschleunigen. Wenn Sie beispielsweise höhere Übertragungsraten gegenüber einer einzelnen HTTP-Verbindung oder Latenzen im einstelligen Millisekundenbereich wünschen, verwenden Sie Amazon CloudFront oder Amazon ElastiCache für das Caching mit Amazon S3.

Und wenn Sie eine schnelle Datenübertragung über große Distanzen zwischen einem Client und einem S3-Bucket wünschen, verwenden Sie Amazon S3 Transfer Acceleration (p. 75). Die Übertragungsbeschleunigung verwendet die weltweit verteilten Edge-Standorte in CloudFront zur Beschleunigung des Datentransports über geographische Entfernungen hinweg. Wenn Ihr Workload in Amazon S3 eine serverseitige Verschlüsselung mit AWS Key Management Service (SSE-KMS) verwendet, finden Sie unter Limits für AWS KMS im AWS Key Management Service-Entwicklerhandbuch weitere Informationen zu den für Ihren Anwendungsfall unterstützten Anfrageraten.

Die folgenden Themen beschreiben bewährte Verfahren und Designmuster zur Optimierung der Leistung von Anwendungen, die Amazon S3 verwenden. Diese Anleitungen ersetzen alle vorherigen Anleitungen zur Optimierung der Leistung für Amazon S3. Zum Beispiel empfahlen Amazon S3-Leistungsanleitungen früher, die Präfixbenennung mit gehashten Zeichen zu randomisieren, um die Leistung für häufige Datenabrufe zu optimieren. Sie müssen Präfixbenennungen nicht mehr für Leistung randomisieren und können sequenzielle datumsbasierte Benennungen für Ihre Präfixe verwenden. Die neuesten Informationen zur Leistungsoptimierung für Amazon S3 finden Sie unter Leistungsanleitungen für Amazon S3 (p. 732) und Leistungsdesignmuster für Amazon S3 (p. 734).

Themen

Leistungsanleitungen für Amazon S3

Bei der Erstellung von Anwendungen, die Objekte zu Amazon S3 hochladen und davon abrufen, sollten Sie unsere bewährten Methoden befolgen, um die Leistung zu optimieren. Wir bieten auch detailliertere Leistungsdesignmuster (p. 734).

Zur Erzielung der besten Leistung für Ihre Anwendung auf Amazon S3 empfehlen wir die folgenden Vorgehensweisen.

Themen

Messen der Leistung

Betrachten Sie für die Optimierung der Leistung den Netzwerkdurchsatz sowie die CPU- und DRAM-Anforderungen. Ja nach der Mischung der Anforderungen für diese verschiedenen Ressourcen kann es sinnvoll sein, verschiedene Amazon EC2-Instance-Typen zu erwägen. Weitere Informationen zu diesen Instance-Typen finden Sie unter Instance-Typen im Amazon EC2-Benutzerhandbuch für Linux-Instances.

Weiterhin ist es für die Messung der Leistung sinnvoll, die DNS-Lookup-Zeit, die Latenz und die Datenübertragungsgeschwindigkeit mithilfe von HTTP-Analysetools zu untersuchen.

Horizontale Skalierung von Speicherverbindungen

Die Verteilung von Anfragen über zahlreiche Verbindungen ist ein übliches Designmuster für die horizontale Skalierung der Leistung. Wenn Sie hochleistungsfähige Anwendungen erstellen, stellen Sie sich Amazon S3 als sehr großes verteiltes System vor, nicht als einen einzelnen Netzwerkendpunkt wie ein herkömmlicher Speicherserver. Sie erreichen die beste Leistung durch die Ausgabe mehrerer gleichzeitiger Anfragen an Amazon S3. Verteilen Sie diese Anfragen über separate Verbindungen, um die zugängliche Bandbreite von Amazon S3 zu maximieren. Amazon S3 hat keine Begrenzungen für die Anzahl der zu Ihrem Bucket hergestellten Verbindungen.

Verwenden von Byte Range Fetches

Mit dem `Range`-HTTP-Header in einer GET Object-Anforderung können Sie einen Bytebereich von einem Objekt abrufen, wobei nur der angegebene Teil übertragen wird. Sie können gleichzeitige Verbindungen zu Amazon S3 verwenden, um verschiedene Bytebereiche aus demselben Objekt abzurufen. Dies hilft beim Erreichen eines höheren aggregierten Durchsatzes als bei einer einzelnen Anforderung eines ganzen Objekts. Der Abruf kleinerer Bereiche eines größeren Objekts ermöglicht Ihrer Anwendung auch die Verbesserung der Zeiten für Wiederholungsversuche, wenn Anforderungen unterbrochen werden. Weitere Informationen finden Sie unter Objekte abrufen (p. 184).

Typische Größen für Bytebereichanforderungen sind 8 oder 16 MB. Wenn für Objekte eine PUT-Aktion unter Verwendung eines mehrteiligen Uploads durchgeführt werden, ist es sinnvoll, die GET-Aktion mit

denselben Teilgrößen (oder zumindest orientiert an den Teilgrenzen) durchzuführen, um eine optimale Leistung zu erzielen. GET-Anforderungen können sich direkt auf einzelne Teile richten, zum Beispiel `GET ?partNumber=N.`.

Wiederholungsanforderungen für latenzsensitive Anwendungen

Aggressive Timeouts und Wiederholungsversuche sorgen für konsistente Latenz. Aufgrund des Umfangs von Amazon S3 gilt: Wenn die erste Anforderung langsam ist, nimmt eine alte Anfrage wahrscheinlich einen anderen Pfad und ist schnell erfolgreich. Die AWS SDKs verfügen über konfigurierbare Timeout- und Wiederholungsversuchwerte, die Sie an die Toleranzen Ihrer spezifischen Anwendung anpassen können.

Kombinieren von Amazon S3 (Speicherung) und Amazon EC2 (Computing) in derselben AWS-Region

Obwohl Namen von S3-Buckets global eindeutig sind, wird jeder Bucket in einer Region gespeichert, die Sie bei dessen Erstellung auswählen. Zur Optimierung der Leistung empfehlen wir, nach Möglichkeit auf den Bucket von Amazon EC2-Instances in derselben region aus zuzugreifen. Dies hilft bei der Reduzierung der Netzwerklatenz und der Datenübertragungskosten.

Weitere Informationen über die Kosten von Datenübertragungen finden Sie unter Amazon S3 – Preise.

Verwenden der Amazon S3 Transfer Acceleration zur Minimierung der durch die Entfernung verursachten Latenz

Amazon S3 Transfer Acceleration (p. 75) ermöglicht die schnelle, einfache und sichere Übertragung von Dateien über größere Entfernungen zwischen Ihrem Client und einem S3-Bucket. Transfer Acceleration nutzt die weltweit verteilten Edge-Standorte von Amazon CloudFront. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Amazon S3 weitergeleitet. Transfer Acceleration ist ideal für die regelmäßige Übertragung von Daten im Gigabyte- bis Terabyte-Bereich von Kontinent zu Kontinent geeignet. Die Funktion ist auch für Kunden nützlich, die Uploads in einen zentralen Bucket aus der ganzen Welt vornehmen.

Sie können das Amazon S3 Transfer Acceleration Speed Comparison Tool verwenden, um beschleunigte und nicht beschleunigte Upload-Geschwindigkeiten über Amazon S3-Regionen hinweg zu vergleichen. Das Speed Comparison-Tool verwendet mehrteilige Uploads, um eine Datei von Ihrem Browser in verschiedene Amazon S3-Regionen mit und ohne die Verwendung der Amazon S3 Transfer Acceleration zu übertragen.

Verwendung der neuesten Version der AWS SDKs

Die AWS SDKs bieten integrierte Unterstützung für viele Empfehlungen zur Optimierung der Amazon S3-Leistung. Die SDKs bieten eine einfachere API zur Nutzung von Amazon S3 aus einer Anwendung heraus und werden regelmäßig nach bewährten verfahren aktualisiert. Beispielsweise enthalten die SDKs Logik für die automatische Wiederholung von Anforderungen bei HTTP 503-Fehlern und investieren in Code zur Reaktion auf langsame Verbindungen.

Die SDKs bieten dazu den Transfer Manager, der die horizontale Skalierung von Verbindungen automatisiert, um, wo möglich mithilfe von Bytebereichanforderungen, Tausende von Anforderungen pro Sekunde zu erreichen. Es ist wichtig, immer die neueste Version der AWS SDKs zu verwenden, um stets die aktuellsten Leistungsoptimierungsfunktionen nutzen zu können.

Sie können auch die Leistung optimieren, wenn Sie HTTP REST API-Anforderungen verwenden. Bei der Verwendung der REST API sollten Sie ebenfalls die bewährten Verfahren verwenden, die Teil der SDKs sind. Lassen Sie bei langsamen Anforderungen Timeouts und Wiederholungsversuche zu, sowie mehrere Verbindung, um den parallelen Abruf von Objektdaten zu ermöglichen. Weitere Informationen zur Verwendung der REST-API finden Sie unter Amazon Simple Storage Service API Reference.

Leistungsdesignmuster für Amazon S3

Beim Entwurf von Anwendungen zum Upload und Abruf von Objekten von Amazon S3 sollten Sie unsere bewährten Designmuster verwenden, um die optimale Leistung für Ihre Anwendung zu erzielen. Dazu bieten wir Ihnen Leistungsanleitungen (p. 732) für Überlegungen zur Planung Ihrer Anwendungsarchitektur an.

Zur Optimierung der Leistung können Sie die folgenden Designmuster verwenden.

Themen

Verwendung von Caching für Inhalte mit häufigen Zugriffen

Viele Anwendungen, die Daten in Amazon S3 speichern, stellen einen „Arbeitssatz“ der Daten bereit, der von Benutzern häufig angefragt wird. Wenn ein Workload wiederholt GET-Anforderungen für einen Objektsatz sendet, können Sie zur Optimierung der Leistung einen Cache wie Amazon CloudFront, Amazon ElastiCache oder AWS Elemental MediaStore verwenden. Die erfolgreiche Cache-Nutzung kann zu niedriger Latenz und zu hohen Datenübertragungsraten führen. Anwendungen, die Caching verwenden, senden auch weniger direkte Anforderungen an Amazon S3, was zur Senkung der Anfragenkosten beitragen kann.

Amazon CloudFront ist ein schnelles CDN (Content Delivery Network), das Daten von Amazon S3 in transparenter Weise an zahlreichen geographisch verteilten PoPs (Points of Presence) zwischenspeichert. Wenn Objekte aus mehreren Regionen oder über das Internet zugänglich sein sollen, ermöglicht CloudFront die Zwischenspeicherung der Daten in der Nähe der Benutzer, die auf die Objekte zugreifen. Dies kann zu hohen Leistungen bei der Bereitstellung beliebter Amazon S3-Inhalte führen. Weitere Informationen zu CloudFront finden Sie im .Entwicklerhandbuch für Amazon CloudFront.

Amazon ElastiCache ist ein verwalteter In-Memory-Cache. Mit ElastiCache können Sie Amazon EC2-Instances bereitstellen, die Objekte im Arbeitsspeicher zwischenspeichern. Dieses Caching führt zur Reduzierung der GET-Latenz im Bereich mehrerer Größenordnungen und zu einer erheblichen Zunahme des Downloaddurchsatzes. Zur Verwendung von ElastiCache modifizieren Sie die Anwendungslogik so, dass der Zwischenspeicher mit aktiven Objekten gefüllt wird, und dass er auf solche Objekte überprüft wird, bevor diese von Amazon S3 angefordert werden. Beispiele für die Verwendung von ElastiCache zur Verbesserung der Amazon S3-GET-Leistung finden Sie im Blogartikel TurbochargeAmazon S3 mit Amazon ElastiCache für Redis.

AWS Elemental MediaStore ist ein Caching- und Content Distribution-System, das speziell für Video-Workflows und die Medienbereitstellung von Amazon S3 entwickelt wurde. MediaStore bietet durchgehende Speicher-APIs speziell für Video und wird für leistungssensitive Video-Workloads

empfohlen. Weitere Informationen zu MediaStore finden Sie im AWS Elemental MediaStore-Benutzerhandbuch.

Timeouts und Wiederholungsversuche für latenzsensitive Anwendungen

Es gibt bestimmte Situationen, in denen eine Anwendung eine Antwort von Amazon S3 erhält, die besagt, dass ein Wiederholungsversuch erforderlich ist. Amazon S3 gleicht Bucket- und Objektnamen mit den diesen zugeordneten Objektdaten ab. Wenn eine Anwendung hohe Anforderungsraten generiert (typischerweise dauerhaft über 5.000 Anforderungen pro Sekunde für eine kleine Zahl von Objekten), erhält sie möglicherweise HTTP 503 Slowdown-Antworten. Wenn solche Fehler auftreten, implementiert jedes AWS SDK eine automatische Wiederholungsversuchlogik mit exponentiellem Backoff. Wenn Sie kein AWS SDK verwenden, sollten Sie eine Wiederholungsversuchlogik implementieren, wenn Sie den HTTP-Fehler 503 erhalten. Für Informationen zu Backoff-Techniken vgl. Wiederholungsversuche bei Fehlern und exponentielles Backoff in AWS im Allgemeine Amazon Web Services-Referenz.

Amazon S3 wird automatisch in Reaktion auf andauernde neue Anforderungsraten skaliert und optimiert so die Leistung in dynamischer Weise. Während Amazon S3 Optimierungen für eine neue Anforderungsrate durchführt, erhalten Sie temporär HTTP 503-Anforderungsantworten, bis die Optimierung abgeschlossen ist. Nachdem Amazon S3 die Leistung intern für die neue Anforderungsrate optimiert hat, werden alle Anforderungen generell ohne Wiederholungsversuche bereitgestellt.

Für latenzsensitive Anwendungen empfiehlt Amazon S3 Nachverfolgung und aggressive Wiederholungsversuche bei langsameren Operationen. Wenn Sie eine Anforderung wiederholen, empfehlen wir die Verwendung einer neuen Verbindung zu Amazon S3 und die Durchführung eines neuen DNS-Lookup-Vorgangs.

Wenn Sie sehr große Anforderungen unterschiedlicher Größe (beispielsweise mit mehr als 128 MB) durchführen, sollten Sie den erreichten Durchsatz nachverfolgen und für die langsamsten 5 Prozent der Anforderungen Wiederholungsversuche durchführen. Wenn Sie kleinere Anforderungen (etwa unter 512 KB) durchführen, bei denen die mittleren Latenzen oft im zweistelligen Millisekundenbereich liegen, ist es sinnvoll, nach zwei Sekunden eine GET- oder PUT-Operation zu wiederholen. Wenn weitere Wiederholungsversuche erforderlich sind, ist ein Backoff die beste Lösung. So empfehlen wir beispielsweise die Ausgabe eines Wiederholungsversuchs nach zwei Sekunden und eines zweiten Versuchs nach weiteren vier Sekunden.

Wenn Ihre Anwendung Anforderungen mit fester Größe an Amazon S3 sendet, sollten Sie konsistentere Reaktionszeiten für diese einzelnen Anforderungen erwarten. In diesem Fall ist es eine einfache Strategie, das langsamste Prozent der Anforderungen zu identifizieren und diese zu wiederholen. Selbst ein einziger Wiederholungsversuch ist oft erfolgreich für die Reduzierung der Latenz.

Wenn Sie AWS Key Management Service (AWS KMS) für die serverseitige Verschlüsselung verwenden, vgl. Limits im AWS Key Management Service Developer Guide für Informationen zu den Anforderungsraten, die für Ihren Anwendungsfall unterstützt werden.

Horizontale Skalierung und Anforderungsparallelisierung für hohen Durchsatz

Amazon S3 ist ein sehr großes verteiltes System. Um diese Größe zu nutzen, sollten Sie parallele Anforderungen horizontal zu den Amazon S3-Serviceendpunkten skalieren. Zusätzlich zur Verteilung der Anforderungen in Amazon S3 hilft dieses Skalierungskonzept dabei, die Last über mehrere Pfade im Netzwerk zu verteilen.

Für Übertragungen mit hohem Durchsatz empfiehlt Amazon S3 die Verwendung von Anwendungen mit mehreren Verbindungen für die Parallele Ausführung von GET- und PUT-Aktionen. Dies wird

beispielsweise von Amazon S3Transfer Manager im AWS Java SDK unterstützt, und die meisten anderen AWS SDKs bieten ähnliche Konstrukte. Für manche Anwendungen können Sie parallele Verbindungen dadurch erreichen, dass Sie Anforderungen gleichzeitig in verschiedenen Anwendungsthreads oder in verschiedenen Anwendungsinstances starten. Das beste Konzept hängt von Ihrer Anwendung und der Struktur der Objekte ab, auf die Sie zugreifen.

Sie können die AWS SDKs verwenden, um GET- und PUT-Anforderungen direkt auszugeben, anstatt die Verwaltung von Übertragungen in dem AWS SDK zu nutzen. Dieses Konzept ermöglicht die direktere Abstimmung Ihrer Workloads, bei gleichzeitiger Nutzung der SDK-Unterstützung für Wiederholungsversuche und den Umgang mit eventuell auftretenden HTTP 503-Antworten. Generell gilt: Wenn Sie in einer Region große Objekte von Amazon S3 zu Amazon EC2 herunterladen, sollten Sie gleichzeitige Anforderungen für Bytebereiche eines Objekts in der Größe von 8–16 MB starten. Starten Sie eine gleichzeitige Anforderung für jeweils 85–90 MB/s des gewünschten Netzwerkdurchsatzes. Zur Ausnutzung einer Netzwerkschnittstellenkarte (NIC) mit 10 Gb/s können Sie etwa 15 gleichzeitige Anforderungen über separate Verbindungen durchführen. Sie können die gleichzeitigen Anforderungen über mehr Verbindungen hochfahren, um schnellere NICs zu nutzen, etwa 25 oder 100 Gb/s.

Die Messung der Leistung ist wichtig für die Einstellung der Anzahl der gleichzeitig auszugebenden Anforderungen. Wir empfehlen, mit jeweils einer einzigen Anforderung zu beginnen. Messen Sie die erreichte Netzwerkbandbreite und die Nutzung weiterer Ressourcen durch Ihre Anwendung bei der Verarbeitung der Daten. Sie können dann die Engpassressource (die Ressource mit der höchsten Nutzung) identifizieren und so die Anzahl der Anforderungen ermitteln, die wahrscheinlich nützlich waren. Zum Beispiel: Wenn die Verarbeitung einzelner Anforderungen zu einer CPU-Nutzung von 25 Prozent führt, zeigt dies an, dass bis zu vier gleichzeitige Anforderungen möglich sind. Messungen sind sehr wichtig, und es lohnt sich, die Ressourcennutzung zu ermitteln, wenn die Anforderungsrate erhöht wird.

Wenn ihre Anwendungen Anforderungen direkt an Amazon S3 mit der REST API ausgibt, sollten Sie einen Pool von HTTP-Verbindungen verwenden und jede Verbindung für eine Serie von Anforderungen wiederverwenden. Die Vermeidung der Einrichtung von Verbindungen für jede Anforderung macht TCP-Slow-Starts und SSL-Handshakes bei jeder Anforderung überflüssig. Weitere Informationen zur Verwendung der REST-API finden Sie unter Amazon Simple Storage Service API Reference.

Schließlich ist es auch sinnvoll, auf den DNS zu achten und genau zu prüfen, ob Anforderungen über einen großen Pool von Amazon S3-IP-Adressen verteilt werden. DNS-Anforderungen für Amazon S3 durchlaufen eine lange Liste von IP-Endpunkten. Das Caching von Resolvern oder Anwendungscode, der eine einzelne IP-Adresse wiederverwendet, profitiert nicht von der Adressendiversität und dem daraus resultierenden Lastenausgleich. Network Utility-Tools wie das `netstat`-Befehlszeilentool können die IP-Adressen anzeigen, die für die Kommunikation mit Amazon S3 verwendet werden, und wir bieten Anleitungen zu den zu verwendenden DNS-Konfigurationen. Weitere Informationen zu diesen Anleitungen finden Sie unter Überlegungen zu DNS (p. 730).

Verwendung von Amazon S3 Transfer Acceleration zur Beschleunigung geographisch disparater Datenübertragungen

Amazon S3 Transfer Acceleration (p. 75) ist effektiv bei der Minimierung oder Beseitigung der durch geographische Entfernung zwischen global verteilten Clients und einer regionalen Anwendung mit Amazon S3 verursachten Latenz. Transfer Acceleration nutzt die weltweit verteilten Edge-Standorte in CloudFront für den Datentransport. Das AWS Edge-Netzwerk verfügt über Präsenzpunkte an mehr als 50 Standorten. Heute wird es für die Verteilung von Inhalten über CloudFront und für die Bereitstellung schneller Antworten auf DNS-Anfragen an Amazon Route 53 verwendet.

Dazu hilft das Edge-Netzwerk bei der Beschleunigung von Datenübertragungen zu und aus Amazon S3. Es ist ideal für Anwendungen, die Daten über oder zwischen Kontinenten übertragen, eine schnelle Internetverbindung nutzen, große Objekte verwenden oder zahlreiche Inhalte hochladen müssen. Sobald die Daten an einem Edge-Standort eingehen, werden sie über einen optimierten Netzwerkpfad an Ihren

Amazon S3 -Bucket weitergeleitet. Allgemein gilt: Je weiter Sie von einer Amazon S3-Region entfernt sind, um so größer ist die Geschwindigkeitsverbesserung, die Sie von Transfer Acceleration erwarten können.

Sie können Transfer Acceleration auf neuen oder bestehenden Buckets einrichten. Sie können einen separaten Amazon S3 Transfer Acceleration-Endpunkt verwenden, um AWS-Edge-Standorte zu nutzen. Die beste Möglichkeit zur Prüfung, ob Transfer Acceleration die Client-Leistung unterstützt, ist die Verwendung des Amazon S3 Transfer Acceleration Speed Comparison-Tools. Netzwerkkonfigurationen und Bedingungen variieren von zeit zu Zeit und von Standort zu Standort. Sie werden daher nur für Übertragungen belastet, bei denen Amazon S3 Transfer Acceleration Ihre Uploadleistung potentiell verbessern kann. Informationen zur Verwendung von Transfer Acceleration mit verschiedenen AWS SDKs finden Sie unter Beispiele für Amazon S3 Transfer Acceleration (p. 78).

Überwachung von Amazon S3

Überwachung ist wichtig, um die Zuverlässigkeit, Verfügbarkeit und Performance von Amazon S3 und der AWS-Lösungen aufrechtzuerhalten. Sie sollten Überwachungsdaten aus allen Teilen der AWS-Lösung erfassen, damit Sie Ausfälle, die sich über mehrere Punkte erstrecken, leichter debuggen können. Bevor Sie mit der Überwachung von Amazon S3 beginnen, sollten Sie einen Überwachungsplan mit Antworten auf die folgenden Fragen erstellen:

- Was sind Ihre Ziele bei der Überwachung?
- Welche Ressourcen werden überwacht?
- Wie oft werden diese Ressourcen überwacht?
- Welche Überwachungstools werden verwendet?
- Wer soll die Überwachungsaufgaben ausführen?
- Wer soll benachrichtigt werden, wenn Fehler auftreten?

Themen

Überwachungstools

AWS bietet verschiedene Tools, mit deren Hilfe Sie Amazon S3 überwachen können. Sie können einige dieser Tools so konfigurieren, dass diese die Überwachung für Sie übernehmen, während bei anderen Tools ein manuelles Eingreifen nötig ist. Wir empfehlen, dass Sie die Überwachungsaufgaben möglichst automatisieren.

Automatisierte Überwachungstools

Sie können die folgenden automatisierten Tools zur Überwachung von Amazon S3 verwenden und möglicherweise auftretende Probleme melden:

- Amazon CloudWatch-Alarme: Überwachen Sie eine Metrik über einen bestimmten, von Ihnen definierten Zeitraum und führen Sie einzelne oder mehrere Aktionen durch, die vom Wert der Metrik im Vergleich zu einem festgelegten Schwellenwert in einer Reihe von Zeiträumen abhängen. Die Aktion ist eine Benachrichtigung, die an ein Amazon Simple Notification Service (Amazon SNS)-Thema oder eine Amazon EC2 Auto Scaling-Richtlinie gesendet wird. CloudWatch-Alarme rufen keine Aktionen auf, weil sie einen besonderen Status haben. Der Status muss sich geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).
- AWS CloudTrail Log Monitoring – Teilen Sie Protokolldateien zwischen Konten, überwachen Sie CloudTrail-Protokolldateien in Echtzeit, indem Sie sie an CloudWatch Logs senden, schreiben Sie Anwendungen zur Protokollverarbeitung in Java und vergewissern Sie sich, dass nach der Lieferung durch CloudTrail keine Änderungen an den Protokolldaten vorgenommen wurden. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail (p. 750).

Manuelle Überwachungstools

Ein weiterer wichtiger Bestandteil der Überwachung von Amazon S3 ist die manuelle Überwachung derjenigen Elemente, die die CloudWatch-Alarme nicht abdecken. Die Dashboards von Amazon S3, CloudWatch und Trusted Advisor sowie andere AWS Management Console-Dashboards stellen leicht Übersichten zum Zustand der AWS-Umgebung bereit. Möglicherweise wollen Sie die Server-Zugriffsprotokollierung aktivieren, die Zugriffsanforderungen für den Bucket verfolgt. Jeder Zugriffsprotokolldatensatz enthält Details über eine Zugriffsanforderung, z. B. Auftraggeber, Bucket-Name, Anforderungszeit, Anforderungsaktion, Antwortstatus und Fehlercode, falls vorhanden. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777) im Entwicklerhandbuch für Amazon Simple Storage Service.

- Amazon S3-Dashboard zeigt folgende Informationen an:
  - Ihre Buckets und die Eigenschaften und Objekte, die sie enthalten.
- CloudWatch-Homepage zeigt folgende Informationen an:
  - Aktuelle Alarme und Status.
  - Diagramme mit Alarmen und Ressourcen.
  - Servicestatus.

  Zusätzlich können Sie CloudWatch für folgende Aufgaben nutzen:
  - Erstellen angepasster Dashboards zur Überwachung der gewünschten Services.
  - Aufzeichnen von Metrikdaten, um Probleme zu beheben und Trends zu erkennen.
  - Durchsuchen aller AWS-Ressourcenmetriken.
  - Erstellen und Bearbeiten von Alarmen, um über Probleme benachrichtigt zu werden.
- Mithilfe von AWS Trusted Advisor können Sie Ihre AWS-Ressourcen überwachen, um Leistung, Zuverlässigkeit, Sicherheit und Kosteneffizienz zu verbessern. Vier Trusted Advisor-Prüfungen sind für alle Benutzer verfügbar; mehr als 50 Prüfungen sind für Benutzer mit einem Business- oder Enterprise-Support-Plan verfügbar. Weitere Informationen finden Sie unter AWS Trusted Advisor.

  Trusted Advisor führt folgende Prüfungen im Zusammenhang mit Amazon S3 aus:
  - Prüfungen der Protokollierungskonfiguration von Amazon S3-Buckets.
  - Sicherheitsprüfungen für Amazon S3-Buckets mit offenen Zugriffsberechtigungen.
  - Fehlertoleranzprüfungen für Amazon S3-Buckets, für die kein Versioning aktiviert oder deren Versioning ausgesetzt ist.

Überwachung von Metriken mit Amazon CloudWatch

Amazon CloudWatch-Metriken für Amazon S3 können Ihnen helfen, die Leistung von Anwendungen, die Amazon S3 verwenden, zu verstehen und zu verbessern. Es gibt mehrere Möglichkeiten für die Verwendung von CloudWatch mit Amazon S3.

- Tägliche Speichermetriken für Buckets - Überwachen Sie den bucket-Speicher mit CloudWatch, um Speicherdaten aus Amazon S3 zu erfassen und zu lesbaren täglichen Metriken zu verarbeiten. Diese Speichermetriken für Amazon S3 werden einmal pro Tag gemeldet und allen Kunden ohne zusätzliche Kosten zur Verfügung gestellt.
- Anforderungsmetriken - Sie können Amazon S3-Anforderungen überwachen, um Probleme bei der Ausführung schnell zu identifizieren und zu beheben. Die Metriken stehen in 1-Minuten-Intervallen nach einer gewissen Latenz für die Verarbeitung zur Verfügung. Diese CloudWatch-Metriken werden zum selben Preis wie benutzerdefinierte Amazon CloudWatch-Metriken berechnet. Informationen zu

CloudWatch-Preisen erhalten Sie unter Amazon CloudWatch – Preise. Weitere Informationen zum optionalen Erhalt dieser Metriken finden Sie unter Metrikkonfigurationen für Buckets (p. 747).

Wenn die Anforderungsmetriken aktiviert sind, werden sie für alle Objektoperationen gemeldet. Standardmäßig stehen diese 1-Minuten-Metriken auf Ebene der Amazon S3-Buckets zur Verfügung. Sie können auch einen Filter für die Metriken definieren, die mit einem gemeinsamen Präfix oder Objekt-Tag erfasst wurden. Sie können also Metrikfilter an bestimmte Geschäftsanwendungen, Workflows oder interne Organisationen anpassen.

- Replikationsmetriken - Überwachen Sie die Gesamtzahl und Größe der Objekte, die zur Replikation ausstehen, sowie die maximale Replikationszeit in die Zielregion. Nur Replikationsregeln, für die S3 Replication Time Control (S3 RTC) aktiviert ist, veröffentlichen Replikationsmetriken.

  Im Gegensatz zu CloudWatch-Speicher- und -Anforderungsmetriken können Sie Replikationsmetriken nicht nach Präfixen und Tags filtern. Sie können jedoch eine Replikationsregel basierend auf Präfixen und Tags einrichten. Ihre Replikationsmetriken überwachen dann die Replikation für die von Ihnen angegebenen Präfixe und Tags. Weitere Informationen finden Sie unter the section called “S3 Replication Time Control (S3 RTC)” (p. 687).

Alle CloudWatch-Statistiken werden für einen Zeitraum von 15 Monaten vorgehalten, damit Sie auf Verlaufsinformationen zugreifen besser feststellen können, wie die Webanwendung oder der Service ausgeführt wird. Weitere Informationen finden Sie unter Was ist Amazon CloudWatch? im Amazon CloudWatch-Benutzerhandbuch.

Metriken und Dimensionen

Die Speichermetriken und -dimensionen, die Amazon S3 an CloudWatch sendet, sind unten aufgelistet.

Tägliche Amazon S3-CloudWatch-Speichermetrik für Buckets

Der `AWS/S3`-Namespace enthält die folgenden täglichen Speichermetriken für Buckets.

Metrik	Beschreibung
`BucketSizeBytes`	Datenmenge in Byte, die in einem Bucket in den Speicherklassen STANDARD, INTELLIGENT_TIERING, Standard – Infrequent Access (STANDARD_IA), OneZone – Infrequent Access (ONEZONE_IA), Reduced Redundancy Storage (RRS), Deep Archive Storage (S3 Glacier Deep Archive) oder Glacier (GLACIER) gespeichert ist. Zur Berechnung dieses Werts wird die Größe aller (aktuellen und nicht aktuellen) Objekte im Bucket summiert – einschließlich der Größe aller Teile für sämtliche unvollständige mehrteilige Uploads in den Bucket. Gültige Speichertypfilter: `StandardStorage`, `IntelligentTieringStorage`, `StandardIAStorage`, `StandardIASizeOverhead`, `StandardIAObjectOverhead`, `OneZoneIAStorage`, `OneZoneIASizeOverhead`, `ReducedRedundancyStorage`, `GlacierStorage`, `GlacierStagingStorage`, `GlacierObjectOverhead`, `GlacierS3ObjectOverhead`, `DeepArchiveStorage`, `DeepArchiveObjectOverhead`, `DeepArchiveS3ObjectOverhead` und `DeepArchiveStagingStorage` (siehe `StorageType`-Dimension) Einheiten: Byte

Metrik	Beschreibung
	Gültige Statistiken: Durchschnitt
`NumberOfObjects`	Die Gesamtanzahl von Objekten, die in einem Bucket für alle Speicherklassen gespeichert sind (mit Ausnahme von GLACIER). Zur Berechnung dieses Werts werden alle aktuellen und nicht aktuellen Objekte im Bucket sowie die Gesamtanzahl der Teile sämtlicher unvollständiger mehrteiliger Uploads in den Bucket gezählt. Gültige Speichertypfilter: nur `AllStorageTypes` (siehe `StorageType`-Dimension) Einheiten: Anzahl Gültige Statistiken: Durchschnitt

Amazon S3-CloudWatch-Anforderungsmetriken

Der `AWS/S3`-Namespace enthält die folgenden Anforderungsmetriken.

Metrik	Beschreibung
`AllRequests`	Die Gesamtanzahl von HTTP-Anfragen an einen Amazon S3-Bucket, unabhängig vom Typ. Wenn Sie eine Metrikkonfiguration mit einem Filter verwenden, gibt diese Metrik nur die HTTP-Anfragen an die Bucket-Objekte zurück, die den Filteranforderungen entsprechen. Einheiten: Anzahl Gültige Statistiken: Summe
`GetRequests`	Die Gesamtanzahl von HTTP GET-Anforderungen an Objekte in einem Amazon S3-Bucket. Das Auflisten von Vorgängen ist hierin nicht inbegriffen. Einheiten: Anzahl Gültige Statistiken: Summe **Note** Paginierte, listenorientierte Anforderungen, wie z. B. List Multipart Uploads, List Parts, Get Bucket Object Versions u. a., sind nicht in dieser Metrik enthalten.
`PutRequests`	Die Gesamtanzahl von HTTP PUT-Anforderungen an Objekte in einem Amazon S3-Bucket. Einheiten: Anzahl Gültige Statistiken: Summe
`DeleteRequests`	Die Gesamtanzahl von HTTP DELETE-Anforderungen an Objekte in einem Amazon S3-Bucket. Dies umfasst auch Delete Multiple Objects-Anforderungen. Diese Metrik zeigt die Anzahl der Anfragen, nicht die Anzahl der gelöschten Objekte an. Einheiten: Anzahl

Metrik	Beschreibung
	Gültige Statistiken: Summe
`HeadRequests`	Die Gesamtanzahl von HTTP HEAD-Anfragen an ein Amazon S3-Bucket. Einheiten: Anzahl Gültige Statistiken: Summe
`PostRequests`	Die Gesamtanzahl von HTTP POST-Anfragen an ein Amazon S3-Bucket. Einheiten: Anzahl Gültige Statistiken: Summe Note Die Anfragen Delete Multiple Objects und SELECT Object Content sind nicht in dieser Metrik enthalten.
`SelectRequests`	Die Gesamtanzahl der Amazon S3-SELECT Object Content-Anfragen, die für Objekte in einem Amazon S3-Bucket gestellt wurden. Einheiten: Anzahl Gültige Statistiken: Summe
`SelectScannedBytes`	Die Gesamtanzahl der Bytes von mit Amazon S3-SELECT Object Content-Anfragen in einem Amazon S3-Bucket gescannten Daten. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`SelectReturnedBytes`	Die Gesamtanzahl der Bytes von mit Amazon S3-SELECT Object Content-Anfragen in einem Amazon S3-Bucket zurückgegebenen Daten. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`ListRequests`	Die Gesamtanzahl von HTTP-Anforderungen zum Auflisten der Inhalte eines Buckets. Einheiten: Anzahl Gültige Statistiken: Summe
`BytesDownloaded`	Anzahl der heruntergeladenen Bytes für Anfragen an einen Amazon S3-Bucket, wobei die Antwort einen Textkörper enthält. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9

Metrik	Beschreibung
`BytesUploaded`	Anzahl von Bytes mit einem Anfragetextkörper, die in einen Amazon S3-Bucket hochgeladen wurden. Einheiten: Byte Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min, Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`4xxErrors`	Anzahl der HTTP-4xx-Client-Fehlerstatuscode-Anfragen an einen Amazon S3-Bucket mit einem Wert von 0 oder 1. Die `average`-Statistik zeigt die Fehlerrate und die `sum`-Statistik die Anzahl dieses Fehlertyps für den jeweiligen Zeitraum an. Einheiten: Anzahl Gültige Statistiken: Durchschnitt (Berichte pro Anforderung), Summe (Berichte pro Zeitraum), Min, Max, Stichprobenanzahl
`5xxErrors`	Anzahl der HTTP-5xx-Client-Fehlerstatuscode-Anfragen an einen Amazon S3-Bucket mit einem Wert von 0 oder 1. Die `average`-Statistik zeigt die Fehlerrate und die `sum`-Statistik die Anzahl dieses Fehlertyps für den jeweiligen Zeitraum an. Einheiten: Anzahl Gültige Statistiken: Durchschnitt (Berichte pro Anforderung), Summe (Berichte pro Zeitraum), Min, Max, Stichprobenanzahl
`FirstByteLatency`	Zeit pro Anforderung vom Zeitpunkt des Eingangs der vollständigen Anfragen bei einem Amazon S3-Bucket bis das Zurückgeben der Antwort beginnt. Einheiten: Millisekunden Gültige Statistiken: Durchschnitt, Summe, Min, Max (entspricht p100), Stichprobenanzahl, beliebiges Perzentil zwischen p0,0 und p100
`TotalRequestLatency`	Zeit pro Anforderung vom Zeitpunkt des Eingangs der vollständigen Anfragen bei einem Amazon S3-Bucket bis zum Beginn der Rückgabe der Antwort. Dies umfasst die erforderliche Zeit, um den Textkörper zu erhalten und den Antworttextkörper zu senden, welche nicht in `FirstByteLatency` enthalten ist. Einheiten: Millisekunden Gültige Statistiken: Durchschnitt, Summe, Min, Max (entspricht p100), Stichprobenanzahl, beliebiges Perzentil zwischen p0,0 und p100

Amazon S3 CloudWatch-Replikationsmetriken

Mit Replikationsmetriken können Sie die Gesamtzahl und Größe der Objekte, die zur Replikation ausstehen, sowie die maximale Replikationszeit in die Zielregion überwachen. Nur Replikationsregeln, für die S3 Replication Time Control (S3 RTC) aktiviert ist, können Replikationsmetriken veröffentlichen.

Im Gegensatz zu CloudWatch-Speicher- und -Anforderungsmetriken können Sie Replikationsmetriken nicht nach Präfixen oder Tags filtern. Sie können jedoch eine Replikationsregel basierend auf Präfixen und Tags

einrichten. Ihre Replikationsmetriken überwachen dann die Replikation für die von Ihnen angegebenen Präfixe und Tags. Weitere Informationen finden Sie unter the section called “S3 Replication Time Control (S3 RTC)” (p. 687).

Note

Sie können Alarme für Ihre Replikationsmetriken in Amazon CloudWatch aktivieren. Wenn Sie Alarme für Ihre Replikationsmetriken einrichten, stellen Sie das Feld Missing data treatment (Behandlung fehlender Daten) auf Treat missing data as ignore (maintain the alarm state) (Fehlende Daten ignorieren) (Alarmstatus beibehalten)) ein.

Metrik	Beschreibung
`ReplicationLatency`	Die maximale Anzahl von Sekunden, um die sich die Replikationszielregion hinter der Quellregion für eine bestimmte Replikationsregel befindet. Einheiten: Sekunden Gültige Statistiken: Max
`BytesPendingReplication`	Die Gesamtanzahl der Bytes von Objekten, die für eine bestimmte Replikationsregel ausstehen. Einheiten: Byte Gültige Statistiken: Max
`OperationsPendingReplication`	Die Anzahl der Operationen mit ausstehender Replikation für eine bestimmte Replikationsregel. Einheiten: Anzahl Gültige Statistiken: Max

Amazon S3 CloudWatch-Dimensionen

Die nachstehenden Dimensionen werden verwendet, um Amazon S3-Metriken zu filtern.

Dimension	Beschreibung
`BucketName`	Diese Dimension filtert die angeforderten Daten nur für den identifizierten Bucket.
`StorageType`	Diese Dimension filtert die Daten, die Sie in einem Bucket gespeichert haben, nach den folgenden Speichertypen: • `StandardStorage`: Die Gesamtanzahl von Bytes für Objekte in der STANDARD-Speicherklasse. • `IntelligentTieringFAStorage`: Die Anzahl von Bytes für Objekte in der Stufe für häufigen Zugriff der INTELLIGENT_TIERING-Speicherklasse. • `IntelligentTieringIAStorage`: Die Anzahl von Bytes für Objekte in der Stufe für seltenen Zugriff der INTELLIGENT_TIERING-Speicherklasse. • `StandardIAStorage`: Die Anzahl von Bytes für Objekte in der STANDARD_IA-Speicherklasse (Standard – Infrequent Access).

Dimension	Beschreibung
	• `StandardIASizeOverhead`: Die Anzahl von Bytes für Objekte mit einer Größe von weniger als 128 KB in der STANDARD_IA-Speicherklasse. • `OneZoneIAStorage`: Die Anzahl von Bytes für Objekte in der ONEZONE_IA-Speicherklasse (OneZone – Infrequent Access). • `OneZoneIASizeOverhead`: Die Anzahl von Bytes für Objekte mit einer Größe von weniger als 128 KB in der ONEZONE_IA-Speicherklasse. • `ReducedRedundancyStorage`: Die Anzahl von Bytes für Objekte in der RRS-Speicherklasse (Reduced Redundancy Storage). • `GlacierStorage`: Die Anzahl von Bytes für Objekte in der GLACIER-Speicherklasse. • `GlacierStagingStorage`: Die Anzahl von Bytes für Teile von Multipart-Objekten, bevor die CompleteMultipartUpload-Anfrage für Objekte in der GLACIER-Speicherklasse abgeschlossen ist. • `GlacierObjectOverhead`: Für jedes archivierte Objekt fügt GLACIER 32 KB Speicher für den Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederherzustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen GLACIER-Gebühren in Rechnung gestellt. • `GlacierS3ObjectOverhead`: Für jedes in GLACIER archivierte Objekt verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten. Für diesen zusätzlichen Speicherplatz werden Ihnen die STANDARD-Gebühren in Rechnung gestellt. • `DeepArchiveStorage` - Die Gesamtzahl der Bytes für Objekte in der S3 Glacier Deep Archive-Speicherklasse. • `DeepArchiveObjectOverhead` - Für jedes archivierte Objekt fügt S3 Glacier Deep Archive 32 KB Speicher für den Index und zugehörige Metadaten hinzu. Diese zusätzlichen Daten sind erforderlich, um Ihr Objekt zu identifizieren und wiederherzustellen. Für diesen zusätzlichen Speicherplatz werden Ihnen die S3 Glacier Deep Archive-Gebühren in Rechnung gestellt. • `DeepArchiveS3ObjectOverhead` - Für jedes in S3 Glacier Deep Archive archivierte Objekt verwendet Amazon S3 8 KB Speicher für den Namen des Objekts und andere Metadaten. Für diesen zusätzlichen Speicherplatz werden Ihnen die STANDARD-Gebühren in Rechnung gestellt. • `DeepArchiveStagingStorage` - Die Anzahl von Bytes für Teile von Multipart-Objekten, bevor die CompleteMultipartUpload-Anfrage für Objekte in der S3 Glacier Deep Archive-Speicherklasse abgeschlossen ist.
`FilterId`	Diese Dimension filtert Metrikkonfigurationen, die Sie für Anfragemetriken für einen Bucket angeben, z. B. ein Präfix oder einen Tag. Sie geben eine Filter-ID an, wenn Sie eine Metrikkonfiguration erstellen. Weitere Informationen finden Sie unter Metrik-Konfigurationen für Buckets.

Zugreifen auf CloudWatch-Metriken

Sie können die folgenden Vorgehensweisen nutzen, um die Speichermetriken von Amazon S3 anzuzeigen. Damit Amazon S3-Metriken berücksichtigt werden, müssen Sie Zeitstempel für Anfang und Ende angeben. Für Metriken für einen bestimmten 24-Stunden-Zeitraum setzen Sie das Zeitintervall auf 86400 Sekunden, die Anzahl der Sekunden eines Tages. Denken Sie daran, die Dimensionen `BucketName` und `StorageType` festzulegen.

Wenn Sie beispielsweise die AWS CLI verwenden, um die durchschnittliche Größe eines bestimmten Buckets in Bytes zu erhalten, geben Sie den folgenden Befehl an.

```
aws cloudwatch get-metric-statistics --metric-name BucketSizeBytes --namespace AWS/S3
 --start-time 2016-10-19T00:00:00Z --end-time 2016-10-20T00:00:00Z --statistics Average
 --unit Bytes --region us-west-2 --dimensions Name=BucketName,Value=ExampleBucket
 Name=StorageType,Value=StandardStorage --period 86400 --output json
```

Dieses Beispiel erzeugt die folgende Ausgabe.

```
{
    "Datapoints": [
        {
            "Timestamp": "2016-10-19T00:00:00Z",
            "Average": 1025328.0,
            "Unit": "Bytes"
        }
    ],
    "Label": "BucketSizeBytes"
}
```

So zeigen Sie Metriken mithilfe der CloudWatch-Konsole an

1. Öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metrics aus.
3. Wählen Sie den Namespace S3 aus.
4. (Optional) Um eine Metrik anzuzeigen, geben Sie den Metriknamen in das Suchfeld ein.
5. (Optional) Um nach der Dimension StorageType zu filtern, geben Sie den Namen der Speicherklasse in das Suchfeld ein.

So zeigen Sie eine Liste gültiger Metriken, die für das AWS-Konto gespeichert sind, mit der AWS-CLI an

- Geben Sie als Eingabeaufforderung den folgenden Befehl ein.

  ```
  aws cloudwatch list-metrics --namespace "AWS/S3"
  ```

Zugehörige Ressourcen

- Amazon CloudWatch Logs API Reference
- Amazon CloudWatch-Benutzerhandbuch
- Aktion list-metrics in der AWS CLI Command Reference.
- Aktion get-metric-statistics in der AWS CLI Command Reference.
- Metrikkonfigurationen für Buckets (p. 747) aus.

Metrikkonfigurationen für Buckets

Mit Amazon CloudWatch-Anforderungsmetriken für Amazon S3 können Sie 1-minütige CloudWatch-Metriken erhalten, CloudWatch-Alarme festlegen und auf CloudWatch-Dashboards zugreifen, um Operationen und Leistung des Amazon S3-Speichers nahezu in Echtzeit anzuzeigen. Für Anwendungen, die von Cloud-Speicher abhängig sind, ermöglichen diese Metriken Ihnen, Betriebsprobleme schnell zu identifizieren und entsprechende Maßnahmen zu ergreifen. Wenn diese 1-Minuten-Metriken aktiviert sind, stehen sie standardmäßig auf Ebene der Amazon S3-Buckets zur Verfügung.

Sie müssen eine Metrikkonfiguration für einen Bucket erstellen, wenn Sie die CloudWatch-Anforderungsmetrik für die Objekte in diesem Bucket erhalten wollen. Sie können auch einen Filter für die Metriken definieren, die mit einem gemeinsamen Präfix oder gemeinsamen Objekt-Tags erfasst wurden. Sie können also Metrikfilter an bestimmte Geschäftsanwendungen, Workflows oder interne Organisationen anpassen.

Weitere Informationen zu den verfügbaren CloudWatch-Metriken und den Unterschieden zwischen Speicher- und Anforderungsmetriken finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).

Berücksichtigen Sie bei Verwendung von Metrikkonfigurationen Folgendes:

- Sie können maximal 1 000 Metrikkonfigurationen pro Bucket verwenden.
- Unter Verwendung von Filtern können Sie wählen, welche Objekte in einem Bucket Metrikkonfigurationen enthalten sollen. Ein Filtern nach einem gemeinsamen Präfix oder Objekt-Tag ermöglicht Ihnen, Metrik-Filter auf bestimmte Arbeitsabläufe von Geschäftsanwendungen oder interne Organisationen auszurichten. Um Metriken für den gesamten Bucket anzufordern, erstellen Sie eine Metrik-Konfiguration ohne Filter.
- Metrik-Konfigurationen sind nur erforderlich, um Anforderungsmetriken zu aktivieren. Tägliche Speichermetriken auf Bucket-Ebene sind immer aktiviert und werden allen Kunden ohne zusätzliche Kosten zur Verfügung gestellt. Derzeit ist es nicht möglich, tägliche Speichermetriken für eine gefilterte Untermenge von Objekten zu erhalten.
- Jede Metrikkonfiguration unterstützt den gesamten Satz verfügbarer Anforderungsmetriken (p. 741). Operationsspezifische Metriken (z. B. `PostRequests`) werden nur gemeldet, wenn es Anforderungen dieses Typs für den Bucket oder Filter gibt.
- Anforderungsmetriken werden für Operationen auf Objektebene gemeldet. Sie werden außerdem für Operationen gemeldet, die Bucket-Inhalte auflisten, beispielsweise GET Bucket (List Objects), GET Bucket Object Versions und List Multipart Uploads, nicht jedoch für andere Operationen mit Buckets.
- Anforderungsmetriken unterstützen das Filtern mit Präfixen, Speichermetriken dagegen nicht.

Best-Effort-Bereitstellungen von CloudWatch-Metriken

CloudWatch-Metriken werden auf einer Best-Effort-Basis bereitgestellt. Die meisten Anforderungen für ein Amazon S3-Objekt mit Anforderungsmetriken führen zum Senden eines Datenpunkts an CloudWatch.

Die Vollständigkeit und Rechtzeitigkeit der Metriken ist nicht garantiert. Der Datenpunkt für eine bestimmte Anforderung wird möglicherweise mit einem Zeitstempel zurückgegeben, der nach der tatsächlichen Anforderungsverarbeitung liegt. Der Datenpunkt könnte auch um eine Minute verzögert über CloudWatch bereitgestellt werden. Möglicherweise wird er auch überhaupt nicht bereitgestellt. CloudWatch-Anforderungsmetriken geben nahezu in Echtzeit eine Vorstellung von der Art des Datenverkehrs für den Bucket. Sie sind nicht als vollständige Abrechnung aller Anforderungen vorgesehen.

Aufgrund der Best-Effort-Natur dieser Funktion enthalten die im Fakturierungs- und Kostenverwaltungs-Dashboard verfügbaren Berichte möglicherweise eine oder mehrere Zugriffsanforderungen, die nicht in den Bucket-Metriken angezeigt werden.

Filtern von Metrikkonfigurationen

Bei der Arbeit mit CloudWatch-Metrikkonfigurationen haben Sie die Möglichkeit, die Konfiguration in Gruppen verwandter Objekte innerhalb eines einzelnen Buckets zu filtern. Sie können Objekte in einem Bucket zur Aufnahme in eine Metrikkonfiguration abhängig von einem oder mehreren der folgenden Elemente filtern:

- Objektschlüsselnamens-Präfix: Auch wenn das Amazon S3-Datenmodell eine flache Struktur hat, können Sie mittels eines Präfixes eine Hierarchie herstellen. Die Amazon S3-Konsole unterstützt diese Präfixe mit dem Ordnerkonzept. Wenn Sie nach dem Präfix filtern, werden Objekte mit demselben Präfix in die Metrikkonfiguration aufgenommen.
- Tag: Sie können Objekten Tags hinzufügen, d. h. Schlüsselwert-Namenspaare. Mit Tags können Sie Objekte einfacher finden und organisieren. Tags können auch als Filter für Metrikkonfigurationen verwendet werden.

Wenn Sie einen Filter angeben, können nur Anforderungen, die für einzelne Objekte ausgeführt werden, mit dem Filter übereinstimmen und in die gemeldeten Metriken aufgenommen werden. Anforderungen wie zum Löschen mehrerer Objekte und `List`-Anforderungen geben keine Metriken für Konfigurationen mit Filtern zurück.

Um ein komplexeres Filtern anzufordern, wählen Sie zwei oder mehr Elemente aus. Nur Objekte, die alle diese Elemente besitzen, werden in die Metrikkonfiguration aufgenommen. Wenn Sie keine Filter setzen, werden alle Objekte aus dem Bucket in die Metrikenkonfiguration aufgenommen.

Vorgehensweise zum Hinzufügen von Metrikkonfigurationen

Sie können einem Bucket über die Amazon S3-Konsole, die AWS CLI oder die Amazon S3-REST-API Metrikkonfigurationen hinzufügen. Informationen, wie Sie dies in der AWS Management Console ausführen, finden Sie unter Wie konfiguriere ich Anforderungsmetriken für einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Hinzufügen von Metrikkonfigurationen mit der AWS CLI

1. Installieren und Einrichten der AWS CLI Anleitungen finden Sie unter Einrichten der AWS-Befehlszeilenschnittstelle im Benutzerhandbuch für AWS Command Line Interface.
2. Öffnen Sie ein Terminalfenster.
3. Führen Sie den folgenden Befehl aus, um eine Metrikkonfiguration hinzuzufügen.

```
aws s3api put-bucket-metrics-configuration --endpoint https://s3.us-
west-2.amazonaws.com --bucket bucket-name --id metrics-config-id --metrics-
configuration '{"Id":"metrics-config-id","Filter":{"Prefix":"prefix1"}}'
```

4. Geben Sie den folgenden Befehl ein, um zu überprüfen, ob die Konfiguration hinzugefügt wurde.

```
aws s3api get-bucket-metrics-configuration --endpoint https://s3.us-
west-2.amazonaws.com --bucket bucket-name --id metrics-config-id
```

Damit erhalten Sie die folgende Antwort.

```
{
    "MetricsConfiguration": {
        "Filter": {
            "Prefix": "prefix1"
```

```
        },
        "Id": "metrics-config-id"
    }
}
```

Mit der Amazon S3-REST-API können Sie Metrikkonfigurationen programmgesteuert hinzufügen. Weitere Informationen finden Sie unter den folgenden Themen im Amazon Simple Storage Service API Reference:

- PUT Bucket metrics
- GET Bucket-Metrikkonfiguration
- List Bucket-Metrikkonfiguration
- DELETE Bucket-Metrikkonfiguration

Protokollieren mit Amazon S3

Sie können die Aktionen aufzeichnen, die von Benutzern, Rollen oder AWS-Services auf Amazon S3-Ressourcen durchgeführt werden und Protokolldatensätze zu Zwecken der Überprüfung und Compliance pflegen. Dazu können Sie Server access logging (Server-Zugriffsprotokollierung) (p. 777), AWS CloudTrail-Protokolle oder eine Kombination aus beiden verwenden. Wir empfehlen, dass Sie AWS CloudTrail zum Protokollieren der Aktionen auf Bucket- und Objektebene für Ihre Amazon S3-Ressourcen verwenden.

In der folgenden Tabelle sind die wichtigsten Eigenschaften von AWS CloudTrail-Protokollen und Amazon S3-Serverzugriffsprotokollen aufgelistet.

Protokolleigenschaften	AWS CloudTrail	Amazon S3-Serverprotokolle
Kann an andere Systeme (CloudWatch Logs, CloudWatch-Ereignisse) weitergeleitet werden	Ja	
Protokolle an mehreren Zielen bereitstellen (Beispiel: dieselben Protokolle an zwei verschiedene Buckets senden)	Ja	
Protokolle für eine Teilmenge von Objekten aktivieren (Präfix)	Ja	
Kontoübergreifende Protokollbereitstellung (Ziel- und Quell-Bucket in Besitz von verschiedenen Konten)	Ja	
Integritätsprüfung der Protokolldatei mit digitaler Signatur/Hashing	Ja	
Standard/Auswahl der Verschlüsselung für Protokolldateien	Ja	
Objekt-Operationen (mit Amazon S3-APIs)	Ja	Ja
Bucket-Operationen (mit Amazon S3-APIs)	Ja	Ja
Durchsuchbare UI für Protokolle	Ja	
Felder für Objektsperre-Parameter, ausgewählte Amazon S3-Eigenschaften für Protokolldatensätze	Ja	
Felder für `Object Size`, `Total Time`, `Turn-Around Time` und `HTTP Referer` für Protokolldatensätze		Ja

Protokolleigenschaften	AWS CloudTrail	Amazon S3-Serverprotokolle
Lebenszyklusübertragungen, Ablaufaktionen, Wiederherstellungen		Ja
Protokollieren von Schlüsseln in einer Batch-Delete-Operation		Ja
Authentifizierungsfehler[1]		Ja
Konten, an die Protokolle geliefert werden	Bucket-Eigentümer[2] und Auftraggeber	Nur Bucket-Eigentümer
Performance and Cost	AWS CloudTrail	Amazon S3 Server Logs
Preis	Verwaltungsereignisse (erste Bereitstellung) sind kostenlos. Für Datenereignisse fällt zusätzlich zur Speicherung der Protokolle eine Gebühr an	Keine Zusatzkosten neben der Speicherung der Protokolle
Geschwindigkeit der Protokollbereitstellung	Datenereignisse alle 5 Minuten; Verwaltungsereignisse alle 15 Minuten	Innerhalb weniger Stunden
Protokollformat	JSON	Protokolldatei mit durch Leerzeichen getrennten, durch neue Zeilen getrennten Datensätzen

Hinweise:

1. CloudTrail stellt keine Protokolle für Anforderungen bereit, deren Authentifizierung fehlschlägt (in dem die angegebenen Anmeldeinformationen nicht gültig sind). Es enthält jedoch Protokolle für Anforderungen, deren Authentifizierung fehlschlägt (`AccessDenied`), und Anforderungen, die von anonymen Benutzern gestellt werden.
2. Der S3-Bucket-Eigentümer erhält CloudTrail-Protokolle nur, wenn das Konto auch vollständigen Zugriff auf das Objekt in der Anforderung besitzt oder hat. Weitere Informationen finden Sie unter Aktionen auf Objektebene in kontoübergreifenden Szenarien (p. 754).

Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail

Amazon S3 ist mit AWS CloudTrail integriert und zeichnet die Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in Amazon S3 auf. CloudTrail erfasst eine Untermenge der API-Aufrufe für Amazon S3

als Ereignisse, einschließlich Aufrufen über die Amazon S3-Konsole und über Codeaufrufe an die Amazon S3-APIs. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail-Ereignissen an einen Amazon S3-Bucket, einschließlich Ereignissen für Amazon S3, aktivieren. Auch wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail-Konsole in Event history (Ereignisverlauf) anzeigen. Mit den von CloudTrail gesammelten Informationen können Sie die an Amazon S3 gestellte Anforderung, die IP-Adresse, von der die Anforderung gestellt wurde, den Initiator sowie den Zeitpunkt der Anforderung und weitere Angaben bestimmen.

Weitere Informationen über CloudTrail, einschließlich Konfiguration und Aktivierung, finden Sie im AWS CloudTrail User Guide.

Amazon S3-Informationen in CloudTrail

CloudTrail wird beim Erstellen Ihres AWS-Kontos für Sie aktiviert. Die in Amazon S3 auftretenden unterstützten Aktivitäten werden als CloudTrail-Ereignis zusammen mit anderen AWS-Serviceereignissen in Event history (Ereignisverlauf) aufgezeichnet. Sie können die neusten Ereignisse in Ihr(em) AWS-Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail-API-Ereignisverlauf.

Erstellen Sie für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, darunter Ereignisse für Amazon S3, einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3-Bucket. Wenn Sie einen Pfad in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem Amazon S3-Bucket bereit, den Sie angeben. Darüber hinaus können Sie andere AWS-Services konfigurieren, um die in den CloudTrail-Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie unter:

- Übersicht zum Erstellen eines Pfads
- In CloudTrail unterstützte Services und Integrationen
- Konfigurieren von Amazon SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail-Protokolldateien aus mehreren Regionen und EmpfangenCloudTrail von Protokolldateien aus mehreren Konten

Jedes Event oder jeder Protokolleintrag enthält Informationen über den Ersteller der Anforderung. Anhand der Identitätsinformationen zur Benutzeridentität können Sie Folgendes bestimmen:

- Ob die Anforderung mit Root- oder IAM-Benutzeranmeldeinformationen ausgeführt wurde.
- Ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen föderierten Benutzer ausgeführt wurde.
- Ob die Anforderung von einem anderen AWS-Service getätigt wurde.

Weitere Informationen finden Sie unter CloudTrail-Element "userIdentity".

Sie können die Protokolldateien beliebig lange im Bucket speichern. Sie können aber auch Amazon S3-Lebenszyklusregeln aufstellen, anhand derer die Protokolldateien automatisch archiviert oder gelöscht werden. Standardmäßig werden die Protokolldateien mit serverseitiger Amazon S3-Verschlüsselung (SSE) verschlüsselt.

Amazon S3-Aktionen auf Bucket-Ebene, die von der CloudTrail-Protokollierung nachverfolgt werden

CloudTrail protokolliert standardmäßig Aktionen auf Bucket-Ebene. Amazon S3-Datensätze werden zusammen mit anderen AWS-Service-Datensätzen in eine Protokolldatei geschrieben. CloudTrail bestimmt

anhand von Zeitraum und Dateigröße, wann eine neue Datei erstellt und in eine neue Datei geschrieben werden soll.

Die Tabellen in diesem Abschnitt listen die Amazon S3-Aktionen auf Bucket-Ebene auf, die für die Protokollierung durch CloudTrail unterstützt werden.

Amazon S3-Aktionen auf Bucket-Ebene, die von der CloudTrail-Protokollierung nachverfolgt werden

REST API-Name	API-Ereignisname, der im CloudTrail-Protokoll verwendet wird
DELETE Bucket	DeleteBucket
DELETE Bucket cors	DeleteBucketCors
DELETE-Bucket-Verschlüsselung	DeleteBucketEncryption
DELETE Bucket-Lebenszyklus	DeleteBucketLifecycle
DELETE Bucket-Richtlinie	DeleteBucketPolicy
DELETE Bucket replication	DeleteBucketReplication
DELETE Bucket-Markierung	DeleteBucketTagging
DELETE Bucket-Website	DeleteBucketWebsite
GET Bucket ACL	GetBucketAcl
GET Bucket cors	GetBucketCors
GET-Bucket-Verschlüsselung	GetBucketEncryption
GET Bucket-Lebenszyklus	GetBucketLifecycle
GET Bucket-Speicherort	GetBucketLocation
GET Bucket-Protokollierung	GetBucketLogging
GET Bucket-Benachrichtigung	GetBucketNotification
GET Bucket-Richtlinie	GetBucketPolicy
GET Bucket replication	GetBucketReplication
GET Bucket-Zahlungsanforderung	GetBucketRequestPayment
GET Bucket-Markierung	GetBucketTagging
GET Bucket-Versioning	GetBucketVersioning
GET Bucket-Website	GetBucketWebsite
GET Service (Auflisten aller Buckets)	ListBuckets
PUT Bucket	CreateBucket
PUT Bucket ACL	PutBucketAcl
PUT Bucket cors	PutBucketCors
PUT-Bucket-Verschlüsselung	PutBucketEncryption

REST API-Name	API-Ereignisname, der im CloudTrail-Protokoll verwendet wird
PUT Bucket-Lebenszyklus	PutBucketLifecycle
PUT Bucket-Protokollierung	PutBucketLogging
PUT Bucket-Benachrichtigung	PutBucketNotification
PUT Bucket-Richtlinie	PutBucketPolicy
PUT Bucket replication	PutBucketReplication
PUT-Bucket-Zahlungsanforderung	PutBucketRequestPayment
PUT Bucket-Markierung	PutBucketTagging
PUT Bucket-Versioning	PutBucketVersioning
PUT Bucket-Website	PutBucketWebsite

Zusätzlich zu diesen API-Operationen können Sie auch die Objektebenenaktion OPTIONS object verwenden. Diese Aktion wird von der CloudTrail-Protokollierung als Aktion auf Bucket-Ebene behandelt, weil die Aktion die CORS-Konfiguration eines Buckets überprüft.

Amazon S3-Aktionen auf Objektebene, die von der CloudTrail-Protokollierung nachverfolgt werden

Sie können auch CloudTrail-Protokolle für Amazon S3-Aktionen auf Objektebene abrufen. Dazu geben Sie das Amazon S3-Objekt für Ihren Pfad an. Wenn eine Aktion auf Objektebene in Ihrem Konto auftritt, wertet CloudTrail Ihre Pfadeinstellungen aus. Falls das Ereignis mit dem von Ihnen angegebenen Objekt in einem Pfad übereinstimmt, wird das Ereignis protokolliert. Weitere Informationen finden Sie unter Wie aktiviere ich die Protokollierung auf Objektebene für einen S3-Bucket mit AWS CloudTrail-Datenereignissen? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service und Protokollieren von Datenereignisse für Trails im AWS CloudTrail User Guide.

In der folgenden Tabelle sind die Aktionen auf Objektebene aufgeführt, die CloudTrail protokollieren kann:

REST API-Name	API-Ereignisname, der im CloudTrail-Protokoll verwendet wird
Abort Multipart Upload	AbortMultipartUpload
Abschließen eines mehrteiligen Uploads	CompleteMultipartUpload
Löschen mehrerer Objekte	DeleteObjects
DELETE Object	DeleteObject
GET Object	GetObject
GET Object ACL	GetObjectAcl
GET Object-Markierung	GetObjectTagging
GET Object-Torrent	GetObjectTorrent
HEAD Object	HeadObject

REST API-Name	API-Ereignisname, der im CloudTrail-Protokoll verwendet wird
Initiieren eines mehrteiligen Uploads	CreateMultipartUpload
Teile auflisten	ListParts
POST Object	PostObject
POST Object-Wiederherstellung	RestoreObject
PUT Object	PutObject
PUT Object ACL	PutObjectAcl
Put Object Tagging	PutObjectTagging
PUT Object – Kopieren	CopyObject
SELECT-Objektinhalt	SelectObjectContent
Hochladen eines Teiluploads	UploadPart
Hochladen eines Teiluploads – Kopieren	UploadPartCopy

Zusätzlich zu diesen Operationen können Sie die folgenden Operationen auf Bucket-Ebene verwenden, um CloudTrail-Protokolle unter bestimmten Bedingungen als Amazon S3-Aktionen auf Objektebene abzurufen:

- GET Bucket (List Objects) Version 2 – Auswahl eines im Trail angegebenen Präfixes.
- GET Bucket Object versions – Auswahl eines im Trail angegebenen Präfixes.
- HEAD Bucket: Geben Sie einen Bucket und ein leeres Präfix an.
- Delete Multiple Objects – Angabe eines Buckets und eines leeren Präfixes.

  Note

  CloudTrail protokolliert keine Schlüsselnamen für die mit der Operation „Delete Multiple Objects (Mehrere Objekte löschen)“ gelöschten Schlüssel

Aktionen auf Objektebene in kontoübergreifenden Szenarien

Im Folgenden finden Sie spezielle Anwendungsfälle für API-Aufrufe auf Objektebene in kontenübergreifenden Szenarien sowie Informationen dazu, wie CloudTrail-Protokolle gemeldet werden. CloudTrail stellt Protokolle stets für den Anforderer bereit (die Person, die den API-Aufruf ausgeführt hat). Bei der Einrichtung von kontoübergreifendem Zugriff sehen Sie sich die Beispiele in diesem Abschnitt an.

Note

Die Beispiele setzen voraus, dass die CloudTrail-Protokolle ordnungsgemäß konfiguriert sind.

Beispiel 1: CloudTrail liefert Zugriffsprotokolle an den Bucket-Eigentümer

CloudTrail liefert Zugriffsprotokolle nur dann an den Bucket-Eigentümer, wenn der Bucket-Eigentümer die Berechtigungen für die Objekt-API besitzt. Sehen Sie sich das folgende kontenübergreifende Szenario vor:

- Konto A ist Eigentümer des Buckets
- Konto B (Anforderer) versucht, auf ein Objekt in diesem Bucket zuzugreifen.

CloudTrail liefert die API-Zugriffsprotokolle auf Objektebene immer an den Auftraggeber. Darüber hinaus liefert CloudTrail diese Protokolle nur dann an den Bucket-Eigentümer, wenn der Bucket-Eigentümer die Berechtigungen für die API-Aktionen für dieses Objekt besitzt.

> Note
>
> Wenn der Bucket-Eigentümer gleich dem Objekteigentümer ist, erhält der Bucket-Eigentümer die Objekt-Zugriffsprotokolle. Andernfalls muss der Bucket-Eigentümer Berechtigungen über die Objekt-ACL für diese Objekt-API erhalten, um die API-Protokolle für den Objektzugriff zu erhalten.

Beispiel 2: CloudTrail gibt keine E-Mail-Adressen weiter, die bei der Einrichtung von Objekt-ACLs verwendet werden

Sehen Sie sich das folgende kontenübergreifende Szenario vor:

- Konto A ist Eigentümer des Buckets
- Konto B (der Auftraggeber) sendet eine Anforderung, um ein ACL-Recht für ein Objekt unter Verwendung einer E-Mail-Adresse einzurichten. Weitere Informationen zu ACLs finden Sie unter Zugriffskontrolllisten (ACL) – Übersicht (p. 480).

die Anforderung wird zusammen mit der E-Mail-Information in den Protokollen aufgezeichnet. Der Bucket-Eigentümer erhält jedoch das CloudTrail-Protokoll mit dem Ereignis, wenn er zum Empfang von Protokollen berechtigt ist (siehe Beispiel 1). Der Bucket-Eigentümer erhält jedoch keine Informationen über die ACL-Konfiguration, insbesondere die E-Mail des Empfängers und das erteilte Recht. Die einzige Information, die das Protokoll dem Bucket-Eigentümer mitteilt, ist, dass Konto B einen ACL-API-Aufruf ausgeführt hat.

CloudTrail-Nachverfolgung mit Amazon S3-SOAP-API-Aufrufen

CloudTrail verfolgt Amazon S3-SOAP-API-Aufrufe nach. Die Amazon S3-SOAP-Unterstützung über HTTP ist veraltet, jedoch weiterhin über HTTPS verfügbar. Weitere Informationen zur Amazon S3-SOAP-Unterstützung finden Sie unter Anhang A: Verwenden der SOAP-API (p. 816).

> Important
>
> Neuere Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Amazon S3-SOAP-Aktionen, die über die CloudTrail-Protokollierung nachverfolgt werden

SOAP API-Name	API-Ereignisname, der im CloudTrail-Protokoll verwendet wird
ListAllMyBuckets	ListBuckets
CreateBucket	CreateBucket
DeleteBucket	DeleteBucket
GetBucketAccessControlPolicy	GetBucketAcl
SetBucketAccessControlPolicy	PutBucketAcl
GetBucketLoggingStatus	GetBucketLogging
SetBucketLoggingStatus	PutBucketLogging

Verwendung von CloudTrail-Protokollen mit Amazon S3-Serverzugriffsprotokollen und CloudWatch Logs

AWS CloudTrail-Protokolle bieten eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem AWS-Service in Amazon S3 durchgeführten Aktionen, während Amazon S3-Server-Zugriffsprotokolle detaillierte Aufzeichnungen der an einen S3-Bucket gemachten Anforderungen enthalten. Weitere Informationen zur Funktionsweise der unterschiedlichen Protokolle und ihren Eigenschaften, ihrer Leistung und ihrer Kosten finden Sie unter the section called “Protokollieren mit Amazon S3” (p. 749).

Sie können auch AWS CloudTrail-Protokolle zusammen mit Amazon S3-Server-Zugriffsprotokollen verwenden. CloudTrail-Protokolle stellen eine detaillierte API-Nachverfolgung für Amazon S3-Operationen auf Bucket- und Objektebene bereit. Serverzugriff-Protokolle für Amazon S3 bieten Einblicke in Operationen auf Objektebene für Ihre Daten in Amazon S3. Weitere Informationen zu Server-Zugriffsprotokollen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).

Sie können CloudTrail-Protokolle auch zusammen mit CloudWatch für Amazon S3 verwenden. Die CloudTrail-Integration mit CloudWatch Logs liefert Informationen zu von CloudTrail erfassten API-Aktivitäten auf S3-Bucket-Ebene an einen CloudWatch-Protokoll-Stream in der angegebenen CloudWatch-Protokollgruppe. Sie können CloudWatch-Alarme für die Überwachung bestimmter API-Aktivitäten erstellen und erhalten Benachrichtigungen per E-Mail, wenn die spezifische API-Aktivität stattfindet. Weitere Informationen zu CloudWatch-Alarmen für die Überwachung bestimmter API-Aktivitäten finden Sie im AWS CloudTrail User Guide. Weitere Informationen zur Verwendung von CloudWatch mit Amazon S3 finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).

Beispiel: Amazon S3-Protokolldateieinträge

Ein Trail ist eine Konfiguration, durch die Ereignisse an den von Ihnen angegebenen Amazon S3-Bucket übermittelt werden. CloudTrail-Protokolldateien können einen oder mehrere Einträge enthalten. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar. Es enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anforderungsparameter. CloudTrail-Protokolldateien sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Das folgende Beispiel zeigt einen CloudTrail-Protokolleintrag, der die Aktionen GET Service, PUT Bucket ACL und GET Bucket-Versioning enthält.

```
{
    "Records": [
    {
        "eventVersion": "1.03",
        "userIdentity": {
            "type": "IAMUser",
            "principalId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
        "eventTime": "2019-02-01T03:18:19Z",
        "eventSource": "s3.amazonaws.com",
        "eventName": "ListBuckets",
        "awsRegion": "us-west-2",
        "sourceIPAddress": "127.0.0.1",
        "userAgent": "[]",
        "requestParameters": {
            "host": [
                "s3.us-west-2.amazonaws.com"
            ]
```

```
        },
        "responseElements": null,
        "additionalEventData": {
            "SignatureVersion": "SigV2",
            "AuthenticationMethod": "QueryString"
    },
        "requestID": "47B8E8D397DCE7A6",
        "eventID": "cdc4b7ed-e171-4cef-975a-ad829d4123e8",
        "eventType": "AwsApiCall",
        "recipientAccountId": "111122223333"
    },
    {
       "eventVersion": "1.03",
       "userIdentity": {
            "type": "IAMUser",
            "principalId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
      "eventTime": "2019-02-01T03:22:33Z",
      "eventSource": "s3.amazonaws.com",
      "eventName": "PutBucketAcl",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "",
      "userAgent": "[]",
      "requestParameters": {
          "bucketName": "",
          "AccessControlPolicy": {
              "AccessControlList": {
                  "Grant": {
                      "Grantee": {
                          "xsi:type": "CanonicalUser",
                          "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
                          "ID":
"d25639fbe9c19cd30a4c0f43fbf00e2d3f96400a9aa8dabfbbebe1906Example"
                       },
                      "Permission": "FULL_CONTROL"
                   }
              },
              "xmlns": "http://s3.amazonaws.com/doc/2006-03-01/",
              "Owner": {
                  "ID": "d25639fbe9c19cd30a4c0f43fbf00e2d3f96400a9aa8dabfbbebe1906Example"
              }
          }
          "host": [
              "s3.us-west-2.amazonaws.com"
          ],
          "acl": [
              ""
          ]
      },
      "responseElements": null,
      "additionalEventData": {
          "SignatureVersion": "SigV4",
          "CipherSuite": "ECDHE-RSA-AES128-SHA",
          "AuthenticationMethod": "AuthHeader"
      },
      "requestID": "BD8798EACDD16751",
      "eventID": "607b9532-1423-41c7-b048-ec2641693c47",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    },
    {
      "eventVersion": "1.03",
```

```
      "userIdentity": {
          "type": "IAMUser",
          "principalId": "111122223333",
          "arn": "arn:aws:iam::111122223333:user/myUserName",
          "accountId": "111122223333",
          "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
          "userName": "myUserName"
        },
      "eventTime": "2019-02-01T03:26:37Z",
      "eventSource": "s3.amazonaws.com",
      "eventName": "GetBucketVersioning",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "",
      "userAgent": "[]",
      "requestParameters": {
          "host": [
              "s3.us-west-2.amazonaws.com"
          ],
          "bucketName": "AWSDOC-EXAMPLE-BUCKET1",
          "versioning": [
              ""
          ]
      },
      "responseElements": null,
      "additionalEventData": {
          "SignatureVersion": "SigV4",
          "CipherSuite": "ECDHE-RSA-AES128-SHA",
          "AuthenticationMethod": "AuthHeader",
    },
      "requestID": "07D681279BD94AED",
      "eventID": "f2b287f3-0df1-4961-a2f4-c4bdfed47657",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```


Zugehörige Ressourcen

- AWS CloudTrail User Guide
- CloudTrail-Ereignisreferenz
- Verwenden von AWS CloudTrail zum Identifizieren von Amazon S3-Anforderungen (p. 758)

Verwenden von AWS CloudTrail zum Identifizieren von Amazon S3-Anforderungen

Mit Amazon S3 können Sie Anforderungen anhand eines AWS CloudTrail-Ereignisprotokolls identifizieren. AWS CloudTrail ist die bevorzugte Methode zur Identifizierung von Amazon S3-Anforderungen. Wenn Sie allerdings Amazon S3-Serverzugriffsprotokolle verwenden, finden Sie weitere Informationen unter the section called “Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von Amazon S3-Anforderungen” (p. 794).

Themen

- Vorgehensweise, mit der CloudTrail an Amazon S3 gerichtete Anforderungen abfängt (p. 759)
- Aktivieren der CloudTrail-Ereignisprotokollierung für S3-Buckets und -Objekte (p. 759)
- Identifizieren von Anforderungen an Amazon S3 in einem CloudTrail-Protokoll (p. 760)

Vorgehensweise, mit der CloudTrail an Amazon S3 gerichtete Anforderungen abfängt

Standardmäßig protokolliert CloudTrail API-Aufrufe auf S3-Bucket-Ebene für die letzten 90 Tage, aber keine an Objekte gerichteten Anforderungen. Aufrufe auf Bucket-Ebene sind beispielsweise `CreateBucket`, `DeleteBucket`, `PutBucketLifeCycle` und `PutBucketPolicy`. Sie können die Ereignisse auf Bucket-Ebene in der CloudTrail-Konsole anzeigen. Sie können dort jedoch keine Datenereignisse (Amazon S3-Aufrufe auf Objektebene) anzeigen. Sie müssen CloudTrail-Protokolle für sie analysieren oder abfragen.

Für Informationen zu den von CloudTrail abgefangenen Amazon S3-API-Aufrufen siehe Amazon S3-Informationen in CloudTrail (p. 751).

Aktivieren der CloudTrail-Ereignisprotokollierung für S3-Buckets und -Objekte

CloudTrail-Datenereignisse erlauben Ihnen das Abrufen von Informationen über Anforderungen auf Bucket- und auf Objektebene. Weitere Informationen zum Aktivieren von CloudTrail-Datenereignissen für einen bestimmten Bucket finden Sie unter Wie aktiviere ich die Protokollierung auf Objektebene für einen S3-Bucket mit AWS CloudTrail-Datenereignissen? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Um CloudTrail-Datenereignisse für alle oder für eine Liste von Buckets zu aktivieren, müssen Sie manuell einen Trail in CloudTrail erstellen.

Note

- In der Standardeinstellung findet CloudTrail nur Verwaltungsereignisse. Prüfen Sie, ob Datenereignisse für das Konto aktiviert wurden.
- Ein S3-Bucket mit hoher Workload kann in kurzer Zeit Tausende Protokolle generieren. Überlegen Sie gut, für wie lange CloudTrail-Datenereignisse für einen Bucket mit hoher Workload aktiviert werden sollen.

CloudTrail speichert Amazon S3-Datenereignisprotokolle in einem S3-Bucket Ihrer Wahl. Denken Sie über die Verwendung eines Buckets in einem separaten AWS-Konto zur besseren Organisation von Ereignissen aus mehreren Buckets nach, die möglicherweise an einem zentralen Ort in Ihrem Besitz sind, um diese leichter abfragen und analysieren zu können. AWS Organizations vereinfacht das Erstellen eines AWS-Kontos, das mit dem Konto verknüpft ist, das den Bucket besitzt, den Sie überwachen. Weitere Informationen finden Sie unter Was ist AWS Organizations? im AWS Organizations-Benutzerhandbuch.

Wenn Sie einen Trailer in CloudTrail im Bereich für Datenereignisse erstellen, können Sie das Kontrollkästchen Select all S3 buckets in your account (Alle S3-Buckets im Konto auswählen) aktivieren, damit alle Ereignisse auf Objektebene protokolliert werden.

Note

- Das Erstellen einer Amazon S3-Lebenszyklusrichtlinie für Ihren AWS CloudTrail-Datenereignis-Bucket ist eine bewährte Methode. Konfigurieren Sie die Lebenszyklusrichtlinie zum regelmäßigen Entfernen von Protokolldateien nach dem Zeitraum, der Ihres Erachtens für die

Überprüfung erforderlich ist. Dadurch wird die Menge an Daten reduziert, die Athena in einer Abfrage analysiert.

- Informationen zum Protokollformat siehe Protokollieren von Amazon S3-API-Aufrufen unter Verwendung von AWS CloudTrail.
- Beispiele zum Abfragen von CloudTrail-Protokollen finden Sie unter Analyze Security, Compliance, and Operational Activity Using AWS CloudTrail and Amazon Athena.

Identifizieren von Anforderungen an Amazon S3 in einem CloudTrail-Protokoll

Von CloudTrail protokollierte Ereignisse werden in komprimierten GZipped JSON-Objekten im S3-Bucket gespeichert. Um Anforderungen schnell zu finden, sollten Sie einen Service wie Amazon Athena verwenden, der CloudTrail-Protokolle indexiert und abfragt. Weitere Informationen zu CloudTrail und Athena finden Sie unter Abfragen von AWS CloudTrail-Protokollen.

Verwenden von Athena mit CloudTrail-Protokollen

Nachdem CloudTrail zum Senden von Ereignissen an einen Bucket eingerichtet wurde, sollten Sie an der Amazon S3-Konsole beobachten können, wie Objekte in den Ziel-Bucket gelangen. Diese sind folgendermaßen formatiert: *s3://<myawsexamplebucket1>/AWSLogs/<111122223333>/CloudTrail/<Region>/<yyyy>/<mm>/<dd>*

Example – Verwenden von Athena zum Abfragen von CloudTrail-Ereignisprotokollen für spezifische Anforderungen

Suchen Sie Ihre CloudTrail-Ereignisprotokolle:

```
s3://myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/us-east-2/2019/04/14
```

Mit CloudTrail-Ereignisprotokollen können Sie jetzt eine Athena-Datenbank und -Tabelle erstellen und folgendermaßen abfragen:

1. Öffnen Sie die Athena-Konsole unter https://console.aws.amazon.com/athena/.
2. Wechseln Sie zu der AWS-Region, in der sich der CloudTrail-S3-Ziel-Bucket befindet.
3. Erstellen Sie im Abfragefenster eine Athena-Datenbank für die CloudTrail-Ereignisse.

```
CREATE DATABASE s3_cloudtrail_events_db
```

4. Verwenden Sie die folgende Abfrage, um eine Tabelle für alle CloudTrail-Ereignisse im Bucket zu erstellen. Ersetzen Sie dabei den Bucket-Namen *<CloudTrail_myawsexamplebucket1>* durch den Namen Ihres Buckets. Geben Sie außerdem die im Bucket verwendete *AWS-Konto-ID* für CloudTrail an.

```
CREATE EXTERNAL TABLE s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table(
        eventversion STRING,
        useridentity STRUCT<
            type:STRING,
            principalid:STRING,
            arn:STRING,
            accountid:STRING,
            invokedby:STRING,
            accesskeyid:STRING,
            userName:STRING,
            sessioncontext:STRUCT<
                attributes:STRUCT<
```

```
                    mfaauthenticated:STRING,
                    creationdate:STRING>,
                    sessionissuer:STRUCT<
                    type:STRING,
                    principalId:STRING,
                    arn:STRING,
                    accountId:STRING,
                    userName:STRING>
                >
            >,
        eventtime STRING,
        eventsource STRING,
        eventname STRING,
        awsregion STRING,
        sourceipaddress STRING,
        useragent STRING,
        errorcode STRING,
        errormessage STRING,
        requestparameters STRING,
        responseelements STRING,
        additionaleventdata STRING,
        requestid STRING,
        eventid STRING,
        resources ARRAY<STRUCT<
            ARN:STRING,
            accountId:STRING,
            type:STRING>>,
        eventtype STRING,
        apiversion STRING,
        readonly STRING,
        recipientaccountid STRING,
        serviceeventdetails STRING,
        sharedeventid STRING,
        vpcendpointid STRING
    )
    ROW FORMAT SERDE 'com.amazon.emr.hive.serde.CloudTrailSerde'
    STORED AS INPUTFORMAT 'com.amazon.emr.cloudtrail.CloudTrailInputFormat'
    OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
    LOCATION 's3://<myawsexamplebucket1>/AWSLogs/<111122223333>/';
```

5. Testen Sie Athena, um sicherzustellen, dass die Abfrage erwartungsgemäß funktioniert.

```
SELECT * FROM s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table
WHERE eventsource='s3.amazonaws.com'
LIMIT 2;
```

Verwenden von AWS CloudTrail zum Identifizieren von Amazon S3 Signature Version 2-Anforderungen

Amazon S3 ermöglicht das Ermitteln der zum Signieren einer Anforderung verwendeten API-Signaturversion mit einem AWS CloudTrail-Ereignisprotokoll. Diese Funktion ist wichtig, weil die Unterstützung für Signature Version 2 deaktiviert wird (veraltet). Danach akzeptiert Amazon S3 keine Anforderungen mit Signature Version 2 mehr, alle Anforderungen müssen also mit Signature Version 4 signiert werden.

Wir empfehlen dringend, CloudTrail einzusetzen, um gegebenenfalls Workflows zu ermitteln, die mit Signature Version 2 signieren. Korrigieren Sie diese Workflows, indem Sie die betreffenden Bibliotheken

und den Code so aktualisieren, dass mittels Signature Version 4 signiert wird. Dadurch können Beeinträchtigungen der geschäftlichen Prozesse werden.

Weitere Informationen finden Sie unter Announcement: AWS CloudTrail for Amazon S3 adds new fields for enhanced security auditing in den AWS Discussion Forums.

> **Note**
>
> CloudTrail-Ereignisse für Amazon S3 enthalten die Signature Version in den Anforderungsdetails unter dem Schlüsselnamen von „`additionalEventData`“. Um die Signaturversion von Anforderungen zu ermitteln, die an Objekte in Amazon S3 gerichtet wurden (z. B. GET, PUT und DELETE), müssen Sie die standardmäßig deaktivierten CloudTrail-Datenereignisse aktivieren.

AWS CloudTrail ist die bevorzugte Methode zum Identifizieren von Signature Version 2-Anforderungen, wenn Sie Amazon S3-Serverzugriffsprotokolle verwenden. Weitere Informationen finden Sie unter Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von Signature Version 2-Anforderungen (p. 799).

Athena-Abfragebeispiele zum Identifizieren von Amazon S3 Signature Version 2-Anforderungen

Example – Alle Ereignisse auswählen, die Signature Version 2 entsprechen, und nur EventTime, S3 Action, Request_Parameters, Region, SourceIP und UserAgent drucken

Ersetzen Sie in der folgenden Athena-Abfrage *<s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table>* durch die Athena-Daten und erhöhen oder entfernen Sie die Beschränkung nach Bedarf.

```
SELECT EventTime, EventName as S3_Action, requestParameters as Request_Parameters,
 awsregion as AWS_Region, sourceipaddress as Source_IP, useragent as User_Agent
FROM s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table
WHERE eventsource='s3.amazonaws.com'
AND json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
LIMIT 10;
```

Example – Alle Anforderer auswählen, die Signature Version 2-Datenverkehr senden

```
SELECT useridentity.arn, Count(requestid) as RequestCount
FROM s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table
WHERE eventsource='s3.amazonaws.com'
    and json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
Group by useridentity.arn
```

Partitionieren von Signature Version 2-Daten

Wenn Sie über eine große Datenmenge verfügen, die abgefragt werden muss, können Sie Kosten und Laufzeit für Athena reduzieren, indem Sie eine partitionierte Tabelle erstellen.

Erstellen Sie zu diesem Zweck folgendermaßen eine neue Tabelle mit Partitionen.

```
   CREATE EXTERNAL TABLE
 s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table_partitioned(
        eventversion STRING,
```

```
        userIdentity STRUCT<
            type:STRING,
            principalid:STRING,
            arn:STRING,
            accountid:STRING,
            invokedby:STRING,
            accesskeyid:STRING,
            userName:STRING,
         sessioncontext:STRUCT<
                    attributes:STRUCT<
                    mfaauthenticated:STRING,
                    creationdate:STRING>,
                    sessionIssuer:STRUCT<
                    type:STRING,
                    principalId:STRING,
                    arn:STRING,
                    accountId:STRING,
                    userName:STRING>
                >
             >,
        eventTime STRING,
        eventSource STRING,
        eventName STRING,
        awsRegion STRING,
        sourceIpAddress STRING,
        userAgent STRING,
        errorCode STRING,
        errorMessage STRING,
        requestParameters STRING,
        responseElements STRING,
        additionalEventData STRING,
        requestId STRING,
        eventId STRING,
        resources ARRAY<STRUCT<ARN:STRING,accountId: STRING,type:STRING>>,
        eventType STRING,
        apiVersion STRING,
        readOnly STRING,
        recipientAccountId STRING,
        serviceEventDetails STRING,
        sharedEventID STRING,
        vpcEndpointId STRING
    )
    PARTITIONED BY (region string, year string, month string, day string)
    ROW FORMAT SERDE 'com.amazon.emr.hive.serde.CloudTrailSerde'
    STORED AS INPUTFORMAT 'com.amazon.emr.cloudtrail.CloudTrailInputFormat'
    OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
    LOCATION 's3://myawsexamplebucket1/AWSLogs/111122223333/';
```

Erstellen Sie dann die einzelnen Partitionen. Sie können keine Resultate aus noch nicht erstellten Daten ermitteln.

```
ALTER TABLE s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table_partitioned ADD
    PARTITION (region= 'us-east-1', year= '2019', month= '02', day= '19') LOCATION 's3://
myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/us-east-1/2019/02/19/'
    PARTITION (region= 'us-west-1', year= '2019', month= '02', day= '19') LOCATION 's3://
myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/us-west-1/2019/02/19/'
    PARTITION (region= 'us-west-2', year= '2019', month= '02', day= '19') LOCATION 's3://
myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/us-west-2/2019/02/19/'
    PARTITION (region= 'ap-southeast-1', year= '2019', month= '02', day= '19') LOCATION
 's3://myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/ap-southeast-1/2019/02/19/'
    PARTITION (region= 'ap-southeast-2', year= '2019', month= '02', day= '19') LOCATION
 's3://myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/ap-southeast-2/2019/02/19/'
```

```
    PARTITION (region= 'ap-northeast-1', year= '2019', month= '02', day= '19') LOCATION
 's3://myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/ap-northeast-1/2019/02/19/'
    PARTITION (region= 'eu-west-1', year= '2019', month= '02', day= '19') LOCATION 's3://
myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/eu-west-1/2019/02/19/'
    PARTITION (region= 'sa-east-1', year= '2019', month= '02', day= '19') LOCATION 's3://
myawsexamplebucket1/AWSLogs/111122223333/CloudTrail/sa-east-1/2019/02/19/';
```

Anschließend können Sie die Anforderung basierend auf diesen Partitionen erstellen und müssen nicht mehr den gesamten Bucket laden.

```
SELECT useridentity.arn,
Count(requestid) AS RequestCount
FROM s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table_partitioned
WHERE eventsource='s3.amazonaws.com'
AND json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
AND region='us-east-1'
AND year='2019'
AND month='02'
AND day='19'
Group by useridentity.arn
```

Verwenden von AWS CloudTrail zum Identifizieren des Zugriffs auf Amazon S3-Objekte

Sie können Ihr AWS CloudTrail-Ereignisprotokoll zum Identifizieren von Amazon S3-Objektzugriffsanforderungen für Datenereignisse verwenden, wie etwa GetObject, DeleteObject und PutObject, und weitere Informationen über diese Anforderungen entdecken.

Das folgende Beispiel zeigt, wie Sie alle PUT-Objektanforderungen für Amazon S3 vom AWS CloudTrail-Ereignisprotokoll abrufen.

Athena-Abfragebeispiel zum Identifizieren von Amazon S3-Objektzugriffsanforderungen

Ersetzen Sie in den folgenden Athena-Abfragebeispielen *<s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table>* durch Ihre Athena-Details und bearbeiten Sie den Datumsbereich je nach Bedarf.

Example – Alle Ereignisse auswählen, die PUT-Objektzugriffsanforderungen aufweisen, und nur EventTime, EventSource, SourceIP, UserAgent, BucketName, Object und UserARN drucken

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  requestParameters.bucketName as bucketName,
  requestParameters.key as object,
  userIdentity.arn as userArn
FROM
  s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table
WHERE
  eventName = ‘PutObject'
  AND eventTime BETWEEN "2019-07-05T00:00:00Z" and "2019-07-06T00:00:00Z"
```

Example – Alle Ereignisse auswählen, die GET-Objektzugriffsanforderungen aufweisen, und nur EventTime, EventSource, SourceIP, UserAgent, BucketName, Object und UserARN drucken

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  requestParameters.bucketName as bucketName,
  requestParameters.key as object,
  userIdentity.arn as userArn
FROM
  s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table
WHERE
  eventName = 'GetObject'
  AND eventTime BETWEEN "2019-07-05T00:00:00Z" and "2019-07-06T00:00:00Z"
```

Example – Alle anonymen Anforderereereignisse für einen Bucket in einem bestimmten Zeitraum auswählen und nur EventTime, EventSource, SourceIP, UserAgent, BucketName, UserIdentity, und UserARN drucken

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  requestParameters.bucketName as bucketName,
  userIdentity.arn as userArn,
  userIdentity.principalId
FROM
  s3_cloudtrail_events_db.cloudtrail_myawsexamplebucket1_table
WHERE
  userIdentity.principalId='ANONYMOUS_PRINCIPAL'
  AND eventTime BETWEEN "2019-07-05T00:00:00Z" and "2019-07-06T00:00:00Z"
```

Note

- Diese Abfragebeispiele können auch für die Sicherheitsüberwachung nützlich sein. Sie können die Ergebnisse auf `PutObject`- oder `GetObject`-Aufrufe von unerwarteten oder nicht autorisierten IP-Adresen/Anforderern und zum Aufdecken anonymer Anforderungen an Ihre Buckets prüfen.
- Diese Abfrage ruft nur Informationen von der Zeit ab, zu der die Protokollierung aktiviert wurde.

Wenn Sie Amazon S3-Server-Zugriffsprotokolle verwenden, finden Sie weitere Informationen unter Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von Objektzugriffsanforderungen (p. 799).

Zugehörige Ressourcen

- AWS CloudTrail User Guide
- CloudTrail-Ereignisreferenz

BitTorrent für Amazon S3 verwenden

Themen

BitTorrent ist ein offenes Peer-to-Peer-Protokoll, für die Verteilung von Dateien. Mit dem BitTorrent-Protokoll können Sie beliebige öffentlich zugreifbare Objekte in Amazon S3 abrufen. In diesem Abschnitt wird beschrieben, warum Sie BitTorrent verwenden sollten, um Ihre Daten aus Amazon S3 zu verteilen, und wie das geht.

Amazon S3 unterstützt das BitTorrent-Protokoll, sodass Developer bei einer hoch skalierten Verteilung von Inhalten Kosten sparen können. Amazon S3 ist gut geeignet, um Daten einfach und zuverlässig zu speichern. Der Standardverteilmechanismus für Amazon S3-Daten ist ein Client/Server-Download. Bei der Client/Server-Verteilung wird das gesamte Objekt von Punkt zu Punkt von Amazon S3 an jeden berechtigten Benutzer übertragen, der das Objekt anfordert. Während die Client/Server-Bereitstellung für zahlreiche Anwendungsfälle geeignet ist, ist sie nicht für jeden Benutzer optimal. Insbesondere steigen die Kosten der Client/Server-Verteilung linear, wenn die Anzahl der Benutzer zunimmt, die Objekte herunterladen. Damit kann es teuer werden, beliebte Objekte zu verteilen.

BitTorrent löst dieses Problem, indem dieselben Clients, die das Objekt herunterladen, als Verteiler genutzt werden: Jeder Client lädt einige Teile des Objekts aus Amazon S3 herunter, und einige von anderen Clients, während er gleichzeitig Teile desselben Objekts auf andere interessierte "Peers" hochlädt. Der Vorteil für die Herausgeber ist, dass für große, beliebte Dateien die tatsächlich von Amazon S3 bereitgestellte Datenmenge wesentlich kleiner sein kann, als sie bei einer Bedienung derselben Clients über einen Client/Server-Download wäre. Weniger Datenübertragung bedeutet niedrigere Kosten für den Herausgeber des Objekts.

Note

- Amazon S3 unterstützt das BitTorrent-Protokoll nicht in AWS-Regionen, die nach dem 30. Mai 2016 gestartet wurden.
- Sie können eine Torrent-Datei nur für Objekte erhalten, die weniger als 5 GB groß sind.

Gebühren für die BitTorrent-Bereitstellung

Für die Nutzung von BitTorrent mit Amazon S3 fallen keine zusätzlichen Kosten an. Die Datenübertragung über das BitTorrent-Protokoll wird mit demselben Tarif bemessen wie die Client/Server-Bereitstellung. Genauer gesagt, wenn ein herunterladender BitTorrent-Client einen "Teil" eines Objekts vom Amazon S3 "Seeder" anfordert, entstehen dieselben Gebühren wie für eine anonyme Anforderung dieses Teils unter Verwendung des REST- oder SOAP-Protokolls. Diese Gebühren erscheinen auf dieselbe Weise in Ihren Amazon S3-Rechnungen und Nutzungsberichten. Der Unterschied ist, dass wenn viele Clients dasselbe Objekt gleichzeitig über BitTorrent anfordern, die Datenmenge, die Amazon S3 bereitstellen muss, um die Anfragen dieser Clients zu erfüllen, niedriger ist, als bei einer Client/Server-Bereitstellung. Das liegt daran, dass die BitTorrent-Clients gleichzeitig untereinander hoch- und herunterladen.

Note

Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Die Einsparungen bei der Datenübertragung durch BitTorrent können abhängig davon, wie beliebt Ihr Objekt ist, maßgeblich variieren. Weniger beliebte Objekte benötigen eine umfangreichere Nutzung des "Seeders", um die Clients zu bedienen, der Unterschied zwischen BitTorrent-Verteilungskosten und Client/Server-Verteilungskosten ist für solche Objekte deshalb möglicherweise gering. Insbesondere, wenn nur jeweils ein Client ein bestimmtes Objekt herunterlädt, sind die Kosten für die BitTorrent-Bereitstellung dieselben wie für einen direkten Download.

Abrufen von In Amazon S3 gespeicherten Objekten mit BitTorrent

Jedes in Amazon S3 anonym lesbare Objekt kann auch über BitTorrent heruntergeladen werden. Dazu benötigen Sie eine BitTorrent-Client-Anwendung. Amazon stellt keine BitTorrent-Client-Anwendung bereit, aber es gibt zahlreiche kostenlose Clients. Die Amazon S3 BitTorrent-Implementierung wurde auf ihre Kompatibilität mit dem offiziellen BitTorrent-Client getestet (siehe http://www.bittorrent.com/).

Der Ausgangspunkt für einen BitTorrent-Download ist eine .torrent-Datei. Diese kleine Datei beschreibt für BitTorrent-Clients sowohl die herunterzuladende Datei, als auch, wo die Daten zu finden sind. Eine .torrent-Datei umfasst einen kleinen Bruchteil der Größe des eigentlichen herunterzuladenden Objekts. Nachdem Sie eine von Amazon S3 generierte .torrent-Datei in die BitTorrent-Client-Anwendung eingefügt haben, sollte sie sofort aus Amazon S3 und allen BitTorrent-„Peer“-Clients heruntergeladen werden.

Der Abruf einer .torrent-Datei für ein öffentlich verfügbares Objekt ist einfach. Sie fügen einfach einen Abfragezeichenfolgeparameter "?torrent" am Ende der REST GET-Anfrage für das Objekt hinzu. Es ist keine Authentifizierung erforderlich. Nachdem Sie einen BitTorrent-Client installiert haben, erfolgt das Herunterladen eines Objekts mit BitTorrent einfach, indem Sie diese URL in Ihrem Webbrowser öffnen.

Es gibt keinen Mechanismus, um die .torrent-Datei für ein Amazon S3-Objekt mit der SOAP API abzurufen.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Example

Dieses Beispiel ruft die Torrent-Datei für das Objekt "Nelson" im Bucket "quotes" ab.

Sample Request

```
GET /quotes/Nelson?torrent HTTP/1.0
Date: Wed, 25 Nov 2009 12:00:00 GMT
```

Sample Response

```
HTTP/1.1 200 OK
x-amz-request-id: 7CD745EBB7AB5ED9
Date: Wed, 25 Nov 2009 12:00:00 GMT
Content-Disposition: attachment; filename=Nelson.torrent;
Content-Type: application/x-bittorrent
Content-Length: 537
Server: AmazonS3

<body: a Bencoded dictionary as defined by the BitTorrent specification>
```

Veröffentlichung von Inhalten mit Amazon S3 und BitTorrent

Jedes anonym lesbare Objekte, das in Amazon S3 gespeichert ist, steht automatisch zum Download über BitTorrent zur Verfügung. Der Prozess zur Änderung der ACL für ein Objekt, um anonyme `READ`-Operationen zuzulassen, ist in Identity and Access Management in Amazon S3 (p. 329) beschrieben.

Sie können Ihre Clients auf Ihre in BitTorrent verfügbaren Objekte weiterleiten, indem Sie ihnen die .torrent-Datei direkt übergeben, oder indem Sie einen Link auf die ?torrent-URL Ihres Objekts veröffentlichen. Beachten Sie, dass die .torrent-Datei, die ein Amazon S3-Objekt beschreibt, on-demand generiert wird, wenn sie zum ersten Mal angefordert wird (über die REST-Ressource ?torrent). Die Generierung der.torrent-Datei für ein Objekt dauert proportional zur Größe dieses Objekts. Für große Objekte kann diese Zeit relativ lang sein. Daher sollten Sie die erste Anfrage selbst ausführen, bevor Sie einen ?torrent-Link veröffentlichen. Amazon S3 könnte einige Minuten benötigen, um auf diese erste Anfrage zu antworten, da die .torrent-Datei generiert wird. Wenn Sie das betreffende Objekt nicht aktualisieren, sind nachfolgende Anfragen für die .torrent-Datei schnell. Gehen Sie nach diesem Verfahren vor der Weitergabe eines ?torrent-Links vor, um einen reibungslosen BitTorrent-Download für Ihre Kunden sicherzustellen.

Um die Weitergabe einer Datei mit BitTorrent zu beenden, entfernen Sie einfach den anonymen Zugriff darauf. Dazu entfernen Sie entweder die Datei aus Amazon S3 oder Sie ändern Ihre Zugriffskontrollrichtlinie, um anonyme Leseoperationen zu verhindern. Anschließend agiert Amazon S3 nicht mehr als "Seeder" im BitTorrent-Netzwerk für Ihre Datei und bedient die .torrent-Datei nicht mehr über die ?torrent REST API. Nachdem eine .torrent-Datei für Ihre Datei veröffentlicht wurde, unterbricht diese Maßnahme möglicherweise die öffentlichen Downloads Ihres Objekts nicht, die ausschließlich unter Verwendung des BitTorrent Peer-to-Peer-Netzwerks erfolgen.

Behandlung von REST- und SOAP-Fehlern

Themen

Dieser Abschnitt beschreibt REST- und SOAP-Fehler und wie sie gehandhabt werden.

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Die REST-Fehlerantwort

Themen

Wenn auf eine REST-Anfrage ein Fehler zurückgegeben wird, ist die HTTP-Antwort wie folgt aufgebaut:

- Ein XML-Fehlerdokument als Antworttext
- Content-Type: application/xml
- Der entsprechende 3xx-, 4xx- oder 5xx-HTTP-Statuscode

Nachfolgend finden Sie ein Beispiel für eine REST-Fehlermeldung.

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>NoSuchKey</Code>
  <Message>The resource you requested does not exist</Message>
  <Resource>/mybucket/myfoto.jpg</Resource>
  <RequestId>4442587FB7D0A2F9</RequestId>
</Error>
```

Weitere Informationen zu Amazon S3-Fehlern finden Sie unter Liste der Fehlercodes.

Antwort-Header

Die folgenden Antwort-Header werden von allen Operationen zurückgegeben:

- `x-amz-request-id:` Eine eindeutige ID, die das System jeder Anfrage zuordnet. In dem unwahrscheinlichen Fall, dass Sie Probleme mit Amazon S3 Amazon haben, kann Ihnen diese helfen, das Problem zu beheben.
- `x-amz-id-2:` Ein spezielles Token, das uns hilft, Probleme aufzulösen.

Antwort auf einen Fehler

Themen

Bei einer fehlerhaften Amazon S3-Anforderung erhält der Client eine Fehlermeldung zurück. Das genaue Format der Fehlermeldung ist spezifisch für die API. Die REST-Fehlermeldung beispielsweise unterscheidet sich von der SOAP-Fehlermeldung. Alle Fehlermeldungen haben jedoch gemeinsame Elemente.

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Fehlercode

Der Fehlercode ist eine Zeichenfolge handelt, die eine Fehlerbedingung eindeutig identifiziert. Er ist für Programme vorgesehen, die Fehler dem Typ nach erkennen und verarbeiten. Viele Fehlercode gibt es für SOAP- und REST APIs, einige sind jedoch spezifisch für die API. NoSuchKey beispielsweise ist universell, während UnexpectedContent nur in Reaktion auf eine ungültige REST-Anfrage auftreten kann. In jedem Fall haben SOAP-Fehlercodes ein Präfix, das in der Fehlercodetabelle angegeben ist, ein NoSuchKey-Fehler wird also in SOAP als Client.NoSuchKey zurückgegeben.

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Fehlermeldung

Die Fehlermeldung enthält eine generische Beschreibung der Fehlerbedingung in englischer Sprache. Sie ist für dafür vorgesehen, von Menschen gelesen zu werden. Einfache Programme zeigen dem Endbenutzer die Meldung direkt an, wenn sie auf eine Fehlerbedingung treffen, die sie nicht kennen, oder die sie nicht verarbeiten können. Komplexere Programme mit einer umfangreicheren Fehlerverarbeitung und einer geeigneten Internationalisierung werden die Fehlermeldung eher ignorieren.

Weitere Details

Viele Fehlermeldungen enthalten zusätzliche strukturierte Daten, die von einem Entwickler gelesen und interpretiert werden, der Programmfehler diagnostiziert. Wenn Sie beispielsweise einen Content-MD5-Header mit einer REST PUT-Anfrage senden, die nicht mit dem auf dem Server berechneten Digest übereinstimmt, erhalten Sie einen BadDigest-Fehler. Die Fehlermeldung enthält auch als Details den von uns berechneten Digest, sowie den Digest, den Sie erwarten und uns mitgeteilt haben. Während der Entwicklung können Sie diese Informationen nutzen, um den Fehler zu diagnostizieren. In der Produktion

könnte ein ordnungsgemäß funktionierendes Programm diese Informationen in seinem Fehlerprotokoll enthalten.

Die SOAP-Fehlerantwort

Note

Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

In SOAP wird eine Fehlermeldung als SOAP-Fehler an den Client zurückgegeben, mit dem HTTP-Antwortcode 500. Wenn Sie keinen SOAP-Fehler erhalten, war Ihre Anfrage erfolgreich. Der Amazon S3 SOAP-Fehlercode besteht aus einem SOAP 1.1-Standardfehlercode („Server“ oder „Client“), der mit dem Amazon S3-spezifischen Fehlercode verkettet ist. Beispiel: "Server.InternalError" oder "Client.NoSuchBucket". Das SOAP-Fehlerzeichenfolgenelement enthält eine generische, vom Menschen lesbare Fehlermeldung in englischer Sprache. Das SOAP-Fehlerdetailelement schließlich enthält verschiedene für den Fehler relevante Informationen.

Wenn Sie beispielsweise versuchen, das nicht existierende Objekt "Fred" zu löschen, enthält der Rumpf der SOAP-Antwort den SOAP-Fehler "NoSuchKey".

Example

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.NoSuchKey</Faultcode>
    <Faultstring>The specified key does not exist.</Faultstring>
    <Detail>
      <Key>Fred</Key>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

Weitere Informationen zu Amazon S3-Fehlern finden Sie unter Liste der Fehlercodes.

Bewährte Methoden für den Umgang mit Amazon S3-Fehlern

Wenn Sie eine Anwendung für Amazon S3 entwerfen, müssen Sie Amazon S3-Fehler auf geeignete Weise verarbeiten. Dieser Abschnitt beschreibt Aspekte, die Sie beim Entwurf Ihrer Anwendung berücksichtigen sollten.

Wiederholung bei InternalErrors

Interne Fehler sind Fehler, die innerhalb der Amazon S3-Umgebung auftreten.

Anfragen, die eine InternalError-Antwort erhalten, wurden möglicherweise nicht verarbeitet. Gibt eine PUT-Anfrage beispielsweise einen InternalError zurück, ruft ein nachfolgendes GET möglicherweise den alten Wert oder den aktuellen Wert ab.

Wenn Amazon S3 eine InternalError-Antwort zurückgibt, wiederholen Sie die Anfrage.

Optimieren der Anwendung für wiederholte SlowDown-Fehler

Wie jedes verteilte System besitzt S3 Schutzmechanismen, die einen beabsichtigten oder nicht beabsichtigten übermäßigen Ressourcenverbrauch erkennen und entsprechend reagieren. SlowDown-Fehler können auftreten, wenn eine hohe Anfragerate einen dieser Mechanismen auslöst. Eine Reduzierung Ihrer Anfragerate verringert oder eliminiert Fehler dieses Typs. Allgemein gilt, dass der Mehrzahl der Benutzer diese Fehler nicht regelmäßig angezeigt werden. Wenn Sie weitere Informationen benötigen oder sehr viele oder unerwartete SlowDown-Fehler erhalten, veröffentlichen Sie bitte einen Beitrag im Amazon S3-Entwicklerforum https://forums.aws.amazon.com/ oder registrieren sich für den AWS Premium Support https://aws.amazon.com/premiumsupport/.

Isolieren von Fehlern

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Amazon S3 unterstützt verschiedene Fehlercodes, die sowohl von der SOAP, als auch von der REST API verwendet werden. Die SOAP API gibt Amazon S3-Standardfehlercodes zurück. Die REST API ist darauf ausgelegt, wie ein HTTP-Standardserver auszusehen und mit vorhandenen HTTP-Clients zu interagieren (z. B. Browsern, HTTP-Clientbibliotheken, Proxies, Caches usw.). Um sicherzustellen, dass die HTTP-Clients Fehler ordnungsgemäß verarbeiten, bilden wir jeden Amazon S3-Fehler auf einen HTTP-Statuscode ab.

HTTP-Statuscodes sind weniger ausdrucksstark als Amazon S3-Fehlercodes und enthalten weniger Informationen über den Fehler. Beispielsweise sind die Fehler `NoSuchKey` und `NoSuchBucket` Amazon S3 dem Statuscode `HTTP 404 Not Found` zugeordnet.

Obwohl HTTP-Statuscode weniger Informationen über den Fehler enthalten, können Clients, die HTTP verstehen, aber nicht die Amazon S3 API, den Fehler normalerweise korrekt verarbeiten.

Sie sollten daher bei der Fehlerbehandlung oder beim Anzeigen von Amazon S3-Fehlern für Endbenutzer den Amazon S3-Fehlercode anstelle des HTTP-Statuscodes verwenden, da dieser die meisten Informationen zum Fehler enthält. Beim Debuggen Ihrer Anwendung sollten Sie außerdem das lesbare Element <Details> der XML-Fehlerantwort heranziehen.

Amazon S3-Fehlerbehebung

Dieser Abschnitt beschreibt die Fehlerbehebung in Amazon S3 und erklärt, wie Sie Anfrage-IDs erhalten, die Sie brauchen, wenn Sie Kontakt mit dem AWS-Support aufnehmen.

Themen

Amazon S3-Fehlerbehebung nach Symptomen

Die folgenden Themen listen die Symptome auf, die Ihnen dabei helfen, einige der Probleme zu beheben, die bei der Arbeit mit Amazon S3 auftreten können.

Symptome

Deutliche Zunahme von HTTP 503-Antworten auf Amazon S3-Anfragen an Buckets mit aktiviertem Versioning

Wenn Sie eine deutliche Zunahme von HTTP 503-Verlangsamungsantworten feststellen, die für Amazon S3 PUT- oder DELETE-Objektanfragen an einen Bucket mit aktiviertem Versioning eingehen, befindet sich möglicherweise ein oder mehrere Objekte im Bucket, für die Millionen von Versionen vorhanden sind. Wenn Sie Objekte mit Millionen Versionen haben, drosselt Amazon S3 automatisch Anfragen an den Bucket, um den Kunden vor übermäßigem Anfragedatenverkehr zu schützen, wodurch potenziell andere Anfragen an denselben Bucket beeinträchtigt werden können.

Um festzustellen, welche S3-Objekte Millionen Versionen haben, verwenden Sie das Amazon S3-Bestandswerkzeug. Das Bestandswerkzeug generiert einen Bericht, der eine flache Dateiliste der Objekte in einem Bucket enthält. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503).

Das Amazon S3-Team bittet die Kunden, Anwendungen zu überprüfen, die wiederholt dasselbe S3-Objekt überschreiben und damit potenziell Millionen Versionen für dieses Objekt erstellen, um festzustellen, ob die Anwendung wie beabsichtigt funktioniert. Wenn Sie einen Anwendungsfall haben, in dem Millionen von Versionen für ein oder mehrere S3-Objekte benötigt werden, wenden Sie sich an das AWS-Supportteam unter AWS Support, um Ihren Anwendungsfall zu besprechen und uns zu ermöglichen, die optimale Lösung für Ihr Szenario für Sie zu finden.

Um dieses Problem zu vermeiden, sollten Sie die folgenden bewährte Methoden beachten:

- Aktivieren Sie eine Ablaufrichtlinie für das Lebenszyklusmanagement „NonCurrentVersion“ und eine „ExpiredObjectDeleteMarker“-Richtlinie, um die vorherigen Versionen von Objekten veraltern zu lassen und Markierungen ohne zugeordnete Datenobjekte im Bucket zu löschen.
- Halten Sie Ihre Verzeichnisstruktur so flach wie möglich und machen Sie jeden Verzeichnisnamen einzigartig.

Unerwartetes Verhalten beim Zugriff auf Buckets, die mit CORS eingerichtet wurden

Wenn Sie ein unerwartetes Verhalten feststellen, wenn Sie auf Buckets zugreifen, die mit CORS-Konfiguration (Cross-Origin Resource Sharing) eingerichtet wurden, lesen Sie unter Fehlerbehebung bei CORS-Problemen (p. 183) nach.

Amazon S3-Anfrage-IDs für den AWS Support erhalten

Wenn Sie den AWS Support kontaktieren müssen, weil Fehler oder ein unerwartetes Verhalten in Amazon S3 auftreten, brauchen Sie die Anfrage-ID der fehlgeschlagenen Aktion. Diese Anfrage-IDs ermöglichen dem AWS Support, Ihnen bei der Auflösung Ihrer Probleme zu helfen. Anfrage-IDs werden paarweise vergeben. Sie werden in jeder von Amazon S3 verarbeiteten Antwort zurückgegeben (auch in den fehlerhaften). Der Zugriff darauf erfolgt über Verbose-Protokolle. Es gibt verschiedene übliche Verfahren für den Abruf Ihrer Anforderungs-IDs, darunter S3-Zugriffsprotokolle und CloudTrail-Ereignisse/Datenereignisse.

Nachdem Sie diese Protokolle wiederhergestellt haben, kopieren Sie diese beiden Werte. Sie brauchen sie, wenn Sie den AWS Support kontaktieren. Weitere Informationen zum AWS Support finden Sie unter Kontakt.

Themen

Anfrage-IDs mit HTTP ermitteln

Sie können Ihre Anfrage-IDs `x-amz-request-id` und `x-amz-id-2` ermitteln, indem Sie die Abschnitte einer HTTP-Anfrage protokollieren, bevor sie die Zielanwendung erreicht. Es gibt verschiedene Tools von Drittanbietern, mit denen Verbose-Protokolle für HTTP-Anfragen wiederhergestellt werden können. Wählen Sie ein Tool, dem Sie vertrauen, führen Sie es aus, und überwachen Sie den Port, über den Ihr Amazon S3-Datenverkehr läuft, während Sie eine weitere Amazon S3 HTTP-Anfrage senden.

Für HTTP-Anfragen sieht das Paar der Anfrage-IDs wie in den folgenden Beispielen gezeigt aus.

```
x-amz-request-id: 79104EXAMPLEB723
x-amz-id-2: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km
```

> **Note**
>
> HTTPS-Anfragen werden in den meisten Paketerfassungen verschlüsselt und verborgen.

Anfrage-IDs mit einem Webbrowser ermitteln

Die meisten Webbrowser beinhalten Entwicklerwerkzeuge, die Ihnen gestatten, die Anfrage-Header anzuzeigen.

Für auf einem Webbrowser basierte Anfragen, die einen Fehler zurückgeben, sieht das Paar der Anfrage-IDs wie in den folgenden Beispielen gezeigt aus.

```
<Error><Code>AccessDenied</Code><Message>Access Denied</Message>
<RequestId>79104EXAMPLEB723</RequestId><HostId>IOWQ4fDEXAMPLEQM
+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km</HostId></Error>
```

Um die das Paar der Anfrage-IDs aus erfolgreichen Anfragen zu erhalten, brauchen Sie die Entwicklerwerkzeuge, um die HTTP-Antwortheader anzuzeigen. Weitere Informationen über Entwicklerwerkzeuge für spezifische Browser finden Sie unter Amazon S3 Fehlerbehebung – Wiederherstellung Ihrer S3-Anfrage-IDs (Troubleshooting - How to recover your S3 request IDs) in den AWS-Entwicklerforen.

Anfrage-IDs mit AWS SDKs ermitteln

Die folgenden Abschnitte enthalten Informationen für die Konfiguration der Protokollierung unter Verwendung eines AWS SDK. Sie können die Verbose-Protokollierung für jede Anfrage und jede Antwort aktivieren, was Sie aber in Produktionssystemen unterlassen sollten, weil große Anfragen/Antworten eine maßgebliche Verlangsamung einer Anwendung verursachen können.

Für AWS SDK-Anfragen sieht das Paar der Anfrage-IDs wie in den folgenden Beispielen gezeigt aus.

```
Status Code: 403, AWS Service: Amazon S3, AWS Request ID: 79104EXAMPLEB723
AWS Error Code: AccessDenied  AWS Error Message: Access Denied
S3 Extended Request ID: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km
```

Anfrage-IDs mit SDK für PHP ermitteln

Sie können die Protokollierung mit PHP konfigurieren. Weitere Informationen finden Sie unter Wie kann ich sehen, welche Daten übertragen wurden? im FAQ-Abschnitt des AWS SDK for PHP.

Anfrage-IDs mit SDK für Java ermitteln

Sie können die Protokollierung für spezifische Anfragen oder Antworten aktivieren, um nur die relevanten Header zu erfassen. Dazu importieren Sie die Klasse `com.amazonaws.services.s3.s3ResponseMetadata`. Anschließend können Sie die Anfrage in einer Variablen speichern, bevor Sie die eigentliche Anfrage ausführen. Rufen Sie `getCachedResponseMetadata(AmazonWebServiceRequest request).getRequestID()` auf, um die protokollierte Anfrage oder Antwort zu erhalten.

Example

```
PutObjectRequest req = new PutObjectRequest(bucketName, key, createSampleFile());
s3.putObject(req);
S3ResponseMetadata md = s3.getCachedResponseMetadata(req);
System.out.println("Host ID: " + md.getHostId() + " RequestID: " + md.getRequestId());
```

Alternativ können Sie eine Verbose-Protokollierung jeder Java-Anfrage und -Antwort verwenden. Weitere Informationen finden Sie unter Verbose-Protokollierung des Netzwerkverkehrs im Thema „Protokollieren von AWS SDK for Java-Aufrufen“ im AWS SDK for Java Developer Guide.

Anfrage-IDs mit AWS SDK für .NET ermitteln

Mit dem eingebauten Protokollierungswerkzeug `System.Diagnostics` können Sie die Protokollierung in AWS SDK für .NET konfigurieren. Weitere Informationen finden Sie im Post Protokollierung mit dem AWS SDK for .NET (Logging with the AWS SDK for .NET) im AWS Developer Blog.

Note

Standardmäßig enthält das zurückgegebene Protokoll nur Fehlerinformationen. Die Konfigurationsdatei braucht `AWSLogMetrics` (und optional `AWSResponseLogging`), um die Anfrage-IDs zu erhalten.

Anfrage-IDs mit SDK for Python ermitteln

Die Protokollierung in Python konfigurieren Sie, indem Sie Ihrem Code die beiden folgenden Zeilen hinzufügen, um die Debugging-Information in eine Datei auszugeben.

```
import logging
logging.basicConfig(filename="mylog.log", level=logging.DEBUG)
```

Wenn Sie die Boto Python-Schnittstelle für AWS verwenden, können Sie die Debugging-Stufe auf 2 setzen, wie in den Boto-Dokumenten hier erläutert.

Anfrage-IDs mit SDK für Ruby ermitteln

Sie können Ihre Anfrage-IDs mit SDK für Ruby Version 1, Version 2 oder Version 3 ermitteln.

- Verwendung der SDK für Ruby- Version 1– Mit der folgenden Codezeile können Sie eine globale HTTP-Übertragungsprotokollierung aktivieren.

```
s3 = AWS::S3.new(:logger => Logger.new($stdout), :http_wire_trace => true)
```

- Verwendung von SDK für Ruby-Version 2 oder Version 3– Mit der folgenden Codezeile können Sie eine globale HTTP-Übertragungsprotokollierung aktivieren.

```
s3 = Aws::S3::Client.new(:logger => Logger.new($stdout), :http_wire_trace => true)
```

Anfrage-IDs mit AWS CLI ermitteln

Sie erhalten Ihre Anfrage-IDs in der AWS CLI, indem Sie Ihrem Befehl `--debug` hinzufügen.

Verwandte Themen

Weitere Themen zur Fehlersuche und zum Support finden Sie unter:

- Fehlerbehebung bei CORS-Problemen (p. 183)
- Behandlung von REST- und SOAP-Fehlern (p. 769)
- AWS Support – Dokumentation

Weitere Informationen zur Fehlersuche unter Verwendung von Tools von Drittanbietern finden Sie unter Abrufen von Amazon S3-Anfrage-IDs in den AWS-Entwicklerforen.

Amazon S3-Serverzugriffsprotokollierung

Die Server-Zugriffsprotokollierung bietet detaillierte Aufzeichnungen über die Anforderungen, die an einen Bucket gestellt wurden. Server-Zugriffsprotokolle sind für viele Anwendungen nützlich. Beispielsweise können Zugriffsprotokoll-Informationen bei Sicherheits- und Zugriffsprüfungen nützlich sein. Außerdem erfahren Sie damit mehr über Ihren Kundenstamm und verstehen Ihre Amazon S3-Rechnung.

> Note
>
> Server-Zugriffsprotokolle protokollieren keine Informationen zu Fehlern wegen Umleitungen in falsche Regionen, die nach dem 20. März 2019 gestartet wurden. Fehler wegen Umleitungen in falsche Regionen treten auf, wenn eine Anforderung für ein Objekt/einen Bucket an einen anderen Endpunkt als den Endpunkt der Region gerichtet wird, in der sich der Bucket befindet.

Themen

Vorgehensweise zum Aktivieren der Server-Zugriffsprotokollierung

Um Zugriffsanforderungen für Ihren Bucket nachzuverfolgen, können Sie die Server-Zugriffsprotokollierung aktivieren. Jeder Zugriffsprotokolldatensatz enthält Details über eine Zugriffsanforderung, z. B. Auftraggeber, Bucket-Name, Anforderungszeit, Anforderungsaktion, Antwortstatus und Fehlercode, falls relevant.

> Note
>
> Es fallen keine zusätzlichen Gebühren für die Aktivierung der Serverzugriffsprotokollierung in einem Amazon S3-Bucket an, und es wird Ihnen nicht in Rechnung gestellt, wenn die Protokolle mit PUT in Ihrem Bucket abgelegt werden. Für die Speicherung der Protokolldateien, die das System an Ihren Bucket überträgt, fallen die normalen Gebühren für die Speicherung an. Sie können diese Protokolldateien jederzeit löschen. Nachfolgende Lesevorgänge und andere Anforderungen an diese Protokolldateien werden wie bei jedem anderen Objekt normal berechnet, einschließlich Datenübertragungsgebühren.

Die Protokollierung ist standardmäßig deaktiviert. Wenn die Protokollierung aktiviert ist, werden die Protokolle in einem Bucket in derselben AWS-Region wie der des Quellbereichs gespeichert.

Gehen Sie folgendermaßen vor, um die Zugriffsprotokollierung zu aktivieren:

- Aktivieren Sie die Protokollierung, indem Sie die Protokollierungskonfiguration zu dem Bucket hinzufügen, für den Amazon S3 Zugriffsprotokolle bereitstellen soll. Wir bezeichnen diesen Bucket als Quell-Bucket.
- Gewähren Sie der Amazon S3-Protokollbereitstellungsgruppe Schreibberechtigung für den Bucket, in dem die Zugriffsprotokolle gespeichert werden sollen. Wir bezeichnen diesen Bucket als Ziel-Bucket.

  Note

  - Amazon S3 unterstützt nur das Gewähren von Rechten zur Bereitstellung von Zugriffsprotokollen über eine Bucket-ACL, nicht über eine Bucket-Richtlinie.
  - Das Hinzufügen von deny-Bedingungen zu einer Bucket-Richtlinie hindert Amazon S3 möglicherweise an der Bereitstellung von Zugriffsprotokollen.
  - Die Standardbucketverschlüsselung am Zielbucket darf nur verwendet werden, wenn AES256 (SSE-S3) ausgewählt ist. Die SSE-KMS-Verschlüsselung wird nicht unterstützt.
  - S3 Objektsperre kann auf dem Protokollzielbucket nicht aktiviert werden.

Um die Protokollbereitstellung zu aktivieren, geben Sie die folgenden Informationen für die Protokollierungskonfiguration an:

- Den Namen des Ziel-Buckets, in dem Amazon S3 die Zugriffsprotokolle als Objekte speichern soll. Sie können Protokolle in jeden Bucket speichern lassen, der sich in der gleichen Region wie der Quell-Bucket befindet, einschließlich des Quell-Buckets selbst.

  Wir empfehlen, die Zugriffsprotokolle in einem anderen Bucket zu speichern, damit Sie die Protokolle leicht verwalten können. Wenn Sie Zugriffsprotokolle im Quell-Bucket speichern, empfehlen wir Ihnen, ein Präfix für alle Protokollobjektschlüssel anzugeben, damit die Objektnamen mit einer gemeinsamen Zeichenfolge beginnen und die Protokollobjekte leichter zu identifizieren sind.

  Wenn der Quell- und Ziel-Bucket derselbe sind, werden zusätzliche Protokolle für die Protokolle erstellt, die in den Bucket geschrieben werden. Dieses Verhalten ist möglicherweise nicht ideal für Ihren Anwendungsfall, da es zu einer geringfügigen Erhöhung Ihrer Speicherabrechnung führen könnte. Zusätzlich können die zusätzlichen Protokolle über Protokolle das Auffinden des gesuchten Protokolls erschweren.

  Schlüsselpräfixe sind auch nützlich, um zwischen Quell-Buckets zu unterscheiden, wenn mehrere Buckets im selben Ziel-Bucket protokolliert werden.

  Note

  Sowohl der Quell- als auch der Ziel-Bucket müssen demselben AWS-Konto gehören, und die Buckets müssen beide in derselben Region liegen.
- (Optional) Ein Präfix, das Amazon S3 allen Protokollobjektschlüsseln zuweist. Das Präfix macht es Ihnen einfacher, die Protokollobjekte zu finden.

  Wenn Sie beispielsweise den Präfixwert `logs/` angeben, beginnt jedes von Amazon S3 erstellte Protokollobjekt mit dem Präfix `logs/` im Schlüssel wie im folgenden Beispiel gezeigt:

  ```
  logs/2013-11-01-21-32-16-E568B2907131C0C0
  ```

  Das Schlüsselpräfix kann zudem hilfreich sein, wenn Sie die Protokolle löschen. Beispielsweise können Sie eine Lebenszyklus-Konfigurationsregel für Amazon S3 festlegen, nach der Objekte mit einem

bestimmten Schlüsselpräfix gelöscht werden. Weitere Informationen finden Sie unter Löschen von Amazon S3-Protokolldateien (p. 793).

- (Optional) Berechtigungen, um anderen Benutzern Zugriff auf die generierten Protokolle zu gewähren. Standardmäßig hat der Bucket-Eigentümer stets vollen Zugriff auf die Protokollobjekte. Optional können Sie auch anderen Benutzern Zugriff gewähren.

Weitere Informationen zum Aktivieren der Server-Zugriffsprotokollierung finden Sie unter Aktivieren der Protokollierung mithilfe der Konsole (p. 780) und Programmgesteuertes Aktivieren der Protokollierung (p. 781).

Zusätzliche Protokollierungsüberlegungen

Note

- Sowohl der Quell- als auch der S3-Ziel-Bucket müssen demselben AWS-Konto gehören, und die S3-Buckets müssen beide in derselben Region liegen.
- Amazon S3 unterstützt nur das Gewähren von Rechten zur Bereitstellung von Zugriffsprotokollen über eine Bucket-ACL, nicht über eine Bucket-Richtlinie.
- Das Hinzufügen von Verweigerungsbedingungen zu einer Bucket-Richtlinie hindert Amazon S3 möglicherweise an der Bereitstellung von Zugriffsprotokollen.
- Die Standard-Bucket-Verschlüsselung am Zielbucket darf nur verwendet werden, wenn AES256 (SSE-S3) ausgewählt ist. Die SSE-KMS-Verschlüsselung wird nicht unterstützt.

Protokollobjekt-Schlüsselformat

Amazon S3 verwendet das folgende Objektschlüsselformat für die Protokollobjekte, die in den Ziel-Bucket hochgeladen werden:

```
TargetPrefixYYYY-mm-DD-HH-MM-SS-UniqueString/
```

Im Schlüssel sind `YYYY`, `mm`, `DD`, `HH`, `MM` und `SS` die Ziffern von Jahr, Monat, Tag, Stunde, Minute bzw. Sekunden des Zeitpunkts, an dem die Protokolldatei übermittelt wurde. Diese Datums- und Uhrzeitangaben entsprechen der Zeitzone UTC (Coordinated Universal Time).

Eine Protokolldatei, die zu einem bestimmten Zeitpunkt bereitgestellt wurde, kann Datensätze enthalten, die an einem beliebigen Zeitpunkt davor geschrieben wurden. Es lässt sich nicht feststellen, ob alle Protokoll-Datensätze für ein bestimmtes Zeitintervall bereitgestellt wurden oder nicht.

Die `UniqueString`-Komponente des Schlüssels verhindert, dass Dateien überschrieben werden. Sie hat keine Bedeutung und wird normalerweise von Protokollverarbeitungssoftware ignoriert.

Der nachfolgende Schrägstrich / ist erforderlich, um das Ende des Präfixes zu kennzeichnen.

Wie werden Protokolle bereitgestellt?

Amazon S3 sammelt periodisch Zugriffsprotokoll-Datensätze, fasst die Datensätze in Protokolldateien zusammen und lädt anschließend die Protokolldateien als Protokollobjekte in Ihren Ziel-Bucket hoch. Wenn Sie die Protokollierung bei mehreren Quell-Buckets aktivieren, die denselben Ziel-Bucket haben, werden die Zugriffsprotokolle für alle diese Quell-Buckets in diesen Ziel-Bucket geladen. Jedes Protokollobjekt gibt jedoch Zugriffsprotokoll-Datensätze für einen bestimmten Quell-Bucket aus.

Amazon S3 verwendet ein spezielles Protokollbereitstellungskonto zum Schreiben der Zugriffsprotokolle, das als „Protokollbereitstellungsgruppe“ bezeichnet wird. Für diese Protokolle gelten die normalen

Zugriffskontrollbeschränkungen. Sie müssen der Protokollbereitstellungsgruppe eine Schreibberechtigung für den Ziel-Bucket erteilen, indem Sie einen Rechteerteilungseintrag in die Zugriffskontrollliste (ACL) des Buckets einfügen. Wenn Sie die Amazon S3-Konsole zum Aktivieren der Protokollierung für einen Bucket verwenden, aktiviert die Konsole sowohl die Protokollierung für den Quell-Bucket als auch die Aktualisierung der ACL im Ziel-Bucket, um eine Schreibberechtigung für die Protokollbereitstellungsgruppe zu erteilen.

Best-Effort-Protokollbereitstellung der Server

Server-Zugriffsprotokoll-Datensätze werden auf Best-Effort-Basis bereitgestellt. Die meisten Anforderungen nach einem Bucket, der für die Protokollierung richtig konfiguriert ist, führen zu einem ausgelieferten Protokollsatz. Die meisten Protokollsätze werden innerhalb weniger Stunden nach der Aufnahme geliefert, können aber häufiger geliefert werden.

Die Vollständigkeit und Aktualität der Serverprotokollierung wird nicht garantiert. Der Protokolldatensatz für eine bestimmte Anforderung wird möglicherweise viel später bereitgestellt, als die Anforderung tatsächlich verarbeitet wurde; es kann auch sein, dass er gar nicht bereitgestellt wird. Der Zweck der Serverprotokolle besteht darin, Ihnen einen Überblick über die Art des Datenverkehrs zu und von Ihrem Bucket zu vermitteln. Es passiert selten, das Protokolldatensätze verloren gehen, aber die Server-Protokollierung ist nicht als vollständige Auflistung aller Anforderungen vorgesehen.

Aufgrund der Best-Effort-Natur der Serverprotokollierungsfunktion können die im AWS-Portal verfügbaren Nutzungsberichte (Abrechnungs- und Kostenverwaltungsberichte auf der AWS Management Console) eine oder mehrere ZugriffsAnforderungen enthalten, die nicht in einem bereitgestellten Serverprotokoll angezeigt werden.

Statusänderungen in der Bucket-Protokollierung werden mit der Zeit wirksam

Änderungen am Protokollierungsstatus eines Buckets benötigen einige Zeit, bis sie sich auf die Bereitstellung von Protokolldateien auswirken. Wenn Sie beispielsweise die Protokollierung für einen Bucket aktivieren, werden möglicherweise einige Anforderungen, die in der darauffolgenden Stunde gemacht werden, protokolliert, andere hingegen nicht. Wenn Sie den Ziel-Bucket für die Protokollierung von Bucket A zu Bucket B ändern, werden in der nächsten Stunde einige Protokolle möglicherweise zu Bucket A übermittelt, während andere zu dem neuen Ziel-Bucket B übermittelt werden. In jedem Fall werden die neuen Einstellungen letztendlich ohne weiteres Eingreifen Ihrerseits wirksam.

Aktivieren der Protokollierung mithilfe der Konsole

Informationen zur Aktivierung von Amazon S3-Serverzugriffsprotokollierung (p. 777) in der AWS Management Console finden Sie unter Wie aktiviere ich die Serverzugriffsprotokollierung für einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Wenn Sie die Protokollierung für einen Bucket aktivieren, aktiviert die Konsole die Protokollierung für den Quell-Bucket und fügt eine Berechtigung in der Zugriffskontrollliste (ACL) des Ziel-Buckets hinzu, wodurch eine Schreibberechtigung für die Protokollbereitstellungsgruppe gewährt wird.

Weitere Informationen zum Aktivieren der programmgesteuerten Protokollierung finden Sie unter Programmgesteuertes Aktivieren der Protokollierung (p. 781).

Weitere Informationen zum Protokoll-Datensatzformat einschließlich der Liste der Felder und deren Beschreibungen finden Sie unter Amazon S3-Serverzugriff-Protokollformat (p. 784).

Programmgesteuertes Aktivieren der Protokollierung

Sie können die programmgesteuerte Protokollierung aktivieren oder deaktivieren, indem Sie entweder die Amazon S3-API oder die AWS-SDKs verwenden. Zu diesem Zweck müssen Sie die Protokollierung im Bucket aktivieren und außerdem der Protokollbereitstellungsgruppe die Berechtigung erteilen, Protokolle in den Ziel-Bucket zu schreiben.

Themen

Aktivieren der Protokollierung

Zum Aktivieren der Protokollierung senden Sie eine Anforderung zur PUT-Bucket-Protokollierung, um die Protokollierungskonfiguration im Quell-Bucket hinzuzufügen. die Anforderung gibt den Ziel-Bucket und optional das Präfix an, das für alle Protokollobjektschlüssel verwendet werden soll. Das folgende Beispiel gibt `logbucket` als Ziel-Bucket und `logs/` als Präfix an.

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>logbucket</TargetBucket>
    <TargetPrefix>logs/</TargetPrefix>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Die Protokollobjekte werden vom Protokollbereitstellungskonto geschrieben und sind dessen Eigentum. Dem Bucket-Eigentümer werden volle Berechtigungen für die Protokollobjekte erteilt. Darüber hinaus können Sie optional auch anderen Benutzern Berechtigungen erteilen, so dass sie auf die Protokolle zugreifen können. Weitere Informationen finden Sie unter PUT-Bucket-Protokollierung.

Amazon S3 stellt außerdem eine API für die GET-Bucket-Protokollierung für den Abruf der Protokollierungskonfiguration für einen Bucket bereit. Um die Protokollierungskonfiguration zu löschen, senden Sie die Anforderung zur PUT-Bucket-Protokollierung einem leerem Eintrag für `BucketLoggingStatus`BucketLoggingStatus>.

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
</BucketLoggingStatus>
```

Sie können zum Aktivieren der Protokollierung in einem Bucket entweder die Amazon S3-API oder die AWS SDK Wrapper-Bibliotheken verwenden.

Erteilen von WRITE- und READ_ACP-Berechtigungen für die Protokollbereitstellungsgruppe

Amazon S3 schreibt die Protokolldateien in den Ziel-Bucket als Mitglied der vordefinierten Amazon S3-Gruppe für die Protokollbereitstellung. Für diese Protokolle gelten die normalen Zugriffskontrollbeschränkungen. Sie müssen dieser Gruppe `s3:GetObjectAcl`- und `s3:PutObject`-

Berechtigungen erteilen, indem Sie der Zugriffskontrollliste (ACL) des Ziel-Buckets Berechtigungen erteilen. Die Protokollbereitstellungsgruppe wird durch die folgende URL dargestellt.

```
http://acs.amazonaws.com/groups/s3/LogDelivery
```

Um `WRITE`- und `READ_ACP`-Berechtigungen zu erteilen, müssen Sie die folgenden Berechtigungen hinzufügen. Weitere Informationen zu ACLs finden Sie unter Zugriffsverwaltung mit ACLs (p. 479).

```
<Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
    </Grantee>
    <Permission>WRITE</Permission>
</Grant>
<Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
    </Grantee>
    <Permission>READ_ACP</Permission>
</Grant>
```

Beispiele zum Hinzufügen von programmgesteuerten ACL-Berechtigungen mithilfe von AWS-SDKs finden Sie unter Verwalten von ACLs mit AWS SDK for Java (p. 487) und Verwalten von ACLs mit AWS SDK für .NET (p. 489).

Beispiel: AWS SDK for .NET

Im folgenden C#-Beispiel wird die Protokollierung für einen Bucket aktiviert. Sie müssen zwei Buckets erstellen: einen Quell-Bucket und einen Ziel-Bucket. Im Beispiel wird zuerst der Protokollbereitstellungsgruppe die erforderliche Berechtigung zum Schreiben von Protokollen in den Ziel-Bucket erteilt und anschließend die Protokollierung im Quell-Bucket aktiviert. Weitere Informationen finden Sie unter Programmgesteuertes Aktivieren der Protokollierung (p. 781). Anweisungen zum Erstellen und Testen eines funktionierenden Beispiels finden Sie unter Ausführen der Amazon S3 .NET-Codebeispiele (p. 811).

Example

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ServerAccesLoggingTest
    {
        private const string bucketName = "*** bucket name for which to enable logging
 ***";
        private const string targetBucketName = "*** bucket name where you want access logs
 stored ***";
        private const string logObjectKeyPrefix = "Logs";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            EnableLoggingAsync().Wait();
```

```
        }

        private static async Task EnableLoggingAsync()
        {
            try
            {
                // Step 1 - Grant Log Delivery group permission to write log to the target
 bucket.
                await GrantPermissionsToWriteLogsAsync();
                // Step 2 - Enable logging on the source bucket.
                await EnableDisableLoggingAsync();
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }

        private static async Task GrantPermissionsToWriteLogsAsync()
        {
            var bucketACL = new S3AccessControlList();
            var aclResponse = client.GetACL(new GetACLRequest { BucketName =
 targetBucketName });
            bucketACL = aclResponse.AccessControlList;
            bucketACL.AddGrant(new S3Grantee { URI = "http://acs.amazonaws.com/groups/s3/
LogDelivery" }, S3Permission.WRITE);
            bucketACL.AddGrant(new S3Grantee { URI = "http://acs.amazonaws.com/groups/s3/
LogDelivery" }, S3Permission.READ_ACP);
            var setACLRequest = new PutACLRequest
            {
                AccessControlList = bucketACL,
                BucketName = targetBucketName
            };
            await client.PutACLAsync(setACLRequest);
        }

        private static async Task EnableDisableLoggingAsync()
        {
            var loggingConfig = new S3BucketLoggingConfig
            {
                TargetBucketName = targetBucketName,
                TargetPrefix = logObjectKeyPrefix
            };

            // Send request.
            var putBucketLoggingRequest = new PutBucketLoggingRequest
            {
                BucketName = bucketName,
                LoggingConfig = loggingConfig
            };
            await client.PutBucketLoggingAsync(putBucketLoggingRequest);
        }
    }
}
```

Zugehörige Ressourcen

- Amazon S3-Serverzugriffsprotokollierung (p. 777)

- AWS::S3::Bucket im AWS CloudFormation-Benutzerhandbuch

Amazon S3-Serverzugriff-Protokollformat

Dieser Abschnitt beschreibt die Amazon S3-Serverzugriffprotokolldateien.

Themen

Die Serverzugriff-Protokolldateien bestehen aus einer Reihe von durch Zeilenschaltungen voneinander getrennten Protokolldatensätzen. Jeder Protokolldatensatz stellt eine Anforderung dar und besteht aus durch Leerzeichen voneinander getrennter Felder. Nachfolgend wird ein Beispielprotokoll mit sechs Protokolldatensätzen gezeigt.

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 awsexamplebucket1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 3E57427F3EXAMPLE
 REST.GET.VERSIONING - "GET /awsexamplebucket1?versioning HTTP/1.1" 200 - 113 - 7 -
 "-" "S3Console/0.4" - s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/
XV/VLi31234= SigV2 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader awsexamplebucket1.s3.us-
west-1.amazonaws.com TLSV1.1
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 awsexamplebucket1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 891CE47D2EXAMPLE
 REST.GET.LOGGING_STATUS - "GET /awsexamplebucket1?logging HTTP/1.1" 200 - 242
 - 11 - "-" "S3Console/0.4" - 9vKBE6vMhrNiWHZmb2L0mXOcqPGzQOI5XLnCtZNPxev+Hf
+7tpT6sxDwDty4LHBUOZJG96N1234= SigV2 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader
 awsexamplebucket1.s3.us-west-1.amazonaws.com TLSV1.1
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 awsexamplebucket1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be A1206F460EXAMPLE
 REST.GET.BUCKETPOLICY - "GET /awsexamplebucket1?policy HTTP/1.1" 404
 NoSuchBucketPolicy 297 - 38 - "-" "S3Console/0.4" - BNaBsXZQQDbssi6xMBdBU2sLt
+Yf5kZDmeBUP35sFoKa3sLLeMC78iwEIWxs99CRUrbS4n11234= SigV2 ECDHE-RSA-AES128-GCM-SHA256
 AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com TLSV1.1
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 awsexamplebucket1 [06/Feb/2019:00:01:00 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 7B4A0FABBEXAMPLE
 REST.GET.VERSIONING - "GET /awsexamplebucket1?versioning HTTP/1.1" 200 - 113 - 33 - "-"
 "S3Console/0.4" - Ke1bUcazaN1jWuUlPJaxF64cQVpUEhoZKEG/hmy/gijN/I1DeWqDfFvnpybfEseEME/
u7ME1234= SigV2 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader awsexamplebucket1.s3.us-
west-1.amazonaws.com TLSV1.1
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 awsexamplebucket1 [06/Feb/2019:00:01:57 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DD6CC733AEXAMPLE REST.PUT.OBJECT s3-dg.pdf "PUT /awsexamplebucket1/
s3-dg.pdf HTTP/1.1" 200 - - 4406583 41754 28 "-" "S3Console/0.4" -
 10S62Zv81kBW7BB6SX4XJ48o6kpcl6LPwEoizZQQxJd5qDSCTLX0TgS37kYUBKQW3+bPdrg1234= SigV4 ECDHE-
RSA-AES128-SHA AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com TLSV1.1
```

Note

Jedes Feld kann auf - gesetzt werden, um darauf hinzuweisen, dass die Daten unbekannt oder nicht verfügbar waren, oder dass das Feld für diese Anforderung nicht relevant war.

In der folgenden Liste werden die wichtigsten Protokolldatensatzfelder beschrieben.

Bucket-Eigentümer

Die kanonische Benutzer-ID des Eigentümer des Quell-Buckets. Die kanonische Benutzer-ID ist eine andere Form der AWS-Konto-ID. Weitere Informationen zur kanonischen Benutzer-ID finden Sie unter AWS-Konto-Kennungen. Informationen darüber, wo Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Suchen Ihrer kanonischen Benutzer-ID für das Konto.

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Bucket

Der Name des Buckets, für den die Anforderung verarbeitet wurde. Wenn das System eine fehlerhaft aufgebaute Anforderung erhält und den Bucket nicht bestimmen kann, erscheint die Anforderung nicht in einem Serverzugriffsprotokoll.

Beispieleintrag

```
awsexamplebucket1
```

Zeit

Die Uhrzeit, zu der die Anforderung empfangen wurde. Diese Datums- und Uhrzeitangaben entsprechen der Zeitzone UTC (Coordinated Universal Time). Das Format unter Verwendung der `strftime()`-Terminologie, nämlich: `[%d/%b/%Y:%H:%M:%S %z]`

Beispieleintrag

```
[06/Feb/2019:00:00:38 +0000]
```

Externe IP

Die offensichtliche Internetadresse des Auftraggebers. Auf dem Weg vorhandene Proxy-Server und Firewalls könnten die tatsächliche Adresse des Computers verbergen, der die Anforderung gestellt hat.

Beispieleintrag

```
192.0.2.3
```

Auftraggeber

Die kanonische Benutzer-ID des Auftraggebers, oder - für nicht authentifizierte Anforderungen. War der Auftraggeber ein IAM-Benutzer, gibt dieses Feld den IAM-Benutzernamen des Auftraggebers zurück, zusammen mit dem AWS-Root-Konto, zu dem der IAM-Benutzer gehört. Diese ID ist dieselbe, die für den Zugriff zu Kontrollzwecken verwendet wird.

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Anforderungs-ID

Eine von Amazon S3 generierte Zeichenfolge, die jede Anforderung eindeutig identifiziert.

Beispieleintrag

```
3E57427F33A59F07
```

Operation

Der hier aufgelistete Operation wird als `SOAP.operation`, `REST.HTTP_method.resource_type`, `WEBSITE.HTTP_method.resource_type`, `BATCH.DELETE.OBJECT` oder `S3.action.resource_type` für Lebenszyklusaktionen deklariert.

Beispieleintrag

```
REST.PUT.OBJECT
```

Schlüssel

Der "Schlüssel"-Anteil der Anforderung, URL-codiert, oder "-", wenn die Operation keinen Schlüsselparameter entgegennimmt.

Beispieleintrag

```
/photos/2019/08/puppy.jpg
```

Anforderungs-URI

Der Teil der Anforderungs-URI der HTTP-Anforderungsmeldung.

Beispieleintrag

```
"GET /awsexamplebucket1/photos/2019/08/puppy.jpg?x-foo=bar HTTP/1.1"
```

HTTP-Status

Der numerische HTTP-Statuscode der Antwort.

Beispieleintrag

```
200
```

Fehlercode

Der Fehlercode (p. 770) in Amazon S3 oder „-“, wenn kein Fehler aufgetreten ist.

Beispieleintrag

```
NoSuchBucket
```

Gesendete Bytes

Die Anzahl der in der Antwort gesendeten Bytes, ausgenommen HTTP-Protokoll-Overhead, oder "-", falls null.

Beispieleintrag

```
2662992
```

Objektgröße

Die Gesamtgröße des betreffenden Objekts.

Beispieleintrag

```
3462992
```

Gesamtzeit

Die Anzahl der Millisekunden, wie lange die Anforderung aus Perspektive des Servers unterwegs war. Dieser Wert wird ab der Zeit gemessen, zu der Ihre Anforderung empfangen wurde, bis zu der Zeit, zu der das letzte Byte der Antwort gesendet wurde. Messungen aus der Perspektive des Clients dauern möglicherweise länger aufgrund der Netzwerklatenz.

Beispieleintrag

```
70
```

Umschlagzeit

Die Anzahl der Millisekunden, wie lange Amazon S3 gebraucht hat, Ihre Anforderung zu verarbeiten. Dieser Wert wird ab der Zeit gemessen, zu der das letzte Byte Ihrer Anforderung empfangen wurde, bis zu der Zeit, zu der das erste Byte der Antwort gesendet wurde.

Beispieleintrag

```
10
```

Referer

Der Wert des HTTP Referrer-Headers, falls vorhanden. HTTP-Benutzeragenten (z. B. Browser) setzen diesen Header normalerweise auf die URL der verlinkenden oder einbettenden Seite, wenn eine Anforderung erfolgt.

Beispieleintrag

```
"http://www.amazon.com/webservices"
```

User-Agent

Der Wert des HTTP-Benutzeragenten-Headers.

Beispieleintrag

```
"curl/7.15.1"
```

Versions-ID

Die Versions-ID der Anforderung, oder "-", wenn die Operation keinen `versionId`-Parameter entgegennimmt.

Beispieleintrag

```
3HL4kqtJvjVBH40Nrjfkd
```

Host-ID

Die x-amz-id-2 oder die erweiterte Amazon S3-Anforderungs-ID.

Beispieleintrag

```
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234=
```

Signaturversion

Die Signaturversion, `SigV2` oder `SigV4`, die für die Authentifizierung der Anforderung verwendet wurde, bzw. ein - für nicht authentifizierte Anforderungen.

Beispieleintrag

```
SigV2
```

Cipher Suite

Das Secure Sockets Layer(SSL)-Verschlüsselungsverfahren, das für die HTTPS-Anforderung ausgehandelt wurde bzw. ein - für HTTP.

Beispieleintrag

```
ECDHE-RSA-AES128-GCM-SHA256
```

Authentifizierungstyp

Die Art der verwendeten Anforderungsauthentifizierung, `AuthHeader` für Authentifizierungsköpfe, `QueryString` für die Anforderungszeichenfolge (vorsignierte URL) oder ein - für nicht authentifizierte Anforderungen.

Beispieleintrag

```
AuthHeader
```

Host Header

Der für die Verbindung zu Amazon S3 verwendete Endpunkt

Beispieleintrag

```
s3.us-west-2.amazonaws.com
```

Einige ältere Regionen unterstützen Legacy-Endpunkte. Möglicherweise sehen Sie diese Endpunkte in Ihren Serverzugriffsprotokollen oder CloudTrail-Protokollen. Weitere Informationen finden Sie unter Legacy-Endpunkte (p. 53). Eine vollständige Liste der Amazon S3-Regionen und -Endpunkte finden Sie unter Amazon S3-Regionen und -Endpunkte in der Allgemeinen AWS-Referenz.

TLS-Version

Die vom Client ausgehandelte Transport Layer Security(TLS)-Version. Einer der folgenden Werte: `TLSv1`, `TLSv1.1`, `TLSv1.2`; oder -, wenn TLS nicht verwendet wurde.

Beispieleintrag

```
TLSv1.2
```

Zusätzliche Protokollierung für Kopieroperationen

Eine Kopieroperation umfasst ein `GET` und ein `PUT`. Aus diesem Grund zeichnen wir für eine Kopieroperation zwei Datensätze auf. Die obige Tabelle beschreibt die Felder für den `PUT`-Teil der Operation. Die folgende Liste beschreibt die Felder in dem Datensatz, die sich auf den `GET`-Teil der Kopieroperation beziehen.

Bucket-Eigentümer

Die kanonische Benutzer-ID des Buckets, der das kopierte Objekt speichert. Die kanonische Benutzer-ID ist eine andere Form der AWS-Konto-ID. Weitere Informationen zur kanonischen Benutzer-ID finden Sie unter AWS-Konto-Kennungen. Informationen darüber, wo Sie die kanonische Benutzer-ID für Ihr Konto finden, finden Sie unter Suchen Ihrer kanonischen Benutzer-ID für das Konto.

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Bucket

Die Name des Buckets, der das kopierte Objekt speichert.

Beispieleintrag

```
awsexamplebucket1
```

Zeit

Die Uhrzeit, zu der die Anforderung empfangen wurde. Diese Datums- und Uhrzeitangaben entsprechen der Zeitzone UTC (Coordinated Universal Time). Das Format unter Verwendung der `strftime()`-Terminologie, nämlich: `[%d/%B/%Y:%H:%M:%S %z]`

Beispieleintrag

```
[06/Feb/2019:00:00:38 +0000]
```

Externe IP

Die offensichtliche Internetadresse des Auftraggebers. Auf dem Weg vorhandene Proxy-Server und Firewalls könnten die tatsächliche Adresse des Computers verbergen, der die Anforderung gestellt hat.

Beispieleintrag

```
192.0.2.3
```

Auftraggeber

Die kanonische Benutzer-ID des Auftraggebers, oder - für nicht authentifizierte Anforderungen. War der Auftraggeber ein IAM-Benutzer, gibt dieses Feld den IAM-Benutzernamen des Auftraggebers

zurück, zusammen mit dem AWS-Root-Konto, zu dem der IAM-Benutzer gehört. Diese ID ist dieselbe, die für den Zugriff zu Kontrollzwecken verwendet wird.

Beispieleintrag

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Anforderungs-ID

Eine von Amazon S3 generierte Zeichenfolge, die jede Anforderung eindeutig identifiziert.

Beispieleintrag

```
3E57427F33A59F07
```

Operation

Die hier aufgeführte Operation ist deklariert al `SOAP.operation`, `REST.HTTP_method.resource_type`, `WEBSITE.HTTP_method.resource_type` oder `BATCH.DELETE.OBJECT`.

Beispieleintrag

```
REST.COPY.OBJECT_GET
```

Schlüssel

Der "Schlüssel" des kopierten Objekts, oder "-", wenn die Operation keinen Schlüsselparameter entgegennimmt.

Beispieleintrag

```
/photos/2019/08/puppy.jpg
```

Anforderungs-URI

Der Teil der Anforderungs-URI der HTTP-Anforderungsmeldung.

Beispieleintrag

```
"GET /awsexamplebucket1/photos/2019/08/puppy.jpg?x-foo=bar"
```

HTTP-Status

Der numerische HTTP-Statuscode des `GET`-Teils der Kopieroperation.

Beispieleintrag

```
200
```

Fehlercode

Der Fehlercode (p. 770) in Amazon S3 des `GET`-Teils der Kopieroperation oder „-“, wenn kein Fehler aufgetreten ist.

Beispieleintrag

```
NoSuchBucket
```

Gesendete Bytes

Die Anzahl der in der Antwort gesendeten Bytes, ausgenommen HTTP-Protokoll-Overhead, oder "-", falls null.

Beispieleintrag

```
2662992
```

Objektgröße

Die Gesamtgröße des betreffenden Objekts.

Beispieleintrag

```
3462992
```

Gesamtzeit

Die Anzahl der Millisekunden, wie lange die Anforderung aus Perspektive des Servers unterwegs war. Dieser Wert wird ab der Zeit gemessen, zu der Ihre Anforderung empfangen wurde, bis zu der Zeit, zu der das letzte Byte der Antwort gesendet wurde. Messungen aus der Perspektive des Clients dauern möglicherweise länger aufgrund der Netzwerklatenz.

Beispieleintrag

```
70
```

Umschlagzeit

Die Anzahl der Millisekunden, wie lange Amazon S3 gebraucht hat, Ihre Anforderung zu verarbeiten. Dieser Wert wird ab der Zeit gemessen, zu der das letzte Byte Ihrer Anforderung empfangen wurde, bis zu der Zeit, zu der das erste Byte der Antwort gesendet wurde.

Beispieleintrag

```
10
```

Referer

Der Wert des HTTP Referrer-Headers, falls vorhanden. HTTP-Benutzeragenten (z. B. Browser) setzen diesen Header normalerweise auf die URL der verlinkenden oder einbettenden Seite, wenn eine Anforderung erfolgt.

Beispieleintrag

```
"http://www.amazon.com/webservices"
```

User-Agent

Der Wert des HTTP-Benutzeragenten-Headers.

Beispieleintrag

```
"curl/7.15.1"
```

Versions-ID

Die Versions-ID des kopierten Objekts, oder „-“, wenn der `x-amz-copy-source`-Header keinen `versionId`-Parameter als Teil der Kopierquelle angegeben hat.

Beispieleintrag

```
3HL4kqtJvjVBH40Nrjfkd
```

Host-ID

Die x-amz-id-2 oder die erweiterte Amazon S3-Anforderungs-ID.

Beispieleintrag

```
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234=
```

Signaturversion

Die Signaturversion, `SigV2` oder `SigV4`, die für die Authentifizierung der Anforderung verwendet wurde, bzw. ein - für nicht authentifizierte Anforderungen.

Beispieleintrag

```
SigV2
```

Cipher Suite

Das Secure Sockets Layer(SSL)-Verschlüsselungsverfahren, das für die HTTPS-Anforderung ausgehandelt wurde bzw. ein - für HTTP.

Beispieleintrag

```
ECDHE-RSA-AES128-GCM-SHA256
```

Authentifizierungstyp

Die Art der verwendeten Anforderungsauthentifizierung, `AuthHeader` für Authentifizierungsköpfe, `QueryString` für die Anforderungszeichenfolge (vorsignierte URL) oder ein - für nicht authentifizierte Anforderungen.

Beispieleintrag

```
AuthHeader
```

Host Header

Der für die Verbindung mit Amazon S3 verwendete Endpunkt.

Beispieleintrag

```
s3.us-west-2.amazonaws.com
```

Einige ältere Regionen unterstützen Legacy-Endpunkte. Möglicherweise sehen Sie diese Endpunkte in Ihren Serverzugriffsprotokollen oder CloudTrail-Protokollen. Weitere Informationen finden Sie unter Legacy-Endpunkte (p. 53). Eine vollständige Liste der Amazon S3-Regionen und -Endpunkte finden Sie unter Amazon S3-Regionen und -Endpunkte in der Allgemeinen AWS-Referenz.

TLS-Version

Die vom Client ausgehandelte Transport Layer Security(TLS)-Version. Einer der folgenden Werte: `TLSv1`, `TLSv1.1`, `TLSv1.2`; oder `-`, wenn TLS nicht verwendet wurde.

Beispieleintrag

```
TLSv1.2
```

Benutzerdefinierte Zugriffsprotokollinformationen

Sie können benutzerdefinierte Informationen im Zugriffsprotokolldatensatz für eine Anforderung speichern, indem Sie der URL für die Anforderung einen benutzerdefinierten Abfragezeichenfolgeparameter hinzufügen. Amazon S3 ignoriert Abfragezeichenfolgenparameter, die mit „x“ beginnen, fügt diese Parameter jedoch in den Zugriffsprotokolldatensatz für die Anforderung als Teil des Protokolldatensatzfelds `Request-URI` ein. Beispielsweise verhält sich die Anforderung `GET` für „s3.amazonaws.com/awsexamplebucket1/photos/2019/08/puppy.jpg?x-user=johndoe“ genauso, wie sich die gleiche Anforderung für „s3.amazonaws.com/awsexamplebucket1/photos/2019/08/puppy.jpg“ verhält, abgesehen davon, dass die Zeichenfolge „x-user=johndoe“ in das Feld `Request-URI` des entsprechenden Protokolldatensatzes eingefügt wird. Diese Funktionalität steht nur auf der REST-Schnittstelle zur Verfügung.

Aspekte zur Programmierung des erweiterbaren Serverzugriff-Protokollformats

Möglicherweise erweitern wir von Zeit zu Zeit das Zugriffsprotokoll-Datensatzformat, indem wir am Ende jeder Zeile neue Felder hinzufügen. Code, der Serverzugriffprotokolle analysiert, muss so geschrieben werden, dass er angefügte Felder verarbeiten kann, die er nicht versteht.

Löschen von Amazon S3-Protokolldateien

Ein S3-Bucket mit aktivierter Server-Zugriffsprotokollierung kann im Laufe der Zeit viele Server-Protokollobjekte ansammeln. Möglicherweise benötigt Ihre Anwendung diese Zugriffsprotokolle für einen bestimmten Zeitraum nach deren Erstellung; danach möchten Sie sie ggf. löschen. Sie können die Amazon S3-Lebenszykluskonfiguration zum Festlegen von Regeln verwenden, sodass Amazon S3 diese Objekte am Ende ihres Lebenszyklus automatisch zum Löschen in eine Warteschlange einfügt.

Sie können eine Lebenszyklus-Konfiguration für eine Teilmenge von Objekten in Ihrem S3-Bucket definieren, indem Sie ein gemeinsames Präfix verwenden (d. h. Objekte, deren Namen mit einer gemeinsamen Zeichenfolge beginnen). Wenn Sie in Ihrer Konfiguration für die Server-Zugriffsprotokollierung ein Präfix angegeben haben, können Sie eine Lebenszyklus-Konfigurationsregel festlegen, um Protokollobjekte mit diesem Präfix zu löschen. Wenn Ihre Protokollobjekte beispielsweise das Präfix `logs/`haben, können Sie eine Lebenszyklus-Konfigurationsregel setzen, um nach einer bestimmten

Zeitspanne alle Objekte im Bucket zu löschen, die das Präfix `/logs` haben. Weitere Informationen zur Lebenszykluskonfiguration finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).

Zugehörige Ressourcen

Amazon S3-Serverzugriffsprotokollierung (p. 777)

Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von -Anforderungen

Sie können Amazon S3-Anforderungen mit Amazon S3-Zugriffsprotokollen identifizieren.

Note

- Wir empfehlen, AWS CloudTrail-Datenereignisse anstelle von Amazon S3-Zugriffsprotokollen zu verwenden. CloudTrail-Datenereignisse lassen sich einfach einrichten und enthalten mehr Informationen. Weitere Informationen finden Sie unter Verwenden von AWS CloudTrail zum Identifizieren von Amazon S3-Anforderungen (p. 758).
- Je nachdem, wie viele Zugriffsanforderungen Sie erhalten, sind möglicherweise mehr Ressourcen und/oder mehr Zeit für die Analyse Ihrer Protokolle erforderlich.

Themen

Aktivieren von Amazon S3-Zugriffsprotokollen für Anforderungen

Wir empfehlen, in jeder AWS-Region mit S3-Buckets einen speziellen Protokoll-Bucket zu erstellen. Veranlassen Sie dann die Bereitstellung des Amazon S3-Zugriffsprotokolls in diesem S3-Bucket.

Example – Aktivieren von Zugriffsprotokollen für fünf Buckets in zwei Regionen

In diesem Beispiel werden die folgenden fünf Buckets verwendet:

- `1-awsexamplebucket1-us-east-1`
- `2-awsexamplebucket1-us-east-1`
- `3-awsexamplebucket1-us-east-1`
- `1-awsexamplebucket1-us-west-2`
- `2-awsexamplebucket1-us-west-2`

1. Erstellen Sie zwei Protokoll-Buckets in den folgenden Regionen:

- `awsexamplebucket1-logs-us-east-1`
- `awsexamplebucket1-logs-us-west-2`

2. Aktivieren Sie dann folgendermaßen die Amazon S3-Zugriffsprotokolle:

   - `1-awsexamplebucket1-us-east-1` protokolliert im S3-Bucket `awsexamplebucket1-logs-us-east-1` mit dem Präfix `1-awsexamplebucket1-us-east-1`
   - `2-awsexamplebucket1-us-east-1` protokolliert im S3-Bucket `awsexamplebucket1-logs-us-east-1` mit dem Präfix `2-awsexamplebucket1-us-east-1`
   - `1-awsexamplebucket1-us-east-1` protokolliert im S3-Bucket `awsexamplebucket1-logs-us-east-1` mit dem Präfix `3-awsexamplebucket1-us-east-1`
   - `1-awsexamplebucket1-us-west-2` protokolliert im S3-Bucket `awsexamplebucket1-logs-us-west-2` mit dem Präfix `1-awsexamplebucket1-us-west-2`
   - `2-awsexamplebucket1-us-west-2` protokolliert im S3-Bucket `awsexamplebucket1-logs-us-west-2` mit dem Präfix `2-awsexamplebucket1-us-west-2`

3. Anschließend können Sie die Amazon S3-Zugriffsprotokolle mit den folgenden Methoden aktivieren:

   - Verwenden der AWS-Managementkonsole oder
   - Programmgesteuertes Aktivieren der Protokollierung (p. 781) oder
   - Verwendung des AWS-CLI-Befehls put-bucket-logging zur programmgesteuerten Aktivierung der Zugriffsprotokolle für einen Bucket mit den folgenden Befehlen:

     1. Gewähren Sie zunächst mit `put-bucket-acl` die Amazon S3-Berechtigung:

```
 aws s3api put-bucket-acl --bucket awsexamplebucket1-logs  --grant-write
 URI=http://acs.amazonaws.com/groups/s3/LogDelivery --grant-read-acp URI=http://
acs.amazonaws.com/groups/s3/LogDelivery
```

     2. Weisen Sie dann die Protokollierungsrichtlinie zu:

```
aws s3api put-bucket-logging --bucket awsexamplebucket1 --bucket-logging-status
 file://logging.json
```

     `Logging.json` ist ein JSON-Dokument im aktuellen Ordner, das die Protokollierungsrichtlinie enthält:

```
 {
     "LoggingEnabled": {
         "TargetBucket": "awsexamplebucket1-logs",
         "TargetPrefix": "awsexamplebucket1/",
         "TargetGrants": [
              {
                 "Grantee": {
                     "Type": "AmazonCustomerByEmail",
                     "EmailAddress": "user@example.com"
                  },
                 "Permission": "FULL_CONTROL"
              }
          ]
      }
  }
```

Note

Der Befehl `put-bucket-acl` wird benötigt, um dem System zur Bereitstellung der Amazon S3-Protokolle die erforderlichen Berechtigungen („write-acp“ und „read-acp“) zu gewähren.

3. Verwenden Sie ein Bash-Skript, um allen Buckets im Konto die Zugriffsprotokollierung hinzuzufügen:

```
  loggingBucket='awsexamplebucket1-logs'
  region='us-west-2'


  # Create Logging bucket
  aws s3 mb s3://$loggingBucket --region $region

  aws s3api put-bucket-acl --bucket $loggingBucket --grant-write URI=http://
acs.amazonaws.com/groups/s3/LogDelivery --grant-read-acp URI=http://
acs.amazonaws.com/groups/s3/LogDelivery

  # List buckets in this account
  buckets="$(aws s3 ls | awk '{print $3}')"

  # Put bucket logging on each bucket
  for bucket in $buckets
      do printf '{
     "LoggingEnabled": {
         "TargetBucket": "%s",
         "TargetPrefix": "%s/"
          }
      }' "$loggingBucket" "$bucket"  > logging.json
      aws s3api put-bucket-logging --bucket $bucket --bucket-logging-status
 file://logging.json
      echo "$bucket done"
  done

  rm logging.json

  echo "Complete"
```

Note

Dies funktioniert nur, wenn sich alle Buckets in derselben Region befinden. Sind die Buckets auf mehrere Regionen verteilt, müssen Sie das Skript anpassen.

Abfragen von Amazon S3-Zugriffsprotokollen für Anforderungen

Amazon S3 speichert Server-Zugriffsprotokolle als Objekte in einem S3-Bucket. Es ist häufig einfacher, ein Tool zu verwenden, das die Protokolle in Amazon S3 analysieren kann. Athena unterstützt die Analyse von S3-Objekten und kann verwendet werden, um Amazon S3-Zugriffsprotokolle abzufragen.

Example

Das folgende Beispiel zeigt, wie Sie Amazon S3-Server-Zugriffsprotokolle in Amazon Athena abfragen können.

Note

Um die Amazon S3-Position in einer Athena-Abfrage anzugeben, benötigen Sie den Namen des Ziel-Buckets und das Zielpräfix: `s3://awsexamplebucket1-logs/prefix/`

1. Öffnen Sie die Athena-Konsole unter https://console.aws.amazon.com/athena/.
2. Führen Sie im Abfrage-Editor einen Befehl wie den folgenden aus:

```
create database s3_access_logs_db
```

Note

Es hat sich bewährt, die Datenbank in der AWS-Region zu erstellen, in der sich der S3-Bucket befindet.

3. Führen Sie im Abfrage-Editor einen Befehl wie den folgenden aus, um in der in Schritt 2 erstellten Datenbank ein Tabellenschema zu erstellen. Die Datentypwerte `STRING` und `BIGINT` sind die Zugriffsprotokolleigenschaften. Sie können diese Eigenschaften in Athena abfragen. Geben Sie für `LOCATION` wie oben erwähnt den Pfad von S3-Bucket und Präfix ein.

```
  CREATE EXTERNAL TABLE IF NOT EXISTS s3_access_logs_db.mybucket_logs(
     BucketOwner STRING,
     Bucket STRING,
     RequestDateTime STRING,
     RemoteIP STRING,
     Requester STRING,
     RequestID STRING,
     Operation STRING,
     Key STRING,
     RequestURI_operation STRING,
     RequestURI_key STRING,
     RequestURI_httpProtoversion STRING,
     HTTPstatus STRING,
     ErrorCode STRING,
     BytesSent BIGINT,
     ObjectSize BIGINT,
     TotalTime STRING,
     TurnAroundTime STRING,
     Referrer STRING,
     UserAgent STRING,
     VersionId STRING,
     HostId STRING,
     SigV STRING,
     CipherSuite STRING,
     AuthType STRING,
     EndPoint STRING,
     TLSVersion STRING
  )
  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
  WITH SERDEPROPERTIES (
              'serialization.format' = '1', 'input.regex' = '([^ ]*) ([^ ]*)
              \\[(.*?)\\] ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) \\\"([^ ]*) ([^ ]*)
 (- |[^ ]*)
              \\\" (-|[0-9]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) (\"[^
\"]*\") ([^ ]*)
              (?: ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*))?.*$' )
     LOCATION 's3://awsexamplebucket1-logs/prefix'
```

4. Wählen Sie im Navigationsbereich unter Database (Datenbank) die Datenbank aus.

5. Wählen Sie unter Tables (Tabellen)neben dem Namen der Tabelle Preview table (Tabellenvorschau) aus.

   Im Fensterbereich Results (Ergebnisse) sollten Daten aus den Server-Zugriffsprotokollen angezeigt werden, also `bucketowner`, `bucket`, `requestdatetime` usw. Dies bedeutet, dass die Athena-Tabelle erstellt wurde. Sie können die Amazon S3-Server-Zugriffsprotokolle jetzt abfragen.

Example – Anzeigen, wer und um welche Uhrzeit (Zeitstempel, IP-Adresse und IAM-Benutzer) ein Objekt gelöscht hat

```
SELECT RequestDateTime, RemoteIP, Requester, Key
FROM s3_access_logs_db.mybucket_logs
WHERE key = 'images/picture.jpg' AND operation like '%DELETE%';
```

Example – Anzeigen aller von einem IAM-Benutzer ausgeführten Operationen

```
SELECT *
FROM s3_access_logs_db.mybucket_logs
WHERE requester='arn:aws:iam::123456789123:user/user_name';
```

Example – Anzeigen aller Operationen, die in einem bestimmten Zeitraum für ein Objekt ausgeführt wurden

```
SELECT *
FROM s3_access_logs_db.mybucket_logs
WHERE Key='prefix/images/picture.jpg'
    AND parse_datetime(RequestDateTime,'dd/MMM/yyyy:HH:mm:ss Z')
    BETWEEN parse_datetime('2017-02-18:07:00:00','yyyy-MM-dd:HH:mm:ss')
    AND parse_datetime('2017-02-18:08:00:00','yyyy-MM-dd:HH:mm:ss');
```

Example – Anzeigen Menge der von einer bestimmten IP-Adresse in einem bestimmten Zeitraum übertragenen Daten

```
SELECT SUM(bytessent) AS uploadTotal,
      SUM(objectsize) AS downloadTotal,
      SUM(bytessent + objectsize) AS Total
FROM s3_access_logs_db.mybucket_logs
WHERE RemoteIP='1.2.3.4'
AND parse_datetime(RequestDateTime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2017-06-01','yyyy-MM-dd')
AND parse_datetime('2017-07-01','yyyy-MM-dd');
```

Note

Um die Aufbewahrungsdauer des Protokolls zu verkürzen, können Sie für den Bucket mit den Server-Zugriffsprotokollen eine Amazon S3-Lebenszyklusrichtlinie erstellen. Konfigurieren Sie die Lebenszyklusrichtlinie so, dass Protokolldateien regelmäßig entfernt werden. Dadurch wird die Menge an Daten reduziert, die Athena in einer Abfrage analysiert.

Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von Signature Version 2-Anforderungen

Die Amazon S3-Unterstützung für Signature Version 2 wird deaktiviert (veraltet). Nach akzeptiert Amazon S3 keine Anforderungen mit Signature Version 2 mehr, alle Anforderungen müssen also mit Signature Version 4 signiert werden. Sie können Signature Version 2-Anforderungen über die Amazon S3-Zugriffsprotokolle identifizieren

Note

- Wir empfehlen, AWS CloudTrail-Datenereignisse anstelle von Amazon S3-Zugriffsprotokollen zu verwenden. CloudTrail-Datenereignisse lassen sich einfach einrichten und enthalten mehr Informationen. Weitere Informationen finden Sie unter Verwenden von AWS CloudTrail zum Identifizieren von Amazon S3 Signature Version 2-Anforderungen (p. 761).

Example – Anzeigen aller Anforderer, die Signature Version 2-Datenverkehr senden

```
SELECT requester, Sigv, Count(Sigv) as SigCount
FROM s3_access_logs_db.mybucket_logs
GROUP BY requester, Sigv;
```

Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von Objektzugriffsanforderungen

Sie können Abfragen an Amazon S3-Serverzugriffsprotokolle verwenden, um Amazon S3-Objektzugriffsanforderungen für Operationen zu identifizieren, wie etwa GET, PUT und DELETE, und weitere Informationen über diese Anforderungen zu erkunden.

Das folgende Amazon Athena-Abfragebeispiel zeigt, wie Sie alle PUT-Objektanforderungen für Amazon S3 vom Serverzugriffsprotokoll abrufen.

Example – Anzeigen aller Anforderer, die PUT-Objektanforderungen in einem bestimmten Zeitraum senden

```
SELECT Bucket, Requester, RemoteIP, Key, HTTPStatus, ErrorCode, RequestDateTime
FROM s3_access_logs_db
WHERE Operation='REST.PUT.OBJECT' AND
parse_datetime(RequestDateTime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Das folgende Amazon Athena-Abfragebeispiel zeigt, wie Sie alle GET-Objektanforderungen für Amazon S3 vom Serverzugriffsprotokoll abrufen.

Example – Anzeigen aller Anforderer, die GET-Objektanforderungen in einem bestimmten Zeitraum senden

```
SELECT Bucket, Requester, RemoteIP, Key, HTTPStatus, ErrorCode, RequestDateTime
FROM s3_access_logs_db
WHERE Operation='REST.GET.OBJECT' AND
```

```
parse_datetime(RequestDateTime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Das folgende Amazon Athena-Abfragebeispiel zeigt, wie alle anonymen Anforderungen aus dem Serverzugriffsprotokoll in Ihre S3-Buckets gelangen.

Example — Anzeigen aller anonymen Anforderer, die in einem bestimmten Zeitraum Anforderungen an einen Bucket machen

```
SELECT Bucket, Requester, RemoteIP, Key, HTTPStatus, ErrorCode, RequestDateTime
FROM s3_access_logs_db.mybucket_logs
WHERE Requester IS NULL AND
parse_datetime(RequestDateTime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Note

- Sie können den Datumsbereich nach Belieben an Ihre Anforderungen anpassen.
- Diese Abfragebeispiele können auch für die Sicherheitsüberwachung nützlich sein. Sie können die Ergebnisse auf `PutObject`- oder `GetObject`-Aufrufe von unerwarteten oder nicht autorisierten IP-Adresen/Anforderern und zum Aufdecken anonymer Anforderungen an Ihre Buckets prüfen.
- Diese Abfrage ruft nur Informationen von der Zeit ab, zu der die Protokollierung aktiviert wurde.
- Informationen zur Verwendung von Amazon S3-AWS CloudTrail-Protokollen finden Sie unter Verwenden von AWS CloudTrail zum Identifizieren des Zugriffs auf Amazon S3-Objekte (p. 764).

Zugehörige Ressourcen

- Amazon S3-Serverzugriff-Protokollformat (p. 784)
- Abfragen von AWS-Service-Protokollen

Verwenden der AWS SDKs, CLI und Explorer

Sie können auch die AWS SDKs für die Entwicklung von Anwendungen mit Amazon S3 verwenden. Die AWS SDKs vereinfachen Ihre Programmierungsaufgaben, indem sie die zugrunde liegende REST API umhüllen. Die AWS Mobile SDKs und die AWS Amplify JavaScript-Bibliothek sind auch für die Erstellung von verbundenen mobilen und Webanwendungen mit AWS verfügbar.

In diesem Abschnitt finden Sie eine Übersicht über die Verwendung von AWS SDKs für die Entwicklung von Amazon S3-Anwendungen. Darüber hinaus beschreibt dieser Abschnitt, wie Sie die AWS SDK-Codebeispiele in dieser Anleitung testen können.

Themen

Neben den AWS SDKs stehen auch AWS Explorer für Visual Studio und Eclipse for Java IDE zur Verfügung. In diesem Fall werden die SDKs und die Explorer zusammen als AWS Toolkits bereitgestellt.

Sie können auch die AWS Command Line Interface (AWS CLI) für die Verwaltung von Amazon S3-Buckets und Objekten verwenden.

AWS Toolkit for Eclipse

Die AWS Toolkit for Eclipse beinhaltet AWS SDK for Java und AWS Explorer for Eclipse. Der AWS Explorer for Eclipse ist ein Open Source Plug-In für die Eclipse for Java IDE, die es den Entwicklern vereinfacht, Java-Anwendungen mit AWS zu entwickeln, zu debuggen und bereitzustellen. Die einfach zu bedienende GUI-Oberfläche gestattet Ihnen, auf Ihre AWS-Infrastruktur zuzugreifen und diese zu verwalten, einschließlich von Amazon S3. Sie können allgemeine Operationen ausführen, wie beispielsweise die Verwaltung Ihrer Buckets und Objekte oder die Einrichtung von IAM-Richtlinien bei der Entwicklung von Anwendungen, und all das innerhalb des Kontexts der Eclipse for Java IDE. Weitere Informationen zur Einrichtung finden Sie unter Einrichten des Toolkits. Beispiele für die Verwendung des Explorers finden Sie unter Zugreifen auf AWS Explorer.

AWS Toolkit for Visual Studio

AWS Explorer for Visual Studio ist eine Erweiterung von Microsoft Visual Studio, der es Entwicklern einfacher macht, .NET-Anwendungen mit Amazon Web Services zu entwickeln, zu debuggen und bereitzustellen. Die einfach zu bedienende GUI-Oberfläche gestattet Ihnen, auf Ihre AWS-Infrastruktur zuzugreifen und diese zu verwalten, einschließlich von Amazon S3. Sie können allgemeine Operationen ausführen, wie beispielsweise die Verwaltung Ihrer Buckets und Objekte oder die Einrichtung von IAM-Richtlinien bei der Entwicklung von Anwendungen, und all das innerhalb des Kontexts von Visual Studio. Weitere Informationen zur Einrichtung finden Sie unter Einrichten von AWS Toolkit for Visual Studio. Beispiele für die Verwendung von Amazon Amazon S3 unter Verwendung des Explorers finden Sie unter Verwenden von Amazon S3 im AWS Explorer.

AWS-SDKs

Sie können nur den SDK herunterladen. Weitere Informationen zum Herunterladen der SDK-Bibliotheken finden Sie unter Beispiel-Code-Bibliotheken.

AWS CLI

Die AWS CLI ist ein vereinheitlichtes Tool zur Verwaltung von AWS-Services, einschließlich von Amazon S3. Weitere Informationen zum Herunterladen von AWS CLI finden Sie unter AWS Command Line Interface.

Angabe der Signature-Version in der Anforderungsauthentifizierung

Amazon S3 unterstützt in den meisten AWS-Regionen nur AWS Signature Version 4. In einigen der älteren AWS-Regionen unterstützt Amazon S3 sowohl Signature Version 4 als auch Signature Version 2. Signature Version 2 wird jedoch deaktiviert (veraltet). Weitere Informationen zum Ende der Unterstützung für Signature Version 2 siehe AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet) (p. 804).

Eine Liste aller Amazon S3 Regionen und der von ihnen unterstützten Signature-Versionen finden Sie unter Regionen und Endpunkte in der Allgemeinen AWS-Referenz.

Für alle AWS-Regionen verwenden AWS SDKs standardmäßig Signature Version 4, um die Anforderungen zu authentifizieren. Bei Verwendung von AWS SDKs, die vor Mai 2016 veröffentlicht wurden, müssen Sie möglicherweise Signature Version 4 anfordern, wie in der folgenden Tabelle gezeigt.

SDK	Anforderung von Signature Version 4 für die Anforderungsauthentifizierung
AWS-CLI	Für das Standardprofil führen Sie den folgenden Befehl aus: `$ aws configure set default.s3.signature_version s3v4` Für ein benutzerdefiniertes Profil führen Sie den folgenden Befehl aus: `$ aws configure set profile.your_profile_name.s3.signature_version s3v4`
Java-SDK	Fügen Sie Ihrem Code Folgendes hinzu: `System.setProperty(SDKGlobalConfiguration.ENABLE_S3_SIGV4_SYSTEM_PROPERT "true");` Oder geben Sie in der Befehlszeile Folgendes an: `-Dcom.amazonaws.services.s3.enableV4`
JavaScript SDK	Setzen Sie den `signatureVersion`-Parameter auf `v4`, wenn Sie den Client konstruieren: `var s3 = new AWS.S3({signatureVersion: 'v4'});`

SDK	Anforderung von Signature Version 4 für die Anforderungsauthentifizierung
PHP SDK	Setzen Sie den Parameter `signature` auf `v4`, wenn Sie den Amazon S3 Service-Client für PHP SDK v2 erstellen: `<?php` `$client = S3Client::factory([` `'region' => 'YOUR-REGION',` `'version' => 'latest',` `'signature' => 'v4'` `]);` Setzen Sie den Parameter `signature_version` bei Verwendung des PHP SDK v3 im Rahmen der Erstellung des Amazon S3-Service-Clients auf `v4`: `<?php` `$s3 = new Aws\S3\S3Client([` `'version' => '2006-03-01',` `'region' => 'YOUR-REGION',` `'signature_version' => 'v4'` `]);`
Python-Boto SDK	Geben Sie Folgendes in der boto-Standardkonfigurationsdatei an: `[s3] use-sigv4 = True`
Ruby SDK	Ruby SDK – Version 1: Setzen Sie den `:s3_signature_version`-Parameter auf `:v4`, wenn Sie den Client konstruieren: `s3 = AWS::S3::Client.new(:s3_signature_version => :v4)` Ruby SDK – Version 3: Setzen Sie den `signature_version`-Parameter auf `v4`, wenn Sie den Client konstruieren: `s3 = Aws::S3::Client.new(signature_version: 'v4')`
.NET SDK	Fügen Sie den folgenden Code hinzu, bevor Sie den Amazon S3-Client erstellen: `AWSConfigsS3.UseSignatureVersion4 = true;` Oder fügen Sie der Konfigurationsdatei Folgendes hinzu: `<appSettings>` `<add key="AWS.S3.UseSignatureVersion4" value="true" />` `</appSettings>`

AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet)

Signature Version 2 wird in Amazon S3 deaktiviert (veraltet). Amazon S3 akzeptiert anschließend nur API-Anforderungen, die mit Signature Version 4 signiert wurden.

Dieser Abschnitt umfasst Antworten auf häufig gestellte Fragen im Bezug auf das Supportende für Signature Version 2.

Um was handelt es sich bei der Signature Version 2/4 und was versteht man unter dem Signieren einer Anforderung?

Der Signiervorgang mittels Signature Version 2 oder Signature Version 4 authentifiziert Ihre API-Anforderungen in Amazon S3. Durch das Signieren von Anforderungen ist Amazon S3 in der Lage, den Absender der Anforderung zu identifizieren und Ihre Anforderungen vor Angreifern zu schützen.

Weitere Informationen zum Signieren von AWS-Anforderungen finden Sie unter Signieren von AWS-API-Anforderungen in der AWS General Reference.

Welche Aktualisierung nehmen Sie vor?

Derzeit unterstützen wir Amazon S3 API-Anforderungen, die mithilfe von Signature Version 2 und Signature Version 4 signiert wurden. Nach dem akzeptiert Amazon S3 nur noch Anforderungen, die mit Signature Version 4 signiert wurden.

Weitere Informationen zum Signieren von AWS-Anforderungen finden Sie unter Änderungen in Signature Version 4 in der AWS General Reference.

Warum nehmen Sie die Aktualisierung vor?

Signature Version 4 bietet mehr Sicherheit, da hierbei im Gegensatz zum geheimen Zugriffsschlüssel ein Signaturschlüssel verwendet wird. Signature Version 4 wird derzeit in allen AWS-Regionen unterstützt, während Signature Version 2 nur von Regionen unterstützt wird, die vor dem Januar 2014 gestartet wurden. Dank dieser Aktualisierung sind wir in der Lage, in allen Regionen eine einheitlichere Erfahrung bereitzustellen.

Wie kann ich sicherstellen, dass ich Signature Version 4 verwende und welche Aktualisierungen benötige ich?

Normalerweise entscheidet das Tool oder das SDK auf Clientseite, welche Signaturversion zum Signieren der Anforderungen verwendet wird. Standardmäßig verwenden die aktuellen Versionen unsere AWS SDKs Signature Version 4. Setzen Sie sich mit dem entsprechenden Supportteam für Ihre Drittanbietersoftware in Verbindung, um zu erfahren, welche Version Sie benötigen. Um direkte REST-Aufrufe Amazon S3zu senden, müssen Sie Ihre Anwendung so anpassen, dass der Signiervorgang mithilfe von Signature Version 4 ausgeführt wird.

Weitere Informationen darüber, welche Version des AWS SDKs beim Wechsel zu Signature Version 4 zu verwenden ist, finden Sie unter Von Signature Version# 2 zur Signature Version 4 wechseln. (p. 805).

Weitere Informationen zur Verwendung von Signature Version 4 mit der Amazon S3 REST-API finden Sie unter Authentifizieren von Anforderungen (AWS Signature Version 4) in der Amazon Simple Storage Service API Reference.

Was passiert, wenn ich keine Aktualisierung durchführe?

Anforderungen, mit Signature Version 2 signiert wurden und nach dem gesendet werden, können dann nicht mehr mit Amazon S3 authentifiziert werden. Anforderer sehen eine Fehlermeldung, dass die Anforderung mit Signature Version 4 signiert sein muss.

Sollte ich Änderungen vornehmen, selbst wenn ich eine vorsignierte URL verwende, die es erforderlich macht, dass ich für mehr als sieben Tage signiere?

Wenn sie eine vorsignierte URL verwenden, für die Sie für mehr als sieben Tage signieren müssen, ist keine Maßnahme erforderlich. Sie können weiterhin AWS Signature Version 2 zum Signieren und Authentifizieren der vorsignierten URL verwenden. Wir werden uns darum kümmern und zusätzliche Einzelheiten zur Migration zu Signature Version 4 für ein vorab signiertes URL-Szenario bereitstellen.

Weitere Infos

- Weitere Informationen zur Verwendung von Signature Version 4 finden Sie unter Signieren von AWS-API-Anforderungen.
- Sehen Sie sich die Änderungen zwischen Signature Version 2 und Signature Version 4 unter Änderungen in Signature Version 4 an.
- Lesen Sie den Beitrag AWS Signature Version 4 ersetzt AWS Signature Version 2 zum Signieren von Amazon S3-API-Anforderungen in den AWS-Foren an.
- Wenden Sie sich bei weiteren Fragen oder Bedenken an den AWS Support.

Von Signature Version# 2 zur Signature Version 4 wechseln.

Wenn Sie aktuell Signatur Version 2 für die Authentifizierung der API-Anforderungen in Amazon S3 verwenden, sollten Sie möglichst bald auf Signature Version 4 umwechseln. Das Ende des Supports für Signature Version 2 wird unter AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet) (p. 804) beschrieben.

Weitere Informationen zur Verwendung von Signature Version 4 mit der Amazon S3 REST-API finden Sie unter Authentifizieren von Anforderungen (AWS Signature Version 4) in der Amazon Simple Storage Service API Reference.

Die folgende Liste stellt die AWS SDKs mit der erforderlichen Mindestversion zur Verwendung von Signature Version 4 (SigV4) dar.
Wenn Sie vorab signierte URLs mit den SDKs für AWS Java, JavaScript (Node.js) oder Python (Boto/CLI) verwenden, müssen Sie in der Client-Konfiguration die richtige AWS Region sowie Signature Version 4 festlegen. Weitere Informationen zum Einrichten von `SigV4` in der Client-Konfiguration finden Sie unter Angabe der Signature-Version in der Anforderungsauthentifizierung (p. 802).

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS SDK for Java v1	Upgrade auf Java 1.11.201+ oder v2 im 4. Quartal 2018.	Ja	Angabe der Signature-Version in der Anforderungsauthentifizierung (p. 802)
AWS SDK for Java v2 (Vorversion)	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK for Java

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS SDK für .NET v1	Auf 3.1.10 oder höher aktualisieren.	Ja	AWS SDK für .NET
AWS SDK für .NET v2	Auf 3.1.10 oder höher aktualisieren.	Nein	AWS SDK für .NET v2
AWS SDK für .NET v3	Auf 3.3.0.0 oder höher aktualisieren.	Ja	AWS SDK für .NET v3
AWS SDK for JavaScript v1	Auf 2.68.0 oder höher aktualisieren.	Ja	AWS SDK for JavaScript
AWS SDK for JavaScript v2	Auf 2.68.0 oder höher aktualisieren.	Ja	AWS SDK for JavaScript
AWS SDK for JavaScript v3	Derzeit ist keine weitere Maßnahme erforderlich. Im Q3 2019 auf Hauptversion V3 aktualisieren.	Nein	AWS SDK for JavaScript
AWS SDK für PHP v1	Empfohlen wird ein Upgrade auf die aktuelle Version von PHP oder mindestens v2.7.4 und der Wert v4 für den Signature-Parameter in der Konfiguration des S3-Clients.	Ja	AWS SDK für PHP

Wenn Sie diese SDK/ dieses Produkt verwenden	Aktualisieren Sie auf diese SDK-Version	Bei Verwendung von Sigv4 Codeänderung am Client erforderlich?	Link zur SDK-Dokumentation
AWS SDK für PHP v2	Empfohlen wird ein Upgrade auf die aktuelle Version von PHP oder mindestens v2.7.4 und der Wert v4 für den Signature-Parameter in der Konfiguration des S3-Clients.	Nein	AWS SDK für PHP
AWS SDK für PHP v3	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK für PHP
Boto2	Auf Boto2 v2.49.0. aktualisieren.	Ja	Boto 2-Aktualisierung
Boto3	Auf 1.5.71 (Botocore), 1.4.6 (Boto3) aktualisieren.	Ja	Boto 3 – AWS SDK für Python
AWS CLI	Auf 1.11.108 aktualisieren.	Ja	AWS-Befehlszeilenschnittstelle
AWS CLI v2 (Vorversion)	Keine SDK-Aktualisierung erforderlich.	Nein	AWS-Befehlszeilenschnittstelle, Version 2
AWS SDK für Ruby v1	Auf Ruby V3 aktualisieren.	Ja	Ruby V3 für AWS
AWS SDK für Ruby v2	Auf Ruby V3 aktualisieren.	Ja	Ruby V3 für AWS
AWS SDK für Ruby v3	Keine SDK-Aktualisierung erforderlich.	Nein	Ruby V3 für AWS
Go	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK for Go
C++	Keine SDK-Aktualisierung erforderlich.	Nein	AWS SDK für C++

AWS-Tools für Windows PowerShell oder AWS-Tools für PowerShell Core

Wenn Sie Modulversionen verwenden, die älter als 3.3.0.0 sind, müssen Sie auf 3.3.0.0 aktualisieren.

Verwenden Sie `Get-Module` cmdlet, um die Versionsinformationen zu erhalten:

```
Get-Module -Name AWSPowershell
Get-Module -Name AWSPowershell.NetCore
```

Um auf die Version 3.3.0.0 zu aktualisieren, verwenden Sie das cmdlet `Update-Module`:

```
Update-Module -Name AWSPowershell
Update-Module -Name AWSPowershell.NetCore
```

Für Signature Version 2-Datenverkehr können Sie vorab signierte URLs verwenden, die länger als sieben Tage gültig sind.

Einrichten der AWS-CLI

Befolgen Sie die Schritte zum Herunterladen und Konfigurieren der AWS CLI (AWS Command Line Interface).

> **Note**
>
> Services in AWS, wie z. B. Amazon S3, fordern beim Zugriff die Eingabe von Anmeldeinformationen. Der Service kann dann feststellen, ob Sie über die Berechtigung für den Zugriff auf seine Ressourcen verfügen. Für die Konsole müssen Sie Ihr Passwort eingeben. Sie können für Ihr AWS-Konto Zugriffsschlüssel erstellen, um auf die AWS CLI oder die API zuzugreifen. Wir raten Ihnen jedoch davon ab, mittels der Anmeldeinformationen für Ihr AWS-Konto auf AWS zuzugreifen. Stattdessen empfehlen wir, AWS Identity and Access Management (IAM) zu verwenden. Erstellen Sie einen IAM-Benutzer und fügen Sie den Benutzer zu einer IAM-Gruppe mit Administrator-Berechtigungen hinzu. Anschließend gewähren Sie dem von Ihnen erstellten IAM-Benutzer administrative Berechtigungen. Danach können Sie mithilfe einer speziellen URL und der Anmeldeinformationen des IAM-Benutzers auf AWS zugreifen. Anweisungen finden Sie unter Erstellen Ihres ersten Administratorbenutzers und Ihrer ersten Administratorgruppe in IAM im IAM-Benutzerhandbuch.

Einrichten von AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im Benutzerhandbuch für die AWS-Befehlszeilenschnittstelle:

   - Einrichten mit der AWS-Befehlszeilenschnittstelle
   - Configuring the AWS Command Line Interface

2. Fügen Sie ein benanntes Profil für den Administratorbenutzer in der AWS CLI-Konfigurationsdatei hinzu. Verwenden Sie dieses Profil beim Ausführen von AWS CLI-Befehlen.

```
[adminuser]
aws_access_key_id = adminuser access key ID
aws_secret_access_key = adminuser secret access key
```

```
region = aws-region
```

Eine Liste der verfügbaren AWS-Regionen finden Sie unter Regionen und Endpunkte im AWS General Reference.

3. Überprüfen Sie die Einrichtung, indem Sie die folgenden Befehle in die Befehlszeile eingeben.

   - Führen Sie den `help`-Befehl aus, um zu überprüfen, ob die AWS CLI auf Ihrem Computer installiert ist:

     ```
     aws help
     ```

   - Führen Sie einen `S3`-Befehl aus, um zu überprüfen, ob der Benutzer Amazon S3 erreichen kann. Dieser Befehl listet Buckets in Ihrem Konto auf. Die AWS CLI verwendet `adminuser`-Anmeldeinformationen, um die Anforderung zu authentifizieren.

     ```
     aws s3 ls --profile adminuser
     ```

Verwendung von AWS SDK for Java

AWS SDK for Java stellt eine API für den Amazon S3-Bucket und Objektoperationen bereit. Für Objektoperationen stellt das SDK – zusätzlich zu der API für das Hochladen von Objekten in einer einzigen Operation – eine API zum mehrteiligen Hochladen großer Objekte bereit. Weitere Informationen finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).

Themen

Das AWS SDK for Java bietet Ihnen die Option, eine High-Level- oder eine Low-Level-API zu verwenden.

API auf niedriger Ebene

Die Low-Level-APIs entsprechen den zugrunde liegenden Amazon S3 REST-Operationen, wie beispielsweise zum Erstellen, Aktualisieren und Löschen, die auf Buckets und Objekte angewendet werden. Wenn Sie große Objekte über die Low-Level-API für mehrteilige Uploads hochladen, bietet dies eine bessere Kontrolle. Beispielsweise können Sie damit mehrteilige Uploads unterbrechen und fortsetzen, die Teilegrößen während des Uploads ändern oder Uploads beginnen, obwohl Sie die Größe der Daten nicht vorab kennen. Wenn Sie diese Anforderungen nicht haben, verwenden Sie die High-Level API zum Hochladen von Objekten.

API auf hoher Ebene

Zum Hochladen von Objekten bietet das SDK einen höheren Abstraktionsgrad, indem es die Klasse `TransferManager` bereitstellt. Die High-Level API ist eine einfachere API, wobei Sie mit nur ein paar Codezeilen Dateien und Streams in Amazon S3 hochladen können. Sie sollten diese API verwenden, um Daten hochzuladen, es sei denn, Sie müssen den Upload kontrollieren, wie im vorigen Abschnitt über die Low-Level API beschrieben.

Für kleinere Datengrößen lädt die `TransferManager`-API Daten in einer einzigen Operation hoch. Der `TransferManager` wechselt jedoch zur Verwendung der API für den mehrteiligen Upload, wenn die Datengröße eine bestimmte Schwelle erreicht. Wenn möglich, verwendet der `TransferManager` mehrere Threads, um die Teile nebenläufig hochzuladen. Wenn der Upload eines Teils fehlschlägt, wiederholt API diesen Upload bis zu dreimal. Dies sind jedoch Optionen, die mit der `TransferManagerConfiguration`-Klasse konfiguriert werden können.

Note

Wenn Sie einen Stream für die Datenquelle verwenden, führt die `TransferManager`-Klasse keine gleichzeitigen Uploads durch.

Die Java API-Organisation

Die folgenden Pakete in AWS SDK for Java stellen die API bereit:

- com.amazonaws.services.s3 Stellt die APIs für das Erstellen von Amazon S3-Clients und für das Arbeiten mit Buckets und Objekten bereit. Dieses Paket ermöglicht Ihnen beispielsweise, Buckets zu erstellen, Objekte hochzuladen, Objekte abzurufen, Objekte zu löschen und Schlüssel aufzulisten.
- com.amazonaws.services.s3.transfer Stellt die High-Level-API-Datenoperationen bereit.

  Diese High-Level-API ist darauf ausgelegt, das Übertragen von Objekten in und aus Amazon S3 zu vereinfachen. Dieses Paket beinhaltet die `TransferManager`-Klasse, die asynchrone Methoden für das Arbeiten mit sowie Abfragen und Bearbeiten von Übertragungen bereitstellt. Außerdem umfasst sie die `TransferManagerConfiguration`-Klasse, mit der Sie die Mindestteilgröße für den mehrteiligen Upload konfigurieren können, sowie den Schwellenwert (in Bytes), ab wann mehrteilige Uploads verwendet werden sollen.
- com.amazonaws.services.s3.model – Stellt die Low-Level API-Klassen bereit, um Anforderungen zu erstellen und Antworten zu verarbeiten. Es beinhaltet beispielsweise die Klasse `GetObjectRequest`, um Ihre "Get object"-Anforderung zu beschreiben, die Klasse `ListObjectsRequest`, um Ihre Listenschlüsselanforderungen zu beschreiben, und die Klasse `InitiateMultipartUploadRequest`, um mehrteilige Uploads zu erstellen.

Weitere Informationen zur AWS SDK for Java-API finden Sie unter AWS SDK for Java API Reference.

Testen der Amazon S3-Java-Codebeispiele

Die Java-Beispiele in diesem Handbuch sind sämtlich mit der AWS SDK for Java-Version 1.11.321 kompatibel. Für Anleitungen zum Einrichten und Ausführen von Codebeispielen vgl. Erste Schritte mit AWS SDK for Java im AWS SDK for Java-Entwicklerhandbuch.

Verwendung von AWS SDK für .NET

AWS SDK für .NET stellt die API für den Amazon S3-Bucket und Objektoperationen bereit. Für Objektoperationen bietet der SDK zusätzlich zu der API für das Hochladen von Objekten in einer einzigen Operation die API zum mehrteiligen Hochladen großer Objekte bereit (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)).

Themen

Das AWS SDK für .NET bietet Ihnen die Option, eine High-Level- oder eine Low-Level-API zu verwenden.

API auf niedriger Ebene

Die Low-Level-APIs entsprechen den zugrunde liegenden Amazon S3 REST-Operationen, wie beispielsweise zum Erstellen, Aktualisieren und Löschen, die auf Buckets und Objekte angewendet werden. Wenn Sie große Objekte über die Low-Level-API für mehrteilige Uploads hochladen (siehe Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198)), bietet dies eine bessere Kontrolle. Beispielsweise können Sie damit mehrteilige Uploads unterbrechen und fortsetzen,

die Teilegrößen während des Uploads ändern oder Uploads beginnen, obwohl Sie die Größe der Daten nicht vorab kennen. Wenn Sie diese Anforderungen nicht haben, verwenden Sie die High-Level API zum Hochladen von Objekten.

API auf hoher Ebene

Zum Hochladen von Objekten bietet das SDK einen höheren Abstraktionsgrad, indem es die Klasse `TransferUtility` bereitstellt. Die High-Level API ist eine einfachere API, wobei Sie mit nur ein paar Codezeilen Dateien und Streams in Amazon S3 hochladen können. Sie sollten diese API verwenden, um Daten hochzuladen, es sei denn, Sie müssen den Upload kontrollieren, wie im vorigen Abschnitt über die Low-Level API beschrieben.

Für kleinere Datengrößen lädt die `TransferUtility`-API Daten in einer einzigen Operation hoch. Der `TransferUtility` wechselt jedoch zur Verwendung der API für den mehrteiligen Upload, wenn die Datengröße eine bestimmte Schwelle erreicht. Standardmäßig verwendet er mehrere Threads, um die Teile nebenläufig hochzuladen. Wenn der Upload eines Teils fehlschlägt, wiederholt API diesen Upload bis zu dreimal. Dies sind jedoch konfigurierbare Optionen.

> **Note**
>
> Wenn Sie einen Stream für die Datenquelle verwenden, führt die `TransferUtility`-Klasse keine gleichzeitigen Uploads durch.

Die .NET API Organisation

Wenn Sie Amazon S3-Anwendungen mit AWS SDK für .NET schreiben, verwenden Sie `AWSSDK.dll`. Die folgenden Namespaces in dieser Gruppe stellen die API für mehrteilige Uploads bereit:

- Amazon.S3.Transfer – Stellt die High-Level API für den mehrteiligen Upload Ihrer Daten bereit.

  Beinhaltet die `TransferUtility`-Klasse, die Ihnen ermöglicht, eine Datei, ein Verzeichnis oder einen Stream zum Hochladen Ihrer Daten bereitzustellen. Darüber hinaus beinhaltet sie die Klassen `TransferUtilityUploadRequest` und `TransferUtilityUploadDirectoryRequest`, um erweiterte Einstellungen zu konfigurieren, wie beispielsweise die Anzahl gleichzeitiger Threads, die Teilegröße, die Objekt-Metadaten, die Speicherklasse (STANDARD, REDUCED_REDUNDANCY) und die Objekt-Zugriffskontrollliste (ACL).
- Amazon.S3 – Stellt die Implementierung für Low-Level APIs bereit.

  Bietet Methoden, die der Amazon S3 REST-API für den mehrteiligen Upload entsprechen (vgl. Verwenden der REST API für einen mehrteiligen Upload (p. 228)).
- Amazon.S3.Model – Stellt die Low-Level API-Klassen bereit, um Anforderungen zu erstellen und Antworten zu verarbeiten. Sie beinhaltet beispielsweise die Klassen `InitiateMultipartUploadRequest` und `InitiateMultipartUploadResponse`, die Sie für die Initiierung eines mehrteiligen Uploads verwenden können, und die Klassen `UploadPartRequest` und `UploadPartResponse`, wenn Teile hochgeladen werden.
- Amazon.S3.Encryption – Stellt `AmazonS3EncryptionClient` bereit.
- Amazon.S3.Util – Stellt verschiedene Klassen wie z. B. `AmazonS3Util` und `BucketRegionDetector` bereit.

Weitere Informationen zur AWS SDK für .NET-API finden Sie unter AWS SDK for .NET Version 3 API-Referenz.

Ausführen der Amazon S3 .NET-Codebeispiele

Die .NET-Codebeispiele in diesem Handbuch sind sämtlich mit der AWS SDK für .NET-Version 3.0 kompatibel. Für Informationen zum Einrichten und Ausführen der Codebeispiele vgl. Erste Schritte mit dem AWS SDK für .NET im AWS SDK for .NET-Entwicklerhandbuch.

Verwenden von AWS SDK für PHP und Ausführen von PHP-Beispielen

AWS SDK für PHP stellt die API für den Zugriff auf die API für Amazon S3-Bucket und Objektoperationen bereit. Der SDK bietet Ihnen die Option, die High-Level-API oder höhere Abstraktionen des Service zu verwenden.

Das SDK steht unter AWS SDK für PHP zur Verfügung und enthält auch Anweisungen für die Installation und die ersten Schritte mit dem SDK.

Die Einrichtung für die Verwendung von AWS SDK für PHP ist von Ihrer Umgebung abhängig, und wie Sie Ihre Anwendung ausführen wollen. Informationen zum Einrichten Ihrer Umgebung für die Ausführung der Beispiele in dieser Dokumentation finden Sie unter AWS SDK für PHP Erste Schritte.

Themen

AWS SDK für PHP-Ebenen

Das AWS SDK für PHP bietet Ihnen die Option, eine High-Level- oder eine Low-Level-API zu verwenden.

API auf niedriger Ebene

Die Low-Level-APIs entsprechen den zugrunde liegenden Amazon S3 REST-Operationen, wie beispielsweise zum Erstellen, Aktualisieren und Löschen, die auf Buckets und Objekte angewendet werden. Die Low-Level-APIs bieten mehr Kontrolle über diese Operationen. Beispielsweise können Sie Ihre Anforderungen zusammenfassen und parallel ausführen. Bei Verwendung der API für mehrteilige Uploads können Sie die Objektteile unabhängig voneinander verwalten. Beachten Sie, dass diese Low-Level-API-Aufrufe ein Ergebnis zurückgeben, das alle Amazon S3-Antwortdetails beinhaltet. Weitere Informationen zur API für mehrteilige Uploads finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).

High-Level-Abstraktionen

Die High-Level-Abstraktionen sind darauf ausgelegt, häufige Anwendungsfälle zu vereinfachen. Um beispielsweise große Objekte unter Verwendung der Low-Level-API hochzuladen, müssen Sie zuerst `Aws\S3\S3Client::createMultipartUpload()` aufrufen, dann die Methode `Aws\S3\S3Client::uploadPart()`, um die Objektteile hochzuladen, und dann die Methode `Aws\S3\S3Client::completeMultipartUpload()`, um den Upload abzuschließen. Sie können statt dessen das `Aws\S3\\MultipartUploader`-Objekt verwenden, das sich auf einer höheren Ebene befindet, und das die Erstellung mehrteiliger Uploads vereinfacht.

Als weiteres Beispiel für die Auflistung von Objekten in einem Bucket können Sie die Iterator-Funktion von AWS SDK für PHP verwenden, um alle Objektschlüssel zurückzugeben, unabhängig davon, wie viele Objekte Sie in dem Bucket gespeichert haben. Wenn Sie die Low-Level-API verwenden, gibt die Antwort maximal 1000 Schlüssel zurück. Wenn ein Bucket mehr als 1000 Objekte enthält, wird das Ergebnis abgeschnitten. Sie müssen die Antwort verwalten und auf Kürzungen überprüfen.

PHP-Beispiele ausführen

Informationen zum Einrichten und Verwenden der Amazon S3-Beispiele für Version 3 von AWS SDK for PHP finden Sie unter Installation im AWS SDK für PHP Developer Guide.

Verwandte Ressourcen

- AWS SDK für PHP für Amazon S3
- AWS SDK für PHP – Dokumentation
- AWS SDK for PHP-API für Amazon S3

Verwenden von AWS SDK für Ruby – Version 3

AWS SDK für Ruby stellt eine API für Amazon S3-Bucket- und Objektoperationen bereit. Für Objektoperationen können Sie die API für das Hochladen von Objekten in einer einzigen Operation oder die API zum mehrteiligen Hochladen großer Objekte verwenden (siehe Verwenden von AWS SDK für Ruby für einen mehrteiligen Upload (p. 227)). Die API für den Upload in einer Operation kann jedoch keine großen Objekte verarbeiten, und hinter den Kulissen wird der mehrteilige Upload für Sie verwaltet. Damit wird die Anzahl der Skripts reduziert, die Sie schreiben müssen.

Die Ruby API-Organisation

Wenn Sie Amazon S3-Anwendungen mit AWS SDK für Ruby erstellen, müssen Sie das SDK für Ruby-Gem installieren. Weitere Informationen finden Sie unter AWS SDK for Ruby - Version 3. Nachdem Sie es installiert haben, können Sie auf die API zugreifen, einschließlich der folgenden Schlüsselklassen:

- Aws::S3::Resource – Stellt die Schnittstelle zu Amazon S3 für das Ruby SDK dar und bietet Methoden zum Erstellen und Auflisten von Buckets.

  Die Klasse`S3` stellt die Instance-Methode `#buckets` für den Zugriff auf vorhandene Buckets oder zum Erstellen neuer Buckets bereit.
- Aws::S3::Bucket – Repräsentiert einen Amazon S3-Bucket.

  Die Klasse `Bucket` stellt die Methoden `#object(key)` und `#objects` für den Zugriff auf die Objekte in einem Bucket bereit, ebenso wie die Methoden zum Löschen eines Buckets und zur Rückgabe von Informationen über einen Bucket, wie beispielsweise die Bucket-Richtlinie.
- Aws::S3::Object – Stellt ein durch seinen Schlüssel identifiziertes Amazon S3-Objekt dar.

  Die `Object`-Klasse bietet Methoden für den Abruf und die Einrichtung der Eigenschaften eines Objekts, mit Angabe der Speicherklasse zum Speichern von Objekten, und der Einrichtung von Objektberechtigungen unter Verwendung von Zugriffskontrolllisten. Die Klasse `Object` enthält auch Methoden zum Löschen, Hochladen und Kopieren von Objekten. Beim mehrteiligen Hochladen von Objekten stellt Ihnen diese Klasse Optionen bereit, um die Reihenfolge der hochgeladenen Teile sowie die Teilegröße anzugeben.

Weitere Informationen zum AWS SDK for Ruby API finden Sie unter AWS SDK for Ruby - Version 2.

Testen der Ruby Script-Codebeispiele

Am einfachsten gelingt der Einstieg in die Ruby Script--Codebeispiele, indem das neuste AWS SDK für Ruby Gem installiert wird. Weitere Informationen zur Installation und Aktualisierung der neuesten Gem finden Sie unter AWS SDK for Ruby - Version 3. Die folgenden Aufgaben führen Sie durch das Erstellen und Testen der Ruby Script-Beispiele, wobei vorausgesetzt wird, dass Sie AWS SDK für Ruby installiert haben.

Allgemeiner Vorgang beim Erstellen und Testen von Ruby Script-Beispielen

1	Für den Zugriff auf AWS müssen Sie die Anmeldeinformationen für Ihre SDK für Ruby-Anwendung bereitstellen. Weitere Informationen finden Sie unter Konfigurieren des AWS-SDK für Ruby.
2	Erstellen Sie ein neues SDK für Ruby-Skript und fügen Sie oben die folgenden Codezeilen ein. `#!/usr/bin/env ruby` `require 'rubygems'` `require 'aws-sdk-s3'` Die erste Zeile ist die Interpreter-Anweisung, und die beiden `require`-Anweisungen importieren zwei erforderliche Gems in Ihr Skript.
3	Kopieren Sie das Code aus dem Abschnitt, den Sie gerade lesen, in Ihr Skript.
4	Aktualisieren Sie den Code mit den erforderlichen Daten. Wenn Sie beispielsweise eine Datei hochladen, geben Sie den Dateipfad und den Bucket-Namen an.
5	Führen Sie das Skript aus. Überprüfen Sie Änderungen an Buckets und Objekten unter Verwendung der AWS Management Console. Weitere Informationen zur AWS Management Console finden Sie unter https://aws.amazon.com/Konsole/.

Ruby-Beispiele

Die folgenden Links enthalten Beispiele, die Ihnen bei den ersten Schritten mit SDK für Ruby - Version 3 helfen:

- Verwenden des AWS SDK für Ruby Version 3 (p. 68)
- Hochladen eines Objekts mit AWS SDK für Ruby (p. 197)

Verwenden von AWS SDK for Python (Boto)

Boto ist ein Python-Paket, das Schnittstellen für AWS bereitstellt, einschließlich Amazon S3. Weitere Informationen zu Boto finden Sie unter AWS SDK for Python (Boto). Unter dem Link für die ersten Schritte auf dieser Seite finden Sie Schritt-für-Schritt-Anleitungen.

Verwenden der AWS Mobile SDKs für iOS und Android

Mit den AWS Mobile SDKs für Android und iOS können Sie robuste Cloud-Backends schnell und einfach in Ihre vorhandenen mobilen Apps integrieren. Sie können Funktionen wie Benutzeranmeldung, Datenbanken, Push-Benachrichtigungen und vieles mehr konfigurieren und nutzen, ohne ein AWS-Experte zu sein.

Die AWS Mobile SDKs bieten einfachen Zugriff auf Amazon S3 und viele andere AWS-Services. Erste Schritte für die Nutzung von AWS Mobile- SDKs finden Sie unter Erste Schritte mit den AWS Mobile SDKs.

Weitere Infos

Verwenden der AWS Amplify JavaScript-Bibliothek (p. 815)

Verwenden der AWS Amplify JavaScript-Bibliothek

AWS Amplify ist eine Open-Source-JavaScript-Bibliothek für Web-Entwickler und Entwickler mobiler Apps, die Cloud-fähige Anwendungen erstellen. AWS Amplify bietet anpassbare UI-Komponenten und eine deklarative Schnittstelle für die Verwendung mit einem S3-Bucket, zusammen mit anderen High-Level-Kategorien für AWS-Services.

Zur Nutzung der AWS Amplify JavaScript-Bibliothek wählen Sie einen der folgenden Links aus:

- Erste Schritte mit der AWS Amplify-Bibliothek für Web
- Erste Schritte mit der AWS Amplify-Bibliothek für React Native

Weitere Informationen zu AWS Amplify finden Sie unter AWS Amplify auf GitHub.

Weitere Infos

Verwenden der AWS Mobile SDKs für iOS und Android (p. 814)

Anhänge

Dieser Anhang zu Entwicklerhandbuch für Amazon Simple Storage Service umfasst folgende Abschnitte.

Themen

Anhang A: Verwenden der SOAP-API

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Dieser Abschnitt enthält spezifische Informationen für die Amazon S3 SOAP API.

> Note
>
> SOAP-Anforderungen, sowohl authentifizierte als auch anonyme, müssen unter Verwendung von SSL an Amazon S3 gesendet werden. Amazon S3 gibt einen Fehler zurück, wenn Sie eine SOAP-Anforderung über HTTP versenden.

Allgemeine SOAP-API-Elemente

> Note
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Sie können mit Amazon S3 unter Verwendung von SOAP 1.1 über HTTP interagieren. Das Amazon S3-WSDL, das die Amazon S3-API auf maschinenlesbare Weise beschreibt, ist verfügbar unter: https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl. Das Amazon S3-Schema ist verfügbar unter https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.xsd.

Die meisten Benutzer interagieren mit Amazon S3 über ein SOAP-Toolkit, das auf ihre Sprache und Entwicklungsumgebung zugeschnittenen ist. Verschiedene Toolkits machen das Amazon S3-API auf unterschiedliche Weise verfügbar. Bitte konsultieren Sie Ihre spezifische Toolkit-Dokumentation für seine Verwendung. Dieser Abschnitt veranschaulicht die Amazon S3-SOAP-Operationen in einer Toolkit-unabhängigen Weise durch Darstellen der XML-Anfragen und Antworten, wie sie "online" erscheinen.

Allgemeine Elemente

Sie können die folgenden autorisierungsbezogenen Elemente in jede SOAP-Anfrage aufnehmen:

- `AWSAccessKeyId:` Die AWS-Zugriffsschlüssel-ID des Anforderers

- `Timestamp:` Die aktuelle Systemzeit
- `Signature:` Die Signatur für die Anfrage

Authentifizieren von SOAP-Anforderungen

Note

Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Jede nicht anonyme Anfrage muss Authentifizierungsinformationen enthalten, um die Identität des Prinzipals einzurichten, der die SOAP-Anfrage stellt. In SOAP wird die Authentifizierungsinformation in den folgenden Elemente der SOAP-Anfrage abgelegt:

- Ihre AWS-Zugriffsschlüssel-ID

  Note

  Bei authentifizierten SOAP-Anfragen werden keine temporären Sicherheitsanmeldeinformationen unterstützt. Weitere Informationen den verschiedenen Anmeldeinformationen finden Sie unter Senden von Anforderungen (p. 11).
- `Timestamp:` Dabei muss es sich um ein dateTime (siehe http://www.w3.org/TR/xmlschema-2/#dateTime) in der Zeitzone Coordinated Universal Time (Greenwich Mean Time) handeln, beispielsweise `2009-01-01T12:00:00.000Z`. Die Autorisierung schlägt fehl, wenn dieser Zeitstempel um mehr als 15 Minuten von der Uhr auf Amazon S3-Servern abweicht.
- `Signature:` Der RFC 2104 HMAC-SHA1-Digest (siehe http://www.ietf.org/rfc/rfc2104.txt) der Verkettung von „AmazonS3“ + OPERATION + Zeitstempel unter Verwendung Ihres geheimen AWS-Zugriffsschlüssels als Schlüssel. In der folgenden CreateBucket-Beispielanfrage würde das Signatur-Element den HMAC-SHA1-Digest mit dem Wert "AmazonS3CreateBucket2009-01-01T12:00:00.000Z" enthalten:

In der folgenden CreateBucket-Beispielanfrage würde das Signatur-Element den HMAC-SHA1-Digest mit dem Wert "AmazonS3CreateBucket2009-01-01T12:00:00.000Z" enthalten:

Example

```
<CreateBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Acl>private</Acl>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-01-01T12:00:00.000Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</CreateBucket>
```

Note

SOAP-Anforderungen, sowohl authentifizierte als auch anonyme, müssen unter Verwendung von SSL an Amazon S3 gesendet werden. Amazon S3 gibt einen Fehler zurück, wenn Sie eine SOAP-Anforderung über HTTP versenden.

Important

Aufgrund von unterschiedlichen Interpretationen dahingehend, wie eine zusätzliche Zeitgenauigkeit wegfallen kann, sollten .NET-Benutzer darauf achten, Amazon S3 keine

übermäßig spezifischen Zeitstempel zu schicken. Dies kann bewerkstelligt werden, indem manuell `DateTime`-Objekte mit einer Millisekunde Genauigkeit erstellt werden.

Festlegen der Zugriffsrichtlinie mit SOAP

Note

Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Die Zugriffskontrolle kann zu dem Zeitpunkt eingerichtet werden, zu dem ein Bucket oder ein Objekt geschrieben werden, einschließlich des „AccessControlList“-Elements, indem `CreateBucket`, `PutObjectInline` oder `PutObject` aufgerufen werden. Das AccessControlList-Element ist in Identity and Access Management in Amazon S3 (p. 329) beschrieben. Wenn für diese Operationen keine Zugriffskontrollliste angegeben ist, wird die Ressource mit einer Standard-Zugriffsrichtlinie erstellt, die dem Auftraggeber FULL_CONTROL-Zugriff erteilt (das erfolgt auch dann, wenn es sich um eine PutObjectInline- oder PutObject-Anfrage für ein bereits vorhandenes Objekt handelt).

Nachfolgend sehen Sie eine Anfrage, die Daten in ein Objekt schreibt, das Objekt lesbar für anonyme Prinzipale macht, und dem angegebenen Benutzer FULL_CONTROL-Rechte für den Bucket erteilt (die meisten Entwickler wollen sich selbst FULL_CONTROL-Zugriff auf ihren eigenen Bucket geben).

Example

Die nachfolgende Anfrage schreibt Daten in ein Objekt und macht das Objekt lesbar für anonyme Prinzipale.

`Sample Request`

```
<PutObjectInline xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <Metadata>
    <Name>Content-Type</Name>
    <Value>text/plain</Value>
  </Metadata>
  <Data>aGEtaGE=</Data>
  <ContentLength>5</ContentLength>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers<URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</PutObjectInline>
```

`Sample Response`

```
<PutObjectInlineResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <PutObjectInlineResponse>
    <ETag>&quot828ef3fdfa96f00ad9f27c383fc9ac7f&quot</ETag>
    <LastModified>2009-01-01T12:00:00.000Z</LastModified>
  </PutObjectInlineResponse>
</PutObjectInlineResponse>
```

Die Zugriffskontrollrichtlinie kann für einen vorhandenen Bucket oder ein vorhandenes Objekt mit den Methoden `GetBucketAccessControlPolicy`, `GetObjectAccessControlPolicy`, `SetBucketAccessControlPolicy` und `SetObjectAccessControlPolicy` gelesen oder eingerichtet werden. Weitere Informationen finden Sie in der detaillierten Erklärung dieser Methoden.

Anhang B: Authentifizieren von Anforderungen (AWS-Signatur Version 2)

Important

In diesem Abschnitt wird das Authentifizieren von Anforderungen unter Verwendung von AWS Signature Version 2 beschrieben. Signature Version 2 wird deaktiviert (veraltet) Amazon S3 akzeptiert nur API-Anforderungen, die mit Signature Version 4 signiert wurden. Weitere Informationen finden Sie unter AWS Signature Version 2 für Amazon S3 deaktiviert (veraltet) (p. 804)
Signatur-Version 4 wird in allen AWS-Regionen unterstützt und ist die einzige Version, die in neuen Regionen unterstützt wird. Weitere Informationen finden Sie unter Authentifizieren von Anforderungen (AWS Signature Version 4) im Amazon Simple Storage Service API Reference. Amazon S3 ermöglicht das Ermitteln der zum Signieren einer Anforderung verwendet API-Signaturversion. Um Beeinträchtigungen der betrieblichen Prozesse zu vermeiden, muss festgestellt werden, ob Workflows mit Signature Version 2 signieren. Diese Workflows müssen dann auf die Verwendung von Signature Version 4 umgestellt werden.

- Wenn Sie CloudTrail-Ereignisprotokolle verwenden (empfohlen), finden Sie unter Verwenden von AWS CloudTrail zum Identifizieren von Amazon S3 Signature Version 2-Anforderungen (p. 761) Informationen zum Abfragen und Identifizieren solcher Anforderungen.
- Wenn Sie die Amazon S3 Server Access-Protokolle verwenden, finden Sie weitere Informationen unter Verwenden von Amazon S3-Zugriffsprotokollen zum Identifizieren von Signature Version 2-Anforderungen (p. 799).

Themen

Authentifizieren von Anforderungen mit der REST-API

Beim Zugriff auf Amazon S3 mit REST müssen Sie in Ihrer Anfrage die folgenden Elemente angeben, damit diese authentifiziert werden kann.:

Anfordern von Elementen

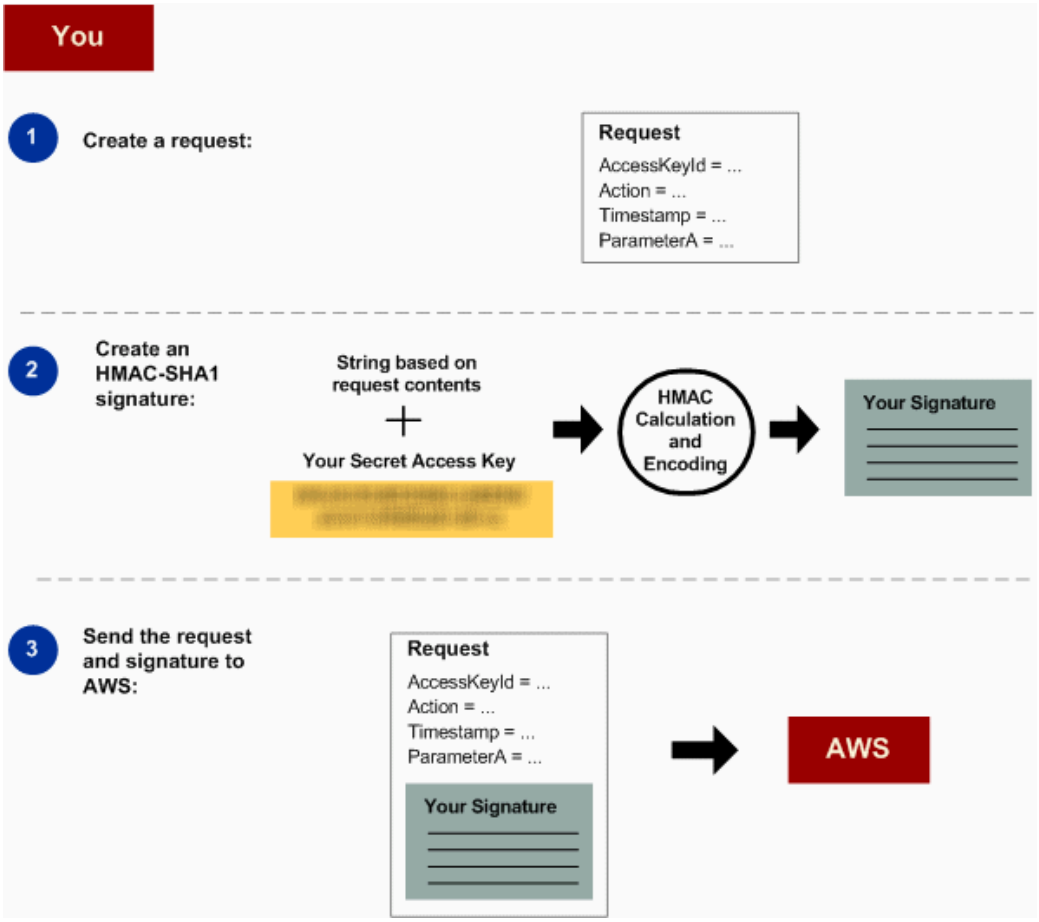
- AWS-Zugriffsschlüssel-ID – Jede Anforderung muss die Zugriffsschlüssel-ID der Identität enthalten, mit der Sie Ihre Anforderung senden.

- Signatur – Jede Anforderung muss eine gültige Anforderungssignatur enthalten. Andernfalls wird die Anforderung abgelehnt.

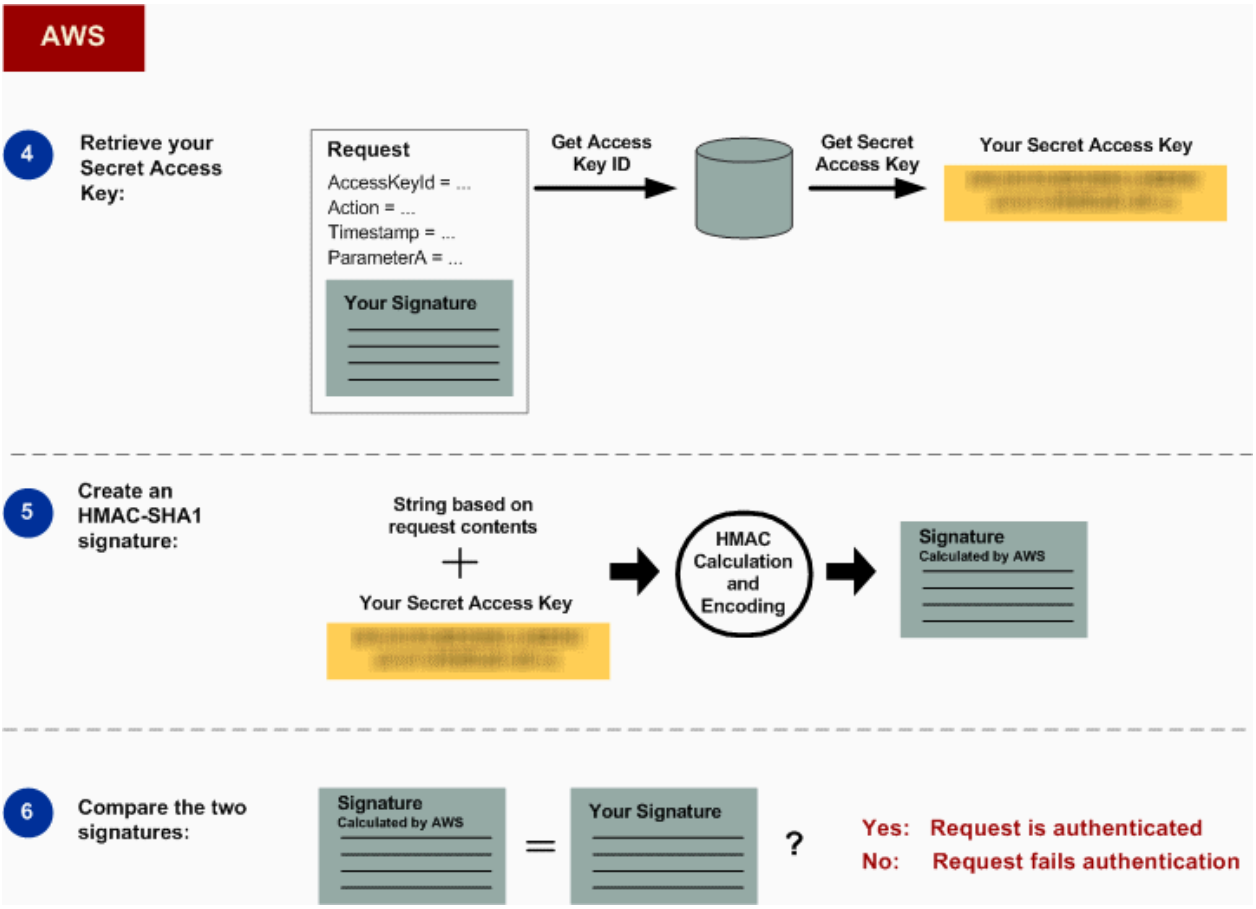
  Eine Anfragesignatur wird unter Verwendung Ihres geheimen Zugriffsschlüssels berechnet, das ist ein gemeinsames Geheimnis, das nur Sie und AWS kennen.
- Zeitstempel – Jede Anforderung muss Datum und Uhrzeit des Zeitpunktes enthalten, zu dem die Anforderung erstellt wurde, dargestellt als Zeichenfolge in UTC.
- Datum – Jede Anforderung muss den Zeitstempel der Anforderung enthalten.

  Abhängig von der verwendeten API-Aktion können Sie ein Ablaufdatum und eine Ablaufzeit für die Anfrage angeben, statt dem Zeitstempel oder zusätzlich zu diesem. Um festzustellen, was notwendig ist, lesen Sie im Authentifizierungsthema für die jeweiligen Aktion nach.

Nachfolgend finden Sie die allgemeinen Schritte für die Authentifizierung von Anfragen für Amazon S3. Es wird vorausgesetzt, dass Sie über die erforderlichen Sicherheitsanmeldeinformationen, die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel verfügen.



1	Erstellen einer Anfrage an AWS.
2	Berechnung der Signatur unter Verwendung Ihres geheimen Zugriffsschlüssels.
3	Sendet der Anfrage an Amazon S3. Integrieren Sie Ihre Zugriffsschlüssel-ID und die Signatur in die Anforderung. Amazon S3 führt die nächsten drei Schritte durch.



4	Amazon S3 verwendet die Zugriffsschlüssel-ID, um Ihren geheimen Zugriffschlüssel abzurufen.
5	Amazon S3 berechnet eine Signatur aus den Anforderungsdaten und dem geheimen Zugriffsschlüssel mit demselben Algorithmus, mit dem die in der Anforderung gesendete Signatur berechnet wurde.
6	Wenn die von Amazon S3 generierte Signatur mit der in der Anforderung gesendeten Signatur übereinstimmt, wird die Anforderung als authentisch eingestuft. Falls der Vergleich fehlschlägt, wird die Anforderung verworfen, und Amazon S3 gibt eine Fehlerantwort zurück.

Detaillierte Authentifizierungsdaten

Weitere Informationen zur REST-Authentifizierung finden Sie unter Signieren und Authentifizieren von REST-Anforderungen (p. 821).

Signieren und Authentifizieren von REST-Anforderungen

Themen

> **Note**
>
> In diesem Thema werden AuthentifizierungsAnforderungen unter Verwendung von Signature Version 2 beschrieben. Amazon S3 unterstützt nun die neueste Version, Signature Version 4. Diese neueste Signatur-Version wird in allen Regionen unterstützt. Alle neuen Regionen unterstützen nach dem 30. Januar 2014 nur Signature Version 4. Weitere Informationen finden Sie unter Authentifizieren von Anforderungen (AWS Signature Version 4) in der Amazon Simple Storage Service API Reference.

Die Authentifizierung ist der Prozess, Ihre Identität gegenüber dem System nachzuweisen. Identität ist ein wichtiger Faktor in den Zugriffssteuerungsentscheidungen von Amazon S3. Anforderungen werden zum Teil basierend auf der Identität des Auftraggebers zugelassen oder abgewiesen. Das Recht, Buckets zu erstellen, ist beispielsweise für registrierte Entwickler reserviert (standardmäßig), und das Recht, Objekte in einem Bucket zu erstellen, ist für den Eigentümer des betreffenden Buckets reserviert. Als Entwickler machen Sie Anforderungen, für die diese Berechtigungen gelten müssen, deshalb müssen Sie Ihre Identität gegenüber dem System belegen, indem Sie Ihre Anforderungen authentifizieren. In diesem Abschnitt erfahren Sie mehr darüber.

> **Note**
>
> Der Inhalt dieses Abschnitts gilt nicht für HTTP POST. Weitere Informationen finden Sie unter Browserbasierte Uploads mit POST (AWS Signature-Version 2) (p. 832).

Die Amazon S3 REST API verwendet ein allgemeines HTTP-Schema basierend auf einem verschlüsselten HMAC (Hash Message Authentication Code) für die Authentifizierung. Um eine Anforderung zu authentifizieren, verknüpfen Sie zuerst ausgewählte Elemente der Anforderung, um eine Zeichenfolge zu erstellen. Anschließend verwenden Sie Ihren geheimen AWS-Zugriffsschlüssel, um den HMAC für diese Zeichenfolge zu berechnen. Informell bezeichnen wir diesen Prozess als „Signieren der Anforderung“. Wir bezeichnen die Ausgabe des HMAC-Algorithmus als die Signatur, weil sie die Sicherheitseigenschaften einer realen Signatur simuliert. Schließlich fügen Sie diese Signatur als Parameter der Anforderung hinzu. Dazu verwenden Sie die in diesem Abschnitt beschriebene Syntax.

Wenn das System eine authentifizierte Anforderung erhält, lädt es den geheimen AWS-Zugriffsschlüssel, den Sie behaupten zu haben, und verwendet ihn genau so, um aus der empfangenen Nachricht eine Signatur zu berechnen. Anschließend vergleicht es die berechnete Signatur mit der Signatur, die der Anforderer vorgezeigt hat. Stimmen die beiden Signaturen überein, schließt das System daraus, dass der Anforderer Zugriff auf den geheimen AWS-Zugriffsschlüssel hat, und damit mit der Genehmigung des Prinzipals handelt, dem der Schlüssel ausgestellt wurde. Stimmen die beiden Signaturen nicht überein, wird die Anforderung verworfen und das System antwortet mit einer Fehlermeldung.

Example Authentifizierte Amazon S3 REST-Anforderung

```
GET /photos/puppy.jpg HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
Date: Mon, 26 Mar 2007 19:37:58 +0000

Authorization: AWS AKIAIOSFODNN7EXAMPLE:frJIUN8DYpKDtOLCwo//yllqDzg=
```

Verwenden von temporären Sicherheitsanmeldeinformationen

Wenn Sie Ihre Anforderung unter Verwendung temporärer Sicherheitsanmeldeinformationen signieren (siehe Senden von Anforderungen (p. 11)), müssen Sie das entsprechende Sicherheitstoken in Ihre Anforderung aufnehmen, indem Sie den `x-amz-security-token`-Header hinzufügen.

Wenn Sie unter Verwendung der AWS Security Token Service API temporäre Sicherheitsanmeldeinformationen erhalten haben, beinhaltet die Antwort temporäre Sicherheitsanmeldeinformationen und ein Sitzungstoken. Sie stellen den Wert des Sitzungstoken im `x-amz-security-token`-Header bereit, wenn Sie Anforderungen an Amazon S3 senden. Informationen zur AWS Security Token Service-API, die von IAM bereitgestellt wird, finden Sie unter Aktion im AWS Security Token Service API Reference-Handbuch.

Der Authentifizierungsheader

Die Amazon S3-REST-API verwendet den HTTP-Standard-Header `Authorization`, um Authentifizierungsinformationen zu übergeben. (Der Name des Standardheaders ist unglücklich gewählt, weil er eine Authentifizierung weitergibt, keine Autorisierung.) Unter dem Amazon S3-Authentifizierungsschema hat der Authorization-Header die folgende Form:

```
Authorization: AWS AWSAccessKeyId:Signature
```

Entwicklern wird eine AWS-Zugriffsschlüssel-ID und ein geheimer AWS-Zugriffsschlüssel ausgestellt, wenn sie sich anmelden. Für die Anforderungsauthentifizierung identifiziert das `AWSAccessKeyId`-Element die Zugriffsschlüssel-ID, die für die Berechnung der Signatur verwendet wurde, und indirekt für den Entwickler steht, der die Anforderung gestellt hat.

Das `Signature`-Element ist das RFC 2104 HMAC-SHA1 ausgewählter Elemente aus der Anforderung, der `Signature`-Teil des Authorization-Headers unterscheidet sich deshalb zwischen verschiedenen Anforderungen. Wenn die vom System berechnete Anforderungssignatur mit der in der Anforderung enthaltenen `Signature` übereinstimmt, hat der Auftraggeber belegt, dass er den geheimen AWS-Zugriffsschlüssel besitzt. die Anforderung wird unter der Identität verarbeitet, und mit der Genehmigung des Entwicklers, dem der Schlüssel ausgestellt wurde.

Die folgende Pseudogrammatik verdeutlicht den Aufbau des `Authorization`-Anforderungs-Headers. (In dem Beispiel steht `\n` für den Unicode-Code `U+000A`, häufig auch als Newline bezeichnet.)

```
Authorization = "AWS" + " " + AWSAccessKeyId + ":" + Signature;

Signature = Base64( HMAC-SHA1( YourSecretAccessKey, UTF-8-Encoding-Of( StringToSign ) ) );

StringToSign = HTTP-Verb + "\n" +
 Content-MD5 + "\n" +
 Content-Type + "\n" +
 Date + "\n" +
 CanonicalizedAmzHeaders +
 CanonicalizedResource;

CanonicalizedResource = [ "/" + Bucket ] +
 <HTTP-Request-URI, from the protocol name up to the query string> +
 [ subresource, if present. For example "?acl", "?location", "?logging", or "?torrent"];

CanonicalizedAmzHeaders = <described below>
```

HMAC-SHA1 ist ein von RFC 2104 - Keyed-Hashing for Message Authentication definierter Algorithmus. Der Algorithmus nimmt zwei Byte-Zeichenfolgen als Eingabe entgegen, einen Schlüssel und eine Nachricht. Verwenden Sie für die Authentifizierung von Amazon S3-Anforderungen Ihren geheimen AWS-

Zugriffsschlüssel (`YourSecretAccessKey`) als Schlüssel und die UTF-8-Codierung von `StringToSign` als Nachricht. Die Ausgabe von HMAC-SHA1 ist ebenfalls eine Byte-Zeichenfolge, der sogenannte Digest. Der `Signature`-Anforderungsparameter wird durch eine Base64-Codierung dieses Digest erstellt.

Anfordern einer Kanonisierung für die Signatur

Wenn das System eine authentifizierte Anforderung erhält, vergleicht es die berechnete Anforderungssignatur mit der in der Anforderung im `StringToSign` breitgestellten Signatur, wie bereits beschrieben. Aus diesem Grund müssen Sie die Signatur nach derselben Methode berechnen, wie Amazon S3 sie verwendet. Wir bezeichnen diesen Prozess, bei dem eine Anforderung in die für die Signatur vereinbarte Form gebracht wird, als Kanonisierung.

Erstellen des CanonicalizedResource-Elements

`CanonicalizedResource` stellt die Amazon S3-Ressource dar, die das Ziel der Anforderung ist. Für eine REST-Anforderung erstellen Sie sie wie folgt:

Starten des Prozesses

1	Beginnen Sie mit einer leeren Zeichenfolge (`""`).
2	Wenn die Anforderung unter Verwendung des HTTP Host-Headers (virtueller gehosteter Stils) einen Bucket angibt, fügen Sie den Bucket-Namen mit vorgestelltem `"/"` an (z. B. „/bucketname“). Für Anforderungen im Pfad-Stil und Anforderungen, die an keinen spezifischen Bucket gerichtet sind, machen Sie nichts. Weitere Informationen zum Wiederholen von Anforderungen finden Sie unter Virtuelles Hosting bei Buckets (p. 48). Für die Virtual Hosted-Anforderung „https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg“ lautet die `CanonicalizedResource` „/awsexamplebucket1“. Für die Path-Anforderung „https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg“ lautet `CanonicalizedResource` „“.
3	Fügen Sie den Pfad-Abschnitt der nicht decodierten HTTP Anforderungs-URI bis zur (aber nicht inklusive der) Abfragezeichenfolge ein. Für die Virtual Hosted-Anforderung „https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg“ lautet die `CanonicalizedResource` „/awsexamplebucket1/photos/puppy.jpg“. Für die Path-Anforderung „https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg“ lautet die `CanonicalizedResource` „/awsexamplebucket1/photos/puppy.jpg“. An dieser Stelle ist der `CanonicalizedResource` für Anforderungen mit virtuell gehostetem Stil und Pfad-Stil gleich. Im Fall einer Anforderung, die sich nicht an einen Bucket richtet, wie GET Service, fügen Sie „/“ an.
4	Wenn sich die Anforderung an eine Subressource richtet, wie beispielsweise `?versioning`, `?location`, `?acl`, `?torrent`, `?lifecycle` oder `?versionid`, fügen Sie die Subressource an, gegebenenfalls ihren Wert und das Fragezeichen. Beachten Sie, dass mehrere Subressourcen alphabetisch nach dem Namen der Subressource sortiert und durch '&' voneinander getrennt werden müssen, z. B. ?acl&versionId=*value*. Die Subressourcen, die bei der Erstellung des CanonicalizedResource-Elements angegeben werden müssen, sind acl, lifecycle, location, logging, notification, partNumber, policy, requestPayment, torrent, uploadId, uploads, versionId, versioning, versions und website. Wenn die Anforderung Abfragezeichenfolgen-Parameter angibt, die die Antwortheader-Werte überschreiben (siehe Get Object), fügen Sie die Abfragezeichenfolgen-Parameter und ihre Werte an. Bei einer Signatur codieren Sie diese Werte nicht. Bei einer Anforderung dagegen müssen

	Sie diese Parameterwerte codieren. Die Abfrageparameter in einer GET-Anforderung sind unter anderem `response-content-type`, `response-content-language`, `response-expires`, `response-cache-control`, `response-content-disposition` und `response-content-encoding`. Wenn Sie die CanonicalizedResource für eine Löschanforderung für mehrere Objekte erstellen, muss der Abfrage-Zeichenfolgenparameter `delete` angegeben werden.

Elemente der CanonicalizedResource, die aus der HTTP Request-URI stammen, sollten genauso signiert werden, wie sie in der HTTP-Anforderung erscheinen, einschließlich der Metazeichen der URL-Codierung.

Möglicherweise unterschiedet sich die `CanonicalizedResource` von der HTTP Anforderungs-URI. Wenn Ihre Anforderung den HTTP-Header `Host` verwendet, um einen Bucket anzugeben, erscheint der Bucket nicht in der HTTP Anforderungs-URI. Die `CanonicalizedResource` beinhaltet den Bucket jedoch weiterhin. Abfrage-Zeichenfolgenparameter können auch in der Anforderungs-URI erscheinen, sind aber nicht in der `CanonicalizedResource` enthalten. Weitere Informationen finden Sie unter Virtuelles Hosting bei Buckets (p. 48).

Erstellen des CanonicalizedAmzHeaders-Elements

Um den CanonicalizedAmzHeaders-Teil von `StringToSign` zu erstellen, wählen Sie alle HTTP-Anforderungsheader aus, die mit 'x-amz-' beginnen (über einen Vergleich, der die Groß-/Kleinschreibung nicht berücksichtigt), und gehen nach dem folgenden Verfahren vor.

CanonicalizedAmzHeaders-Verfahren

1	Wandeln Sie jeden HTTP-Headernamen in Kleinbuchstaben um. Beispielsweise wird '`X-Amz-Date`' zu '`x-amz-date`'.
2	Sortieren Sie die Header alphabetisch nach dem Headernamen.
3	Kombinieren Sie Headerfelder mit demselben Namen zu einem „header-name:comma-separated-value-list“-Paar, wie von RFC 2616 Abschnitt 4.2 vorgegeben, ohne Leerzeichen zwischen den Werten. Die beiden Metadaten-Header '`x-amz-meta-username: fred`' und '`x-amz-meta-username: barney`' würden beispielsweise zu einem einzigen Header '`x-amz-meta-username: fred,barney`' zusammengefasst.
4	„Falten“ Sie lange Header auf, die sich über mehrere Zeilen erstrecken (wie durch RFC 2616 Abschnitt 4.2 erlaubt), indem Sie den faltenden Whitespace (einschließlich von Neue-Zeile-Zeichen) durch ein einzelnes Leerzeichen ersetzen.
5	Entfernen Sie Whitespace um den Doppelpunkt im Header. Beispielsweise würde der Header '`x-amz-meta-username: fred,barney`' zu '`x-amz-meta-username:fred,barney`'
6	Schließlich fügen Sie jedem kanonisierten Header in der Ergebnisliste ein Neue-Zeile-Zeichen (`U+000A`) hinzu. Erstellen Sie das CanonicalizedResource-Element, indem Sie alle Header in dieser Liste zu einer einzelnen Zeichenfolge zusammenfassen.

Positionale vs. benannte StringToSign-Elemente von HTTP-Headern

Die ersten Header-Elemente von `StringToSign` (Content-Type, Date und Content-MD5) sind von ihrer Art her positionsbezogen. `StringToSign` fügt nicht die Namen dieser Header, sondern nur ihre Werte aus der Anforderung ein. Im Gegensatz dazu sind die '`x-amz-`'-Elemente benannt. Sowohl die Header-Namen als auch die Header-Werte erscheinen in `StringToSign`.

Wenn ein positionaler Header, der in der Definition von `StringToSign` vorgegeben ist, in Ihrer Anforderung nicht vorhanden ist (z. B. sind `Content-Type` oder `Content-MD5` optionale für PUT-Anforderungen und sinnlos für GET-Anforderungen), setzen Sie in dieser Position die leere Zeichenfolge ("") ein.

Zeitstempel-Anforderung

Ein gültiger Zeitstempel (unter Verwendung des HTTP-Headers `Date` oder einer `x-amz-date`-Alternative) ist zwingend erforderlich für authentifizierte Anforderungen. Darüber hinaus muss der Client-Zeitstempel in einer authentifizierten Anforderung innerhalb eines Zeitraums von 15 Minuten zur Amazon S3-Systemzeit liegen, wenn die Anforderung empfangen wird. Wenn dies nicht der Fall ist, schlägt die Anforderung mit `RequestTimeTooSkewed`-Fehlercode fehl. Diese Einschränkungen sollen verhindern, dass abgefangene Anforderungen böswillig wiederholt werden. Für einen strengeren Schutz gegen ein Abhören transportieren Sie authentifizierte Anforderung mit HTTPS.

> Note
>
> Die Auswertungsbeschränkung im Hinblick auf das Anforderungsdatum gilt nur für authentifizierte Anforderungen, die keine Abfrage-String-Authentifizierung verwenden. Weitere Informationen finden Sie unter Alternative zur Authentifizierung der Abfragezeichenfolge (p. 830).

Einige HTTP-Client-Bibliotheken unterstützen die Möglichkeit nicht, den `Date`-Header für eine Anforderung einzurichten. Wenn Sie Probleme damit haben, den Wert des 'Date'-Headers in kanonisierte Header aufzunehmen, können Sie den Zeitstempel für die Anforderung stattdessen auch mit einem '`x-amz-date`'-Header einrichten. Der Wert des `x-amz-date`-Headers muss in einem der von RFC 2616 vorgegebenen Formate vorliegen (http://www.ietf.org/rfc/rfc2616.txt). Wenn in einer Anforderung ein `x-amz-date`-Header vorhanden ist, ignoriert das System bei der Berechnung der Anforderungssignatur jeden `Date`-Header. Wenn Sie also den `x-amz-date`-Header aufnehmen, verwenden Sie die leere Zeichenfolge für `Date`, wenn Sie den `StringToSign` erstellen. Ein Beispiel finden Sie im nächsten Abschnitt.

Authentifizierungsbeispiele

Die Beispiele in diesem Abschnitt verwenden die (nicht funktionierenden) Anmeldeinformationen aus der folgenden Tabelle.

Parameter	Value
AWSAccessKeyId	`AKIAIOSFODNN7EXAMPLE`
AWSSecretAccessKey	`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`

In den `StringToSigns` des Beispiels ist die Formatierung nicht relevant, und `\n` steht für den Unicode-Punkt `U+000A`, allgemein als Neue-Zeile-Zeichen bezeichnet. Außerdem verwenden die Beispiele "+0000", um die Zeitzone anzugeben. Sie können die Zeitzone stattdessen mit "GMT" angeben, aber in den Beispielen werden andere Signaturen verwendet.

Object GET

In diesem Beispiel wird ein Objekt aus dem Bucket „awsexamplebucket1“ abgerufen.

Anforderung	StringToSign
`GET /photos/puppy.jpg HTTP/1.1`	`GET\n` `\n`

Anforderung	StringToSign
`Host: awsexamplebucket1.us-west-1.s3.amazonaws.com` `Date: Tue, 27 Mar 2007 19:36:42 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE: bWq2s1WEIj+Ydj0vQ697zp+IXMU=`*	`\n` `Tue, 27 Mar 2007 19:36:42 +0000\n` `/awsexamplebucket1/photos/puppy.jpg`

Beachten Sie, dass die CanonicalizedResource den Bucket-Namen beinhaltet, die HTTP Anforderungs-URI dagegen nicht. (Der Bucket wird vom Host-Header spezifiziert.)

Object PUT

In diesem Beispiel wird ein Objekt in den Bucket „awsexamplebucket1“ eingefügt.

Anforderung	StringToSign
`PUT /photos/puppy.jpg HTTP/1.1` `Content-Type: image/jpeg` `Content-Length: 94328` `Host: awsexamplebucket1.s3.us-west-1.amazonaws.com` `Date: Tue, 27 Mar 2007 21:15:45 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE: MyyxeRY7whkBe+bq8fHCL/2kKUg=`*	`PUT\n` `\n` `image/jpeg\n` `Tue, 27 Mar 2007 21:15:45 +0000\n` `/awsexamplebucket1/photos/puppy.jpg`

Beachten Sie den Content-Type-Header in der Anforderung und im StringToSign. Beachten Sie außerdem, dass Content-MD5 im StringToSign leer ist, weil es in der Anforderung nicht vorhanden ist.

Liste

In diesem Beispiel wird der Inhalt des Buckets „awsexamplebucket1“ aufgelistet.

Anforderung	StringToSign
`GET /?prefix=photos&max-keys=50&marker=puppy HTTP/1.1` `User-Agent: Mozilla/5.0` `Host: awsexamplebucket1.s3.us-west-1.amazonaws.com` `Date: Tue, 27 Mar 2007 19:42:41 +0000` *`Authorization: AWS AKIAIOSFODNN7EXAMPLE: htDYFYduRNen8P9ZfE/s9SuKy0U=`*	`GET\n` `\n` `\n` `Tue, 27 Mar 2007 19:42:41 +0000\n` `/awsexamplebucket1/`

Beachten Sie den abschließenden Schrägstrich an der CanonicalizedResource sowie das Fehlen von Abfrage-Zeichenfolgenparametern.

Fetch

In diesem Beispiel wird die Zugriffskontrollrichtlinien-Subressource für den Bucket „awsexamplebucket1“ abgerufen.

Anforderung	StringToSign
`GET /?acl HTTP/1.1` `Host: awsexamplebucket1.s3.us-west-1.amazonaws.com` `Date: Tue, 27 Mar 2007 19:44:46 +0000` `Authorization: AWS AKIAIOSFODNN7EXAMPLE:` `c2WLPFtWHVgbEmeEG93a4cG37dM=`	`GET\n` `\n` `\n` `Tue, 27 Mar 2007 19:44:46` `+0000\n` `/awsexamplebucket1/?acl`

Beachten Sie, wie der Subressourcen-Abfragezeichenfolgenparameter in die CanonicalizedResource aufgenommen wird.

Löschen

In diesem Beispiel wird ein Objekt über die path-style- und Date-Alternative aus dem Bucket „awsexamplebucket1“ entfernt.

Anforderung	StringToSign
`DELETE /awsexamplebucket1/photos/puppy.jpg` `HTTP/1.1` `User-Agent: dotnet` `Host: s3.us-west-1.amazonaws.com` `Date: Tue, 27 Mar 2007 21:20:27 +0000` `x-amz-date: Tue, 27 Mar 2007 21:20:26 +0000` `Authorization: AWS` `AKIAIOSFODNN7EXAMPLE:lx3byBScXR6KzyMaifNkardMwNk=`	`DELETE\n` `\n` `\n` `Tue, 27 Mar 2007 21:20:26 +0000\n` `/awsexamplebucket1/photos/puppy.jpg`

Beachten Sie, wie wir die alternative Methode 'x-amz-date' verwendet haben, um das Datum anzugeben (beispielsweise weil unsere Client-Bibliothek verhindert hat, dass wir das Datum festgelegt haben). In diesem Fall hat `x-amz-date` Vorrang vor dem `Date`-Header. Aus diesem Grund muss der Datumseintrag in der Signatur den Wert des `x-amz-date`-Headers enthalten.

Hochladen

Dieses Beispiel lädt ein Objekt in einen virtuell gehosteten Bucket mit CNAME-Stil mit Metadaten.

Anforderung	StringToSign
`PUT /db-backup.dat.gz HTTP/1.1` `User-Agent: curl/7.15.5` `Host: static.awsexamplebucket1.net:8080` `Date: Tue, 27 Mar 2007 21:06:08 +0000` `x-amz-acl: public-read` `content-type: application/x-download` `Content-MD5: 4gJE4saaMU4BqNR0kLY+lw==` `X-Amz-Meta-ReviewedBy:` `joe@awsexamplebucket1.net` `X-Amz-Meta-ReviewedBy:` `jane@awsexamplebucket1.net` `X-Amz-Meta-FileChecksum: 0x02661779` `X-Amz-Meta-ChecksumAlgorithm: crc32`	`PUT\n` `4gJE4saaMU4BqNR0kLY+lw==\n` `application/x-download\n` `Tue, 27 Mar 2007 21:06:08 +0000\n` `x-amz-acl:public-read\n` `x-amz-meta-checksumalgorithm:crc32\n` `x-amz-meta-filechecksum:0x02661779\n` `x-amz-meta-reviewedby:` `joe@awsexamplebucket1.net,jane@awsexamplebucket1.net` `\n` `/static.awsexamplebucket1.net/db-` `backup.dat.gz`

Anforderung	StringToSign
`Content-Disposition: attachment;` `  filename=database.dat` `Content-Encoding: gzip` `Content-Length: 5913339` `Authorization: AWS AKIAIOSFODNN7EXAMPLE:` `ilyl83RwaSoYIEdixDQcA4OnAnc=`	

Beachten Sie, wie die 'x-amz-'-Header sortiert, von Whitespaces befreit und in Kleinbuchstaben umgewandelt wurden. Beachten Sie außerdem, dass mehrere Header mit demselben Namen unter Verwendung von Kommas zum Trennen der Werte verknüpft wurden.

Beachten Sie, wie nur die HTTP-Header `Content-Type` und `Content-MD5` in `StringToSign` erscheinen. Die anderen `Content-*`-Header erscheinen nicht.

Beachten Sie ebenfalls, dass`CanonicalizedResource` den Bucket-Namen beinhaltet, die HTTP Anforderungs-URI dagegen nicht. (Der Bucket wird vom Host-Header spezifiziert.)

Auflisten aller meiner Buckets

Anforderung	StringToSign
`GET / HTTP/1.1` `Host: s3.us-west-1.amazonaws.com` `Date: Wed, 28 Mar 2007 01:29:59 +0000` `Authorization: AWS` `AKIAIOSFODNN7EXAMPLE:qGdzdERIC03wnaRNKh6OqZehG9s=`	`GET\n` `\n` `\n` `Wed, 28 Mar 2007 01:29:59` `+0000\n` `/`

Unicode-Schlüssel

Anforderung	StringToSign
`GET /dictionary/fran%C3%A7ais/pr%c3%a9f` `%c3%a8re HTTP/1.1` `Host: s3.us-west-1.amazonaws.com` `Date: Wed, 28 Mar 2007 01:49:49 +0000` `Authorization: AWS` `AKIAIOSFODNN7EXAMPLE:DNEZGsoieTZ92F3bUfSPQcbGm%C3%a8re`	`GET\n` `\n` `\n` `Wed, 28 Mar 2007 01:49:49 +0000\n` `/dictionary/fran%C3%A7ais/pr%c3%a9f` `%c3%a8re`

> **Note**
>
> Die Elemente in `StringToSign`, die von der Anforderungs-URI abgeleitet wurden, werden unverändert übernommen, einschließlich der URL-Codierung und der Großschreibung.

Probleme mit dem Signieren von REST-Anforderungen

Wenn die Authentifizierung einer REST-Anforderung fehlschlägt, reagiert das System mit einem XML-Fehlerdokument auf die Anforderung. Die in diesem Fehlerdokument enthaltene Information soll

Entwicklern helfen, das Problem zu diagnostizieren. Insbesondere erkennen Sie an dem `StringToSign`-Element des `SignatureDoesNotMatch`-Fehlerdokuments genau, welche Anforderungs-Kanonisierung das System verwendet.

Einige Toolkits fügen stillschweigend Header ein, die Sie zuvor nicht kennen, wie beispielsweise den Header `Content-Type` bei einem PUT. In den meisten dieser Fälle bleibt der Wert des eingefügten Headers konstant, sodass Sie fehlende Header unter Verwendung von Tools wie Ethereal oder tcpmon erkennen können.

Alternative zur Authentifizierung der Abfragezeichenfolge

Einige Anforderungstypen können Sie authentifizieren, indem Sie die angeforderten Informationen als Abfragezeichenfolgenparameter übergeben, statt den HTTP-Header `Authorization` zu verwenden. Das ist praktisch, um direkten Zugriff durch einen Browser von Dritten auf Ihre privaten Amazon S3-Daten zu ermöglichen, ohne dass die Anforderung über einen Proxy geschickt wird. Die Idee besteht in der Konstruierung einer „vorsignierten“ Anforderung und ihrer Codierung als einer URL, die vom Browser eines Endbenutzers geladen werden kann. Darüber hinaus können Sie eine vorsignierte Anforderung durch die Angabe einer Ablaufzeit begrenzen.

> Note
>
> Beispiele für die Verwendung von AWS SDKs zum Generieren vorsignierter URLs finden Sie unter Ein Objekt mit anderen teilen (p. 190).

Erstellen einer Signatur

Das folgende Beispiel zeigt eine authentifizierte Amazon S3 REST-Anforderung mit Abfragezeichenfolge.

```
GET /photos/puppy.jpg
?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=1141889120&Signature=vjbyPxybdZaNmGa
%2ByT272YEAiv4%3D HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
Date: Mon, 26 Mar 2007 19:37:58 +0000
```

Die Authentifizierungsmethode für die Abfragezeichenkette benötigt keine spezifischen HTTP-Header. Stattdessen werden die erforderlichen Authentifizierungselemente als Abfragezeichenfolgenparameter angegeben:

Abfragezeichenfolge Parametername	Beispielwert	Beschreibung
`AWSAccessKeyId`	`AKIAIOSFODNN7EXAMPLE`	Ihre AWS-Zugriffsschlüssel-ID. Gibt den geheimen AWS-Zugriffsschlüssel an, der verwendet wurde, um die Anforderung zu signieren, und indirekt die Identität des Entwicklers angibt, der die Anforderung gestellt hat.
`Expires`	`1141889120`	Die Zeit, wann die Signatur abläuft, angegeben als die Anzahl der Sekunden, die seit dem 1. Januar 1970 00:00:00 UTC verstrichen sind. Eine Anforderung, die nach dieser Zeit empfangen wird (abhängig vom Server), wird abgelehnt.

Abfragezeichenfolge Parametername	Beispielwert	Beschreibung
`Signature`	`vjbyPxybdZaNmGa %2ByT272YEAiv4%3D`	Die URL-Codierung der Base64-Codierung des HMAC-SHA1 für StringToSign.

Die Methode zur Authentifizierung der Abfragezeichenfolge unterscheidet sich leicht von der üblichen Methode, aber nur im Format des `Signature`-Anforderungsparameters und des `StringToSign`-Elements. Die folgende Pseudogrammatik verdeutlicht die Authentifizierungsmethode für die Abfragezeichenfolge.

```
Signature = URL-Encode( Base64( HMAC-SHA1( YourSecretAccessKey, UTF-8-Encoding-
Of( StringToSign ) ) ) );

StringToSign = HTTP-VERB + "\n" +
    Content-MD5 + "\n" +
    Content-Type + "\n" +
    Expires + "\n" +
    CanonicalizedAmzHeaders +
    CanonicalizedResource;
```

`YourSecretAccessKey` ist die ID des geheimen AWS-Zugriffsschlüssels, die Ihnen Amazon zuordnet, wenn Sie sich als Amazon Web Service-Entwickler anmelden. Beachten Sie, wie die `Signature` URL-codiert wird, damit sie für die Platzierung im Abfragestring geeignet ist. Beachten Sie auch, dass in `StringToSign` das HTTP-Positionselement `Date` durch `Expires` ersetzt wurde. `CanonicalizedAmzHeaders` und `CanonicalizedResource` sind gleich.

> **Note**
>
> In der Methode für die Abfrage-String-Authentifizierung verwenden Sie den Header `Date` oder `x-amz-date request` nicht, wenn Sie die zu signierende Zeichenfolge berechnen.

Authentifizierung von Abfragezeichenfolgen-Anforderungen

Anforderung	StringToSign
`GET /photos/puppy.jpg? AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE& Signature=NpgCjnDzrM %2BWFzoENXmpNDUsSn8%3D& Expires=1175139620 HTTP/1.1` `Host: awsexamplebucket1.s3.us-west-1.amazonaws.com`	`GET\n \n \n 1175139620\n` `/awsexamplebucket1/photos/puppy.jpg`

Wir gehen davon aus, dass ein Browser bei einer GET-Anforderung keinen Content-MD5- oder Content-Type-Header bereitstellt, und auch keine x-amz--Header einrichten, diese Teile von `StringToSign` bleiben deshalb leer.

Verwendung der Base64-Codierung

Signaturen von HMAC-Anforderungen müssen mit Base64 codiert werden. Die Base64-Codierung wandelt die Signatur in einen einfachen ASCII-String um, der der Anforderung hinzugefügt werden kann. Zeichen, die in der Signatur erscheinen können, wie beispielsweise Plus (+), Schrägstrich (/)

oder Gleichheitszeichen (=), müssen wie in einer URI codiert werden. Enthält beispielsweise der Authentifizierungscode ein Plussymbol (+), codieren Sie es in der Anforderung als &2B. Einen Schrägstrich codieren Sie als &2F, ein Gleichheitszeichen als %3D.

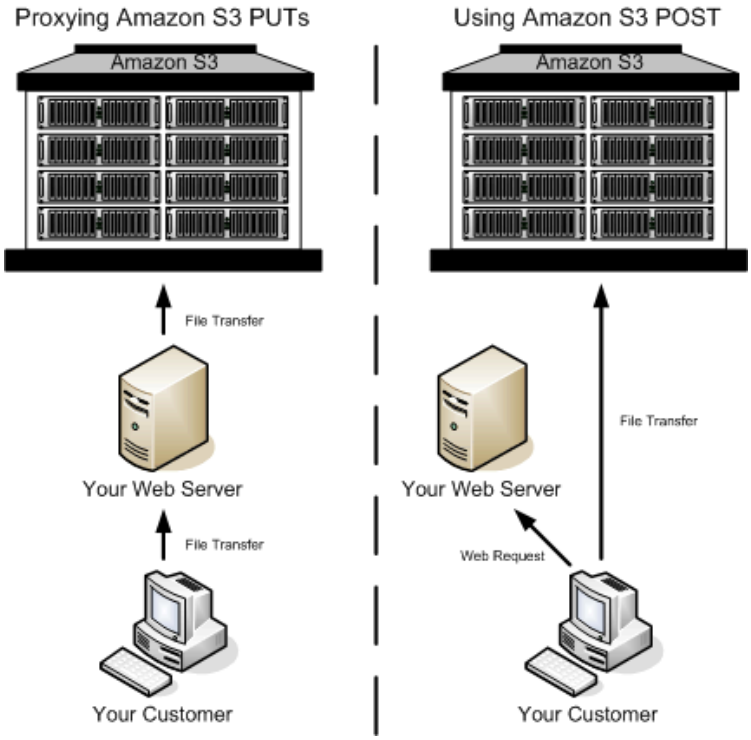
Beispiele für die Base64-Codierung finden Sie in den Authentifizierungsbeispiele (p. 826) für Amazon S3.

Browserbasierte Uploads mit POST (AWS Signature-Version 2)

Amazon S3 unterstützt POST, das Ihren Benutzern das direkte Hochladen von Inhalten zu Amazon S3 ermöglicht. POST dient dazu, Uploadvorgänge zu vereinfachen, die Latenz bei Uploads zu reduzieren, und Ihnen bei Anwendungen Geld zu sparen, bei denen Benutzer Daten zur Speicherung auf Amazon S3 hochladen.

> Note
>
> Die in diesem Abschnitt besprochene Anfrageauthentifizierung basiert auf AWS Signature Version 2, einem Protokoll für die Authentifizierung eingehender API-Anfragen an AWS-Services. Amazon S3 unterstützt jetzt Signature Version 4, ein Protokoll für die Authentifizierung eingehender API-Anfragen an AWS-Services, in allen AWS-Regionen. Derzeit unterstützen vor dem 30. Januar 2014 erstellte AWS-Regionen weiterhin das vorherige Protokoll, Signature Version 2. Alle nach dem 30. Januar 2014 neu erstellten Regionen unterstützen nur Signature Version 4, weshalb alle Anfragen an diese Regionen Signature Version 4 verwenden müssen. Weitere Informationen finden Sie unter Authentifizieren von Anforderungen in browserbasierten Uploads mit POST (AWS Signature Version 4) im Amazon Simple Storage Service API Reference.

Die folgende Abbildung zeigt einen Upload mit Amazon S3 POST.



Upload mit POST

1	Der Benutzer öffnet einen Webbrowser, und greift auf Ihre Webseite zu.

2	Ihre Webseite enthält ein HTTP-Formular mit allen Informationen, die der Benutzer benötigt, um Inhalte zu Amazon S3 hochladen zu können.
3	Der Benutzer lädt die Inhalte direkt zu Amazon S3 hoch.

> **Note**
>
> Die Abfrage-String-Authentifizierung wird für POST nicht unterstützt.

HTML-Formulare (AWS Signature Version 2)

Themen

Wenn Sie mit Amazon S3 kommunizieren, verwenden Sie normalerweise die REST- oder >SOAP-API zum Laden, Abrufen, Löschen und für andere Operationen. Mit POST laden Benutzer Daten direkt über Browser zu Amazon S3 hoch, die weder die SOAP-API verarbeiten noch eine REST `PUT`-Anfrage erstellen können.

> **Note**
>
> Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.

Um Benutzern zu erlauben, Inhalte über ihre Browser zu Amazon S3 hochzuladen, verwenden Sie HTML-Formulare. HTML-Formulare bestehen aus einer Formulardeklaration und Formularfeldern. Die Formulardeklaration enthält allgemeine Informationen zu der Anfrage. Die Formularfelder enthalten detaillierte Informationen zu der Anfragen sowie die Richtlinie, die verwendet wird, um die Anfrage zu authentifizieren und um sicherzustellen, dass sie den von Ihnen angegebenen Bedingungen entspricht.

> **Note**
>
> Daten und Grenzen des Formulars (ausschließlich der Inhalte der Datei) dürfen 20 KB nicht überschreiten.

Dieser Abschnitt erläutert die Verwendung von HTML-Formularen.

HTML-Formular-Kodierung

Das Formular und die Richtlinie müssen gemäß UTF-8 kodiert sein. Sie können die UTF-8-Kodierung für das Formular anwenden, indem Sie sie im HTML-Kopf oder als Anfragekopf angeben.

> **Note**
>
> Die HTML-Formulardeklaration akzeptiert keine Abfrage-String-Authentifizierungsparameter.

Nachfolgend sehen Sie ein Beispiel für die UTF-8-Kodierung im HTML-Kopf:

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

```
  ...
</head>
<body>
```

Nachfolgend sehen Sie ein Beispiel für die UTF-8-Kodierung im Anfragekopf:

```
Content-Type: text/html; charset=UTF-8
```

HTML-Formulardeklaration

Die Formulardeklaration besteht aus drei Komponenten: der Aktion, der Methode und dem Umschließungstyp. Wenn einer dieser Werte nicht korrekt eingestellt ist, schlägt die Anforderung fehl.

Die Aktion gibt die URL an, die die Anforderung verarbeitet; dies muss die URL des Buckets sein. Wenn der Name Ihres Buckets beispielsweise `awsexamplebucket1` ist und die Region USA West (Nordkalifornien) ist, lautet die URL `https://awsexamplebucket1.s3.us-west-1.amazonaws.com/`.

> **Note**
>
> Der Schlüsselname wird in einem Formularfeld angegeben.

Die Methode muss POST sein.

Der Umschließungstyp (enctype) muss angegeben werden und auf Multipart/Form-Daten für Datei- und Textbereich-Uploads gesetzt sein. Weitere Informationen finden Sie unter RFC 1867.

Example

Das folgende Beispiel zeigt eine Formulardeklaration für den Bucket „awsexamplebucket1“.

```
<form action="https://awsexamplebucket1.s3.us-west-1.amazonaws.com/" method="post"

enctype="multipart/form-data">
```

HTML-Formularfelder

Die folgende Tabelle beschreibt die Felder, die in einem HTML-Formular verwendet werden können.

> **Note**
>
> Die Variable `${filename}` wird automatisch durch den Namen der Datei ersetzt, den der Benutzer bereitstellt, und der von allen Formularfeldern anerkannt wird. Wenn der Browser oder Client einen vollständigen oder teilweisen Pfad zu der Datei bereitstellt, wird nur der Text nach dem letzten Schrägstrich (/) oder umgekehrten Schrägstrich (\) verwendet. Beispielsweise wird „C:\Program Files\directory1\file.txt“ als „file.txt“ interpretiert. Wenn keine Datei bzw. kein Dateiname angegeben ist, wird die Variable durch eine leere Zeichenfolge ersetzt.

Feldname	Beschreibung	Erforderlich
`AWSAccessKeyId`	Die AWS-Zugriffsschlüssel-ID des Eigentümers des Buckets, der den anonymen Benutzerzugriff für eine Anforderung gewährt, die den Einschränkungen in der Richtlinie entspricht. Dies ist ein Pflichtfeld, wenn die Anforderung ein Richtliniendokument beinhaltet.	Bedingt

Feldname	Beschreibung	Erforderlich
`acl`	Eine Amazon S3-Zugriffskontrollliste (ACL). Wenn eine ungültige Zugriffskontrollliste angegeben ist, wird ein Fehler generiert. Weitere Informationen zu ACLs finden Sie unter Zugriffskontrolllisten (p. 8). Typ: Zeichenkette Standard: privat Valid Values: `private | public-read | public-read-write | aws-exec-read | authenticated-read | bucket-owner-read | bucket-owner-full-control`	Nein
`Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires`	REST-spezifische Köpfe. Weitere Informationen finden Sie unter PUT Object.	Nein
`key`	Der Name des hochgeladenen Schlüssels. verwenden Sie die Variable $(filename), um den von dem Benutzer angegebenen Dateinamen zu verwenden. Zum Beispiel: Wenn Benutzerin „Betty“ die Datei „lolcatz.jpg“ hochlädt, und Sie „/user/betty/${filename}“ angeben, wird die Datei als „/user/betty/lolcatz.jpg“ gespeichert. Weitere Informationen finden Sie unter Objektschlüssel und Metadaten (p. 117).	Ja
`policy`	Die Sicherheitsrichtlinie, die angibt, was in der Anforderung zulässig ist. Anforderungen ohne Sicherheitsrichtlinie gelten als anonym und sind nur auf öffentlich zugänglichen Buckets erfolgreich.	Nein

Feldname	Beschreibung	Erforderlich
`success_action_redirect, redirect`	Die URL, an die der Client nach einem erfolgreichen Upload umgeleitet wird. Amazon S3 fügt der URL die Bucket-, Schlüssel- und eTag-Werte als Abfragezeichenfolgen-Parameter an. Wenn success_action_redirect nicht angegeben ist, gibt Amazon S3 den leeren Dokumententyp zurück, der im Feld success_action_status angegeben ist. Wenn Amazon S3 die URL nicht interpretieren kann, wird das Feld ignoriert. Wenn der Upload fehlschlägt, zeigt Amazon S3 einen Fehler an und leitet den Benutzer nicht zu einer URL um. Weitere Informationen finden Sie unter Umleitung (p. 841). Note Der Name des Umleitungsfeldes ist veraltet, und das Feld wird künftig nicht mehr unterstützt.	Nein
`success_action_status`	Der an den Client bei einem erfolgreichen Upload ausgegebene Statuscode, wenn success_action_redirect nicht angegeben ist. Gültige Werte sind 200, 201 und 204 (Standard). Wenn der Wert auf 200 oder 204 gesetzt ist, gibt Amazon S3 ein leeres Dokument mit dem Statuscode 200 oder 204 aus. Wenn der Wert auf 201 gesetzt ist, gibt Amazon S3 ein XML-Dokument mit dem Statuscode 201 aus. Informationen zum Inhalt des XML-Dokuments finden Sie unter POST Object. Wenn der Wert nicht oder auf einen falschen Wert gesetzt ist, gibt Amazon S3 ein leeres Dokument mit dem Statuscode 204 aus. Note Einige Versionen von Adobe Flash Player können HTTP-Antworten ohne Text nicht korrekt bearbeiten. Zur Unterstützung von Uploads über Adobe Flash empfehlen wir, `success_action_status` auf 201 zu setzen.	Nein

Feldname	Beschreibung	Erforderlich
`signature`	Die HMAC-Signatur, die mithilfe des geheimen Zugriffsschlüssels konstruiert wird, der der angegebenen AWSAccessKeyId entspricht. Dies ist ein Pflichtfeld, wenn die Anforderung ein Richtliniendokument beinhaltet. Weitere Informationen finden Sie unter Verwendung des Auth-Zugriffs.	Bedingt
`x-amz-security-token`	Ein Sicherheitstoken, das von - Sitzungsanmeldeinformationen verwendet wird Wenn die Anfrage Amazon DevPay verwendet, benötigt sie zwei `x-amz-security-token`-Formularfelder: eins für das Produkttoken und eins für das Benutzertoken. Wenn die Anforderung Sitzungsanmeldedaten verwendet, ist ein `x-amz-security-token`-Formular erforderlich. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch.	Nein
Andere Feldnamen mit dem Präfix x-amz-meta-	Vom Benutzer angegebene Metadaten Amazon S3 validiert oder verwendet diese Daten nicht. Weitere Informationen finden Sie unter PUT Object.	Nein
file	Datei oder Textinhalt. Die Datei oder der Inhalt muss das letzte Feld des Formulars sein. Alle Felder darunter werden ignoriert. Sie können nicht mehr als eine Datei zur gleichen Zeit hochladen.	Ja

Richtlinienerstellung

Themen

Die Richtlinie ist ein in UTF-8 und Base64 kodiertes JSON-Dokument, das die Bedingungen angibt, die die Anforderungen erfüllen muss; sie wird zur Authentifizierung des Inhalts verwendet. Je nach dem, wie Sie Ihre Richtliniendokumente gestalten, können Sie sie pro Upload, pro Benutzer, für alle Uploads oder nach anderen Kriterien, die Ihren Anforderungen entsprechen, verwenden.

Note

Obwohl das Richtliniendokument optional ist, empfehlen nachdrücklich, es zu verwenden, anstatt einen Bucket öffentlich beschreibbar zu machen.

Nachstehend finden Sie ein Beispiel für ein Richtliniendokument:

```
{ "expiration": "2007-12-01T12:00:00.000Z",

  "conditions": [

    {"acl": "public-read" },

    {"bucket": "awsexamplebucket1" },

    ["starts-with", "$key", "user/eric/"],

  ]

}
```

Das Richtliniendokument enthält den Ablauf und die Bedingungen.

Ablauf

Das Ablaufelement gibt das Ablaufdatum der Richtlinie im Datumsformat nach ISO 8601 UTC an. Beispielsweise bedeutet „2007-12-01T12:00:00.000Z“, dass die Richtlinie ab Mitternacht (UTC) am 01. 12. 2007 ungültig ist. Die Angabe des Ablaufs ist in einer Richtlinie erforderlich.

Bedingungen

Die Bedingungen in dem Richtliniendokument validieren den Inhalt des hochgeladenen Objekts. Jedes Formularfeld, das Sie in dem Formular angeben (ausgenommen AWSAccessKeyId, Signatur, Datei, Richtlinie und Feldnamen mit dem Präfix x-ignore-) muss in der Liste der Bedingungen aufgeführt werden.

Note

Wenn Sie mehrere Felder mit dem gleichen Namen haben, müssen die Werte durch Kommata abgeteilt sein. Zum Beispiel: Wenn Sie zwei Felder mit der Bezeichnung „x-amz-meta-tag“ haben und das erste den Wert „Ninja“ und das zweite den Wert „Stallman“ hat, setzen Sie das Richtliniendokument auf `Ninja,Stallman`.
Alle Variablen in dem Formular werden vor der Validierung der Richtlinie erweitert. Daher müssen alle Bedingungsabgleiche anhand der erweiterten Felder vorgenommen werden. Zum Beispiel: Wenn Sie das Schlüsselfeld auf `user/betty/${filename}` setzen, ist Ihre Richtlinie möglicherweise `[ "starts-with", "$key", "user/betty/" ]`. Geben Sie nicht `[ "starts-with", "$key", "user/betty/${filename}" ]` ein. Weitere Informationen finden Sie unter Übereinstimmung von Bedingungen (p. 839).

Die folgende Tabelle beschreibt die Bedingungen für Richtliniendokumente.

Elementname	Beschreibung
acl	Gibt die Bedingungen an, die die ACL erfüllen muss. Unterstützt exakte Übereinstimmung und `starts-with`.
content-length-range	Gibt die erlaubte Mindest- und Höchstgröße des hochgeladenen Inhalts an. Unterstützt den Bereichsabgleich.

Elementname	Beschreibung
Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires	REST-spezifische Köpfe. Unterstützt exakte Übereinstimmung und `starts-with`.
Schlüssel	Der Name des hochgeladenen Schlüssels. Unterstützt exakte Übereinstimmung und `starts-with`.
success_action_redirect, redirect	Die URL, zu der der Client nach einem erfolgreichen Upload umgeleitet wird. Unterstützt exakte Übereinstimmung und `starts-with`.
success_action_status	Der an den Client bei einem erfolgreichen Upload ausgegebene Statuscode, wenn success_action_redirect nicht angegeben ist. Unterstützt exakte Übereinstimmung.
x-amz-security-token	Amazon DevPay-Sicherheitstoken. Jede Anfrage, die Amazon DevPay verwendet, erfordert zwei `x-amz-security-token`-Formularfelder erforderlich: eins für das Produkttoken und eins für das Benutzertoken. Daher müssen die Werte durch Kommata voneinander getrennt werden. Zum Beispiel: Wenn der Benutzer-Token `eW91dHViZQ==` und der Produkt-Token `b0hnNVNKWVJIQTA=` ist, setzen Sie den Richtlinieneintrag auf: `{ "x-amz-security-token": "eW91dHViZQ==,b0hnNVNKWVJIQTA=" }`.
Andere Feldnamen mit dem Präfix x-amz-meta-	Vom Benutzer angegebene Metadaten Unterstützt exakte Übereinstimmung und `starts-with`.

> **Note**
>
> Wenn Ihr Toolkit weitere Felder hinzufügt (beispielsweise fügt Flash den Dateinamen hinzu), müssen Sie diese dem Richtliniendokument hinzufügen. Wenn Sie diese Funktionalität steuern können, setzen Sie dem Feld `x-ignore-` als Präfix voran, sodass Amazon S3 die Funktion ignoriert, ohne dass sich dies auf zukünftige Versionen dieser Funktion auswirkt.

Übereinstimmung von Bedingungen

Die folgende Tabelle beschreibt die Übereinstimmungstypen von Bedingungen. Obwohl Sie für jedes Formularfeld, das Sie in dem Formular angeben, eine Bedingung angeben müssen, können Sie auch komplexere Übereinstimmungskriterien erstellen, indem Sie für ein Formularfeld mehrere Bedingungen angeben.

Bedingung	Beschreibung
Genaue Übereinstimmung	Hierbei müssen die Felder spezifische Werte enthalten. In diesem Beispiel ist angegeben, dass die ACL auf „public-read“ gesetzt werden muss: `{"acl": "public-read" }` Dieses Beispiel zeigt eine alternative Möglichkeit, um anzuzeigen, dass die ACL auf „public-read“ gesetzt werden muss:

Bedingung	Beschreibung
	`[ "eq", "$acl", "public-read" ]`
Beginnt mit	Verwenden Sie diese Bedingung, wenn der Wert mit einem bestimmten Wert beginnen muss. In diesem Beispiel wird angegeben, dass der Schlüssel mit „user/betty“ beginnen muss: `["starts-with", "$key", "user/betty/"]`
Übereinstimmung mit beliebigem Inhalt	Verwenden Sie „starts-with“ mit einem leeren Wert, um die Richtlinie so zu konfigurieren, dass in einem Feld beliebiger Inhalt zulässig ist. Dieses Beispiel lässt jeden Wert für success_action_redirect zu: `["starts-with", "$success_action_redirect", ""]`
Angabe von Bereichen	Wenn Felder Bereiche akzeptieren, trennen Sie die Ober- und die Untergrenze durch ein Komma voneinander ab. In diesem Beispiel sind Dateigrößen von 1 bis 10 Megabyte erlaubt: `["content-length-range", 1048579, 10485760]`

Escape-Zeichen

Die folgende Tabelle beschreibt Zeichen, für die in einem Richtliniendokument ein Escape-Zeichen verwendet werden muss.

Escape-Sequenz	Beschreibung
\\	Umgekehrter Schrägstrich
\$	Dollarzeichen
\b	Backspace
\f	Seitenvorschub
\n	Neue Zeile
\r	Zeilenumschaltung
\t	Horizontaler Tabulator
\V	Vertikaler Tabulator
\u*xxxx*	Alle Unicode-Zeichen

Erstellen einer Signatur

Schritt	Beschreibung
1	Kodieren Sie die Richtlinie mit UTF-8.

Schritt	Beschreibung
2	Kodieren Sie diese UTF-8-Bytes mit Base64.
3	Signieren Sie die Richtlinie mit Ihrem geheimen Zugriffsschlüssel unter Verwendung von HMAC SHA-1.
4	Kodieren Sie die SHA-1-Signatur mit Base64.

Allgemeine Informationen zur Authentifizierung finden Sie unter Verwendung des Auth-Zugriffs.

Umleitung

In diesem Abschnitt wird beschrieben, wie Sie mit Umleitungen umgehen.

Allgemeine Umleitung

Beim Abschluss der POST-Anforderung wird der Benutzer zu dem Ort umgeleitet, den Sie im Feld `success_action_redirect` angeben. Wenn Amazon S3 die URL nicht interpretieren kann, wird das Feld `success_action_redirect` ignoriert.

Wenn `success_action_redirect` nicht angegeben ist, gibt Amazon S3 den leeren Dokumententyp zurück, der im Feld `success_action_status` angegeben ist.

Wenn die POST-Anforderung fehlschlägt, zeigt Amazon S3 einen Fehler an und führt keine Umleitung durch.

Umleitung vor dem Upload

Wenn Ihr Bucket mit <CreateBucketConfiguration> erstellt wurde, benötigen Ihre Endbenutzer möglicherweise eine Umleitung. Wenn dies der Fall ist, kann es sein, das einige Browser mit der Umleitung nicht korrekt umgehen. Dies kommt relativ selten vor, meistens dann, wenn der Bucket gerade eben erstellt wurde.

Upload-Beispiele (AWS Signature Version 2)

Themen

- Datei-Upload (p. 841)
- Textbereich-Upload (p. 844)

> **Note**
>
> Die in diesem Abschnitt besprochene Anfrageauthentifizierung basiert auf AWS Signature Version 2, einem Protokoll für die Authentifizierung eingehender API-Anfragen an AWS-Services. Amazon S3 unterstützt jetzt Signature Version 4, ein Protokoll für die Authentifizierung eingehender API-Anfragen an AWS-Services, in allen AWS-Regionen. Derzeit unterstützen vor dem 30. Januar 2014 erstellte AWS-Regionen weiterhin das vorherige Protokoll, Signature Version 2. Alle nach dem 30. Januar 2014 neu erstellten Regionen unterstützen nur Signature Version 4, weshalb alle Anfragen an diese Regionen Signature Version 4 verwenden müssen. Weitere Informationen finden Sie unter Beispiele: Browserbasierter Upload mit HTTP POST (und AWS Signature Version 4) im Amazon Simple Storage Service API Reference.

Datei-Upload

Dieses Beispiel zeigt den vollständigen Vorgang der Erstellung einer Richtlinie und eines Formulars für den Upload eines Dateianhangs.

Erstellung von Richtlinie und Formular

Die folgende Richtlinie unterstützt Uploads zu Amazon S3 für den Bucket „awsexamplebucket1“.

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html"},
    ["starts-with", "$Content-Type", "image/"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Für diese Richtlinie ist Folgendes erforderlich:

- Der Upload muss vor 12:00 Uhr UTC am 1. Dezember 2007 erfolgen.
- Der Inhalt muss zum Bucket „awsexamplebucket1“ hochgeladen werden.
- Der Schlüssel muss mit „user/eric/“ beginnen.
- Die ACL ist auf „public-read“ gesetzt.
- Die URL für success_action_redirect wird auf https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html festgelegt.
- Bei dem Objekt handelt es sich um eine Bilddatei.
- Der Tag x-amz-meta-uuid muss auf 14365123651274 gesetzt sein.
- Der Tag x-amz-meta-tag kann einen beliebigen Wert enthalten.

Nachfolgend sehen Sie eine mit Base64 kodierte Version dieser Richtlinie.

```
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWNrZXQiOiA
```

Erstellen Sie unter Verwendung Ihrer Anmeldedaten eine Signatur, zum Beispiel ist `0RavWzkygo6QX9caELEqKi9kDbU=` die Signatur für das vorangehende Richtliniendokument.

Das folgende Formular unterstützt eine POST-Anforderung an den Bucket „awsexamplebucket1.net“, der diese Richtlinie verwendet.

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
  ...
  <form action="https://awsexamplebucket1.s3.us-west-1.amazonaws.com/" method="post"
 enctype="multipart/form-data">
    Key to upload: <input type="input" name="key" value="user/eric/" /><br />
    <input type="hidden" name="acl" value="public-read" />
    <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html" />
    Content-Type: <input type="input" name="Content-Type" value="image/jpeg" /><br />
    <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
```

```
    Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
    <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
    <input type="hidden" name="Policy" value="POLICY" />
    <input type="hidden" name="Signature" value="SIGNATURE" />
    File: <input type="file" name="file" /> <br />
    <!-- The elements after this will be ignored -->
    <input type="submit" name="submit" value="Upload to Amazon S3" />
  </form>
  ...
</html>
```

Beispielanforderung

Diese Anforderung geht davon aus, dass das hochgeladene Bild 117.108 Bytes groß ist, die Bilddaten nicht eingeschlossen.

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
 Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: 118698

--9431149156168
Content-Disposition: form-data; name="key"

user/eric/MyPicture.jpg
--9431149156168
Content-Disposition: form-data; name="acl"

public-read
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html
--9431149156168
Content-Disposition: form-data; name="Content-Type"

image/jpeg
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

Some,Tag,For,Picture
--9431149156168
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--9431149156168
Content-Disposition: form-data; name="Policy"

eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWNrZXQiOiA
--9431149156168
Content-Disposition: form-data; name="Signature"
```

```
0RavWzkygo6QX9caELEqKi9kDbU=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg

...file content...
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--9431149156168--
```

Beispielantwort

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html?bucket=awsexamplebucket1&key=user/eric/
MyPicture.jpg&etag=&quot;39d459dfbc0faabbb5e179358dfb94c3&quot;
Server: AmazonS3
```

Textbereich-Upload

Themen

- Erstellung von Richtlinie und Formular (p. 844)
- Beispielanforderung (p. 845)
- Beispielantwort (p. 846)

Das folgende Beispiel zeigt den vollständigen Prozess der Erstellung einer Richtlinie und eines Formulars für den Upload eines Textbereiches. Der Upload eines Textbereiches ist nützlich für die Übermittlung von Benutzern erstellter Inhalte, etwa von Blog-Postings.

Erstellung von Richtlinie und Formular

Die folgende Richtlinie unterstützt das Hochladen von Textbereichen zu Amazon S3 für den Bucket „awsexamplebucket1“.

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
new_post.html"},
    ["eq", "$Content-Type", "text/html"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Für diese Richtlinie ist Folgendes erforderlich:

- Der Upload muss vor 12:00 Uhr GMT am 1. Dezember 2007 erfolgen.

- Der Inhalt muss zum Bucket „awsexamplebucket1“ hochgeladen werden.
- Der Schlüssel muss mit „user/eric/“ beginnen.
- Die ACL ist auf „public-read“ gesetzt.
- Die URL für success_action_redirect ist auf „https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html“ festgelegt.
- Bei dem Objekt handelt es sich um HTML-Text.
- Der Tag x-amz-meta-uuid muss auf 14365123651274 gesetzt sein.
- Der Tag x-amz-meta-tag kann einen beliebigen Wert enthalten.

Nachfolgend sehen Sie eine mit Base64 kodierte Version dieser Richtlinie.

```
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXR
pb25zIjogWwogICAgeyJidWNrZXQiOiAiam9obnNtaXRoIn0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiRrZXkiLCAidXNlci9lcmljLy
LAogICAgeyJhY2wiOiAicHVibGljLXJlYWQifSwKICAgIHsic3VjY2Vzc19hY3Rpb25fcmVkaXJlY3QiOiAiaHR0cDovL2pvaG5zbWl
C5zMy5hbWF6b25hd3MuY29tL25ld19wb3N0Lmh0bWwifSwKICAgIFsiZXEiLCAiJENvbnRlbnQtVHlwZSIsICJ0ZXh0L2h0bWwiXSwK
CAgIHsieC1hbXotbWV0YS11dWlkIjogIjE0MzY1MTIzNjUxMjc0In0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiR4LWFtei1tZXRhLXRh
IsICIiXQogIF0KfQo=
```

Erstellen Sie unter Verwendung Ihrer Anmeldedaten eine Signatur. Beispielsweise ist `qA7FWXKq6VvU68lI9KdveT1cWgF=` die Signatur für das vorherige Richtliniendokument.

Das folgende Formular unterstützt eine POST-Anforderung an den Bucket „awsexamplebucket1.net“, der diese Richtlinie verwendet.

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
  ...
  <form action="https://awsexamplebucket1.s3.us-west-1.amazonaws.com/" method="post"
 enctype="multipart/form-data">
    Key to upload: <input type="input" name="key" value="user/eric/" /><br />
    <input type="hidden" name="acl" value="public-read" />
    <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html" />
    <input type="hidden" name="Content-Type" value="text/html" />
    <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
    Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
    <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
    <input type="hidden" name="Policy" value="POLICY" />
    <input type="hidden" name="Signature" value="SIGNATURE" />
    Entry: <textarea name="file" cols="60" rows="10">

Your blog post goes here.

    </textarea><br />
    <!-- The elements after this will be ignored -->
    <input type="submit" name="submit" value="Upload to Amazon S3" />
  </form>
  ...
</html>
```

Beispielanforderung

Diese Anforderung geht davon aus, dass das hochgeladene Bild 117.108 Bytes groß ist, die Bilddaten nicht eingeschlossen.

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
 Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=178521717625888
Content-Length: 118635

-178521717625888
Content-Disposition: form-data; name="key"

ser/eric/NewEntry.html
--178521717625888
Content-Disposition: form-data; name="acl"

public-read
--178521717625888
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html
--178521717625888
Content-Disposition: form-data; name="Content-Type"

text/html
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-tag"

Interesting Post
--178521717625888
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--178521717625888
Content-Disposition: form-data; name="Policy"
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWNrZXQiOiA

--178521717625888
Content-Disposition: form-data; name="Signature"

qA7FWXKq6VvU68lI9KdveT1cWgF=
--178521717625888
Content-Disposition: form-data; name="file"

...content goes here...
--178521717625888
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--178521717625888--
```

Beispielantwort

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
```

```
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html?
bucket=awsexamplebucket1&key=user/eric/NewEntry.html&etag=40c3271af26b7f1672e41b8a274d28d4
Server: AmazonS3
```

POST mit Adobe Flash

In diesem Abschnitt wird die Verwendung von `POST` mit Adobe Flash beschrieben.

Adobe Flash Player-Sicherheit

Standardmäßig verbietet das Sicherheitsmodell von Adobe Flash, dass Adobe Flash Players Netzwerkverbindungen zu Servern außerhalb der Domäne der SWF-Datei herstellen.

Um diesen Standard zu übergehen, müssen Sie eine öffentlich lesbare crossdomain.xml-Datei zu dem Bucket hochladen, der die POST-Uploads annehmen soll. Nachfolgend sehen Sie ein Beispiel für eine solche crossdomain.xml-Datei.

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">
<cross-domain-policy>
<allow-access-from domain="*" secure="false" />
</cross-domain-policy>
```

> **Note**
>
> Weitere Informationen zum Sicherheitsmodell von Adobe Flash finden Sie auf der Adobe-Website. Durch die Hinzufügung der crossdomain.xml-Datei kann jeder Adobe Flash Player eine Verbindung zu der crossdomain.xml-Datei in Ihrem Bucket herstellen; diese Datei gewährt jedoch keinen Zugriff auf den Amazon S3-Bucket selbst.

Überlegungen zu Adobe Flash

Die FileReference-API in Adobe Flash fügt das Formularfeld `Filename` zu der POST-Anforderung hinzu. Wenn Sie Adobe Flash-Anwendungen erstellen, die unter Verwendung der FileReference-API-Aktion Uploads zu Amazon S3 durchführen, fügen Sie die folgende Bedingung in Ihre Richtlinie ein:

```
['starts-with', '$Filename', '']
```

Einige Versionen von Adobe Flash Player können HTTP-Antworten ohne Text nicht korrekt bearbeiten. Um POST so zu konfigurieren, dass eine Antwort ausgegeben wird, die nicht leer ist, legen Sie `success_action_status` auf 201 fest. Amazon S3 gibt anschließend ein XML-Dokument mit dem Statuscode 201 zurück. Informationen zum Inhalt des XML-Dokuments finden Sie unter POST Object. Informationen zu den Formularfeldern finden Sie unter HTML-Formularfelder (p. 834).

Amazon S3-Ressourcen

Die folgende Tabelle enthält verwandte Ressourcen, die für die Arbeit mit diesem Service nützlich sind.

Ressource	Beschreibung
Amazon Simple Storage Service Handbuch Erste Schritte	Das Handbuch Erste Schritte bietet eine schnelle Einführung in den Service anhand eines einfachen Anwendungsfalls.
Amazon Simple Storage Service API Reference	Die API-Referenz beschreibt Amazon S3-Vorgänge im Detail.
Häufig gestellte technische Fragen zu Amazon S3	Die Webseite „Häufig gestellte Fragen” deckt alle wichtigsten Fragen ab, die Entwickler zu diesem Produkt gestellt haben.
AWS Developer Resource Center	Ein zentraler Ausgangspunkt für Dokumentationen, Codebeispiele, Versionshinweise und andere Informationen, die Ihnen helfen, mit AWS innovative Anwendungen zu erstellen.
AWS Management Console	Die Konsole ermöglicht es Ihnen, die meisten Funktionen von Amazon S3 ohne Programmierung auszuführen.
https://forums.aws.amazon.com/	Ein auf der Community basierendes Forum, das für Entwickler eingerichtet wurde, um technische Fragen zu AWS zu klären
AWS Support-Center	Die Startseite für den technischen Support von AWS, einschließlich Zugang zu unseren Entwicklerforen, häufig gestellten technischen Fragen, der Servicestatusseite und dem Premium Support.
AWS Premium Support	Die Hauptwebsite mit Informationen zu AWS Premium Support ist ein persönlicher und reaktionsschneller Supportkanal. Er bietet Ihnen Hilfe beim Konfigurieren und Verwendung von Anwendungen auf AWS Infrastruktur-Services.
Amazon S3-Produktinformationen	Hauptwebsite für Informationen zu Amazon S3
Kontakt	Eine zentrale Seite von AWS zur Kontaktaufnahme bei Anfragen zu Abrechnungen, Konto, Vorfällen und Missbrauch usw.
Nutzungsbedingungen	Nähere Informationen zum Urheberrecht, der Verwendung von Markenzeichen auf amazon.com und weiteren Themen.

SQL-Referenz für Amazon S3 Select und S3 Glacier Select.

In dieser Referenz finden Sie eine Beschreibung der SQL (Structured Query Language)-Elemente, die von Amazon S3 Select und S3 Glacier Select unterstützt werden.

Themen

Der Befehl SELECT

Amazon S3 Select und S3 Glacier Select unterstützen nur den SQL-Befehl `SELECT`. Die folgenden ANSI-Standardklauseln werden für `SELECT` unterstützt:

- `SELECT` table
- `FROM`-Klausel
- `WHERE`-Klausel
- `LIMIT`-Klausel (nur Amazon S3 Select)

> **Note**
>
> Amazon S3 Select- und S3 Glacier Select-Abfragen unterstützen derzeit keine Unterabfragen oder Joins.

SELECT-Liste

Die `SELECT`-Liste nennt die Spalten, Funktionen und Ausdrücke, die die Abfrage zurückgeben soll. Der Liste stellt die Ausgabe der Abfrage dar.

```
SELECT *
SELECT projection [ AS column_alias | column_alias ] [, ...]
```

Bei der ersten Form mit `*` (Sternchen) werden alle Zeilen zurückgegeben, die die Bedingung der `WHERE`-Klausel erfüllen. Bei der zweiten Form wird eine Zeile mit benutzerdefinierten skalaren Ausgabeausdrücken projection für jede Spalte erstellt.

FROM-Klausel

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden Formen der `FROM`-Klausel:

```
FROM table_name
FROM table_name alias
FROM table_name AS alias
```

Wobei `table_name` entweder `S3Object` (für Amazon S3 Select) bzw. `ARCHIVE` oder `OBJECT` (für S3 Glacier Select) ist und auf das abzurufende Archiv verweist. Benutzer traditioneller relationaler Datenbanken können sich dies als Datenbankschema mit mehreren Ansichten einer Tabelle vorstellen.

Gemäß Standard-SQL erstellt die `FROM`-Klausel Zeilen, die in der `WHERE`-Klausel gefiltert und in der Liste `SELECT` projiziert werden.

Im Fall von JSON-Objekten, die in Amazon S3 Select gespeichert sind, können Sie auch die folgenden Formate der `FROM`-Klausel verwenden:

```
FROM S3Object[*].path
FROM S3Object[*].path alias
FROM S3Object[*].path AS alias
```

Unter Verwendung dieses Format der `FROM`-Klausel können Sie aus Arrays oder Objekten innerhalb eines JSON-Objekts auswählen. Sie können `path` unter Verwendung eines der folgenden Formate angeben:

- Nach Name (in einem Objekt): `.name` oder `['name']`
- Nach Index (in einem Array): `[index]`
- Nach Platzhalter (in einem Objekt): `.*`
- Nach Platzhalter (in einem Array): `[*]`

Note

- Das Format der `FROM`-Klausel funktioniert nur mit JSON-Objekten.
- Platzhalter übermitteln stets mindestens einen Datensatz. Wenn es keinen übereinstimmenden Datensatz gibt, übermittelt Amazon S3 Select den Wert `MISSING`. Während der Ausgabeserialisierung (nachdem die Abfrage abgeschlossen wurde), ersetzt Amazon S3 Select `MISSING`-Werte durch leere Datensätze.
- Aggregierte Funktionen (`AVG`, `COUNT`, `MAX`, `MIN` und `SUM`) überspringen `MISSING`-Werte.
- Wenn Sie bei Verwendung eines Platzhalters keinen Alias angeben, können Sie auf die Zeile verweisen, die das letzte Element im Pfad verwendet. Sie könnten beispielsweise alle Preise aus einer Liste von Büchern unter Verwendung der Abfrage `SELECT price FROM S3Object[*].books[*].price` auswählen. Wenn der Pfad mit einem Platzhalter anstelle eines Namens endet, können Sie den Wert `_1` verwenden, um auf die Zeile zu verweisen. Beispielsweise könnten Sie anstelle von `SELECT price FROM S3Object[*].books[*].price` die Abfrage `SELECT _1.price FROM S3Object[*].books[*]` verwenden.
- Amazon S3 Select behandelt ein JSON-Dokument stets als Array aus Werten auf Root-Ebene. Daher muss die `FROM`-Klausel mit `S3Object[*]` beginnen, auch wenn das von Ihnen abgefragte JSON-Objekt nur ein Root-Element hat. Aus Kompatibilitätsgründen ermöglicht Amazon S3 Select Ihnen das Auslassen des Platzhalters, wenn Sie keinen Pfad einfügen. Daher ist die vollständige Klausel `FROM S3Object` gleichwertig mit `FROM S3Object[*] as S3Object`. Wenn Sie einen Pfad einfügen, müssen Sie auch den Platzhalter verwenden. Daher sind sowohl `FROM S3Object` als auch `FROM S3Object[*].path` gültige Klauseln, nicht aber `FROM S3Object.path`.

Example

Beispiele:

Beispiel 1

Dieses Beispiel zeigt Ergebnisse unter Verwendung des folgenden Datensatzes und der folgenden Abfrage:

```
{ "Rules": [ {"id": "1"}, {"expr": "y > x"}, {"id": "2", "expr": "z = DEBUG"} ]}
{ "created": "June 27", "modified": "July 6" }
```

```
SELECT id FROM S3Object[*].Rules[*].id
```

```
{"id":"1"}
{}
{"id":"2"}
{}
```

Amazon S3 Select produziert die einzelnen Ergebnisse aus den folgenden Gründen:

- {"id":"id-1"} — S3Object[0].Rules[0].id produzierte eine Übereinstimmung.
- {} — S3Object[0].Rules[1].id produzierte keine Übereinstimmung mit einem Datensatz. Daher übermittelte Amazon S3 Select `MISSING`, was anschließend während der Ausgabeserialisierung in einen leeren Daten geändert und zurückgegeben wurde.
- {"id":"id-2"} — S3Object[0].Rules[2].id produzierte eine Übereinstimmung.
- {} — S3Object[1] produzierte keine Übereinstimmung für `Rules`. Daher übermittelte Amazon S3 Select `MISSING`, was anschließend während der Ausgabeserialisierung in einen leeren Daten geändert und zurückgegeben wurde.

Wenn Sie nicht möchten, dass Amazon S3 Select leere Datensätze zurückgibt, wenn keine Übereinstimmung gefunden wird, können Sie einen Test auf den `MISSING` ausführen. Die folgende Abfrage gibt dieselben Ergebnisse wie die vorherige Abfrage zurück, jedoch mit Auslassung der leeren Werte:

```
SELECT id FROM S3Object[*].Rules[*].id WHERE id IS NOT MISSING
```

```
{"id":"1"}
{"id":"2"}
```

Beispiel 2

Dieses Beispiel zeigt Ergebnisse unter Verwendung des folgenden Datensatzes und der folgenden Abfragen:

```
{ "created": "936864000", "dir_name": "important_docs", "files": [ { "name": "." },
 { "name": ".." }, { "name": ".aws" }, { "name": "downloads" } ], "owner": "AWS S3" }
{ "created": "936864000", "dir_name": "other_docs", "files": [ { "name": "." }, { "name":
 ".." }, { "name": "my stuff" }, { "name": "backup" } ], "owner": "User" }
```

```
SELECT d.dir_name, d.files FROM S3Object[*] d
```

```
{"dir_name":"important_docs","files":[{"name":"."},{"name":".."},{"name":".aws"},
{"name":"downloads"}]}
{"dir_name":"other_docs","files":[{"name":"."},{"name":".."},{"name":"my stuff"},
{"name":"backup"}]}
```

```
SELECT _1.dir_name, _1.owner FROM S3Object[*]
```

```
{"dir_name":"important_docs","owner":"AWS S3"}
```

```
{"dir_name":"other_docs","owner":"User"}
```

WHERE-Klausel

Die `WHERE`-Klausel hat die folgende Syntax:

```
WHERE condition
```

Die `WHERE`-Klausel filtert Zeilen basierend auf der Bedingung. Eine Bedingung ist ein Ausdruck, der als Ergebnis einen booleschen Wert zurückgibt. Nur Zeilen, deren Bedingung als `TRUE` ausgewertet wird, werden als Ergebnis zurückgegeben.

LIMIT-Klausel (nur Amazon S3 Select)

Die `LIMIT`-Klausel hat die folgende Syntax:

```
LIMIT number
```

Die `LIMIT`-Klausel begrenzt die Anzahl der von der Abfrage zurückgegebenen Datensätze basierend auf dem number-Wert.

> Note
>
> Die `LIMIT`-Klausel wird von S3 Glacier Select nicht unterstützt.

Attributzugriff

Die Klauseln `SELECT` und `WHERE` können mithilfe einer der nachfolgend genannten Methoden auf einen Datensatz verweisen. Das ist abhängig davon, ob die abzufragende Datei im CSV- oder JSON-Format vorliegt.

CSV

- Spaltennummern – Sie können auf die Nte-Spalte einer Zeile mit dem Spaltennamen `_N` verweisen, wobei "N" die Spaltenposition angibt. Der Positionszähler beginnt mit 1. Die erste Spalte heißt demnach `_1`, die zweite Spalte heißt `_2`.

  Sie können mit `_N` oder `alias._N` auf eine Spalte verweisen. Beispielsweise sind sowohl `_2` und `myAlias._2` gültige Möglichkeiten, um auf eine Spalte in der `SELECT`-Liste und der `WHERE`-Klausel zu verweisen.
- Spalten-Header – Bei Objekten im CSV-Format, die eine Kopfzeile enthalten, sind die Header für die `SELECT`-Liste und die `WHERE`-Klausel verfügbar. Besonders in `SELECT`- und `WHERE`-Klauselausdrücken in traditionellem SQL können Sie mit `alias.column_name` oder `column_name` auf die Spalten verweisen.

JSON (nur Amazon S3 Select)

- Dokument – Sie können mit `alias.name` auf JSON-Dokumentfelder zugreifen. Auch der Zugriff auf verschachtelte Felder ist möglich, z. B. `alias.name1.name2.name3`.
- Liste – Sie können über nullbasierte Indizes mit dem Operator `[]` auf Elemente in einer JSON-Liste zugreifen. Beispielsweise lässt sich das zweite Element einer Liste mithilfe von `alias[1]` aufrufen. Der Zugriff auf Listenelemente kann mit Feldern kombiniert werden, z. B. `alias.name1.name2[1].name3`.
- Beispiele: Betrachten Sie dieses JSON-Objekt als Beispieldatensatz:

```
{"name": "Susan Smith",
"org": "engineering",
"projects":
    [
     {"project_name":"project1", "completed":false},
     {"project_name":"project2", "completed":true}
    ]
}
```

Beispiel 1

Die folgende Abfrage gibt die folgenden Ergebnisse zurück:

```
Select s.name from S3Object s
```

```
{"name":"Susan Smith"}
```

Beispiel 2

Die folgende Abfrage gibt die folgenden Ergebnisse zurück:

```
Select s.projects[0].project_name from S3Object s
```

```
{"project_name":"project1"}
```

Groß-/Kleinschreibung bei Header-/Attributnamen

In Amazon S3 Select und S3 Glacier Select können Sie mithilfe von doppelten Anführungszeichen angeben, dass bei Spalten-Headern (für CSV-Objekte) und Attributen (für JSON-Objekte) die Groß-/Kleinschreibung beachtet werden muss. Ohne doppelte Anführungszeichen muss die Groß-/Kleinschreibung bei Objekt-Headern/-attributen nicht berücksichtigt werden. Im Falle einer Zweideutigkeit wird ein Fehler ausgegeben.

In den folgenden Beispielen handelt es sich entweder um 1) Amazon S3- oder S3 Glacier-Objekte im CSV-Format mit spezifizierten Spalten-Headern und mit auf "Use" (Verwenden) gesetztem `FileHeaderInfo`-Wert für die Abfrageanforderung oder um 2) Amazon S3-Objekte im JSON-Format mit spezifizierten Attributen.

Beispiel 1: Das abzufragende Objekte hat den Header/das Attribut "NAME".

- Mit folgendem Ausdruck werden erfolgreich Werte vom Objekt zurückgegeben (keine Anführungszeichen: Groß-/Kleinschreibung muss nicht beachtet werden):

```
SELECT s.name from S3Object s
```

- Der folgende Ausdruck führt zum 400-Fehler `MissingHeaderName` (Anführungszeichen: Groß-/Kleinschreibung muss beachtet werden):

```
SELECT s."name" from S3Object s
```

Beispiel 2: Das abzufragende Amazon S3-Objekt hat den Header/das Attribut "NAME" sowie den Header/das Attribut "name".

- Der folgende Ausdruck führt zum 400-Fehler `AmbiguousFieldName` (keine Anführungszeichen: Groß-/Kleinschreibung muss nicht beachtet werden, aber es gibt zwei Übereinstimmungen):

```
SELECT s.name from S3Object s
```

- Mit folgendem Ausdruck werden erfolgreich Werte vom Objekt zurückgegeben (Anführungszeichen: Groß-/Kleinschreibung muss beachtet werden und damit wird die Zweideutigkeit aufgelöst):

```
SELECT s."NAME" from S3Object s
```

Verwenden von reservierten Schlüsselwörtern als benutzerdefinierte Begriffe

Amazon S3 Select und S3 Glacier Select verfügen über eine Reihe reservierter Schlüsselwörter, die zur Ausführung der SQL-Ausdrücke benötigt werden, mit denen Objektinhalte abgefragt werden. Zu den reservierten Schlüsselwörtern zählen Funktionsnamen, Datentypen, Operatoren etc. Gelegentlich können sich benutzerdefinierte Begriffe – wie z. B. Spalten-Header (bei CSV-Dateien) oder Attribute (bei JSON-Objekten) – mit einem reservierten Schlüsselwort überschneiden. In diesem Fall geben Sie mithilfe von doppelten Anführungszeichen an, dass Sie absichtlich einen benutzerdefinierten Begriff verwenden, der mit einem reservierten Schlüsselwort übereinstimmt. Andernfalls wird ein 400-Parse-Fehler ausgegeben.

Eine vollständige Liste der reservierten Schlüsselwörter finden Sie unter Reservierte Schlüsselwörter (p. 857).

Im folgenden Beispiel handelt es sich entweder um 1) Amazon S3- oder S3 Glacier-Objekte im CSV-Format mit spezifizierten Spalten-Headern und mit auf "Use" (Verwenden) gesetztem `FileHeaderInfo`-Wert für die Abfrageanforderung oder um 2) Amazon S3-Objekte im JSON-Format mit spezifizierten Attributen.

Beispiel: Das abzufragende Objekt hat den Header/das Attribut "CAST" und das ist ein reserviertes Schlüsselwort.

- Mit folgendem Ausdruck werden erfolgreich Werte vom Objekt zurückgegeben (Anführungszeichen: benutzerdefinierter Header/Attribut):

```
SELECT s."CAST" from S3Object s
```

- Der folgende Ausdruck führt zu einem 400-Parse-Fehler (keine Anführungszeichen: Übereinstimmung mit reserviertem Schlüsselwort):

```
SELECT s.CAST from S3Object s
```

Skalare Ausdrücke

Innerhalb der `WHERE`-Klausel und der `SELECT`-Liste können Sie skalare SQL-Ausdrücke verwenden. Dies sind Ausdrücke, die skalare Werte zurückgeben. Sie haben das folgende Format:

- Literal

  Ein SQL-Literal
- column_reference

  Ein Verweis auf eine Spalte im Format spaltenname oder alias.spaltenname.

- unary_op Ausdruck

  Wobei unary_op ein unärer SQL-Operator ist
- Ausdruck binary_op Ausdruck

  Wobei binary_op ein binärer SQL-Operator ist
- func_name

  Wobei func_name der Name einer aufzurufenden skalaren Funktion ist
- Ausdruck `[ NOT ] BETWEEN` Ausdruck `AND` Ausdruck
- Ausdruck `LIKE` Ausdruck [`ESCAPE` Ausdruck]

Datentypen

Amazon S3 Select und S3 Glacier Select unterstützen mehrere primitive Datentypen.

Datentypkonvertierungen

Allgemein sollte die Funktion `CAST` verwendet werden, wenn sie definiert ist. Falls `CAST` nicht definiert ist, werden alle Eingabedaten als Zeichenfolge betrachtet. Sie müssen bei Bedarf in geeignete Datentypen umgewandelt werden.

Weitere Informationen zur Funktion `CAST` finden Sie unter CAST (p. 863).

Unterstützte Datentypen

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden primitiven Datentypen.

Name	Beschreibung	Beispiele
bool	FALSE oder TRUE	`FALSE`
int, integer	8-Byte-Ganzzahl im Bereich -9.223.372.036.854.775.808 bis 9.223.372.036.854.775.807.	`100000`
string	UTF-8-kodierte Zeichenfolge mit variabler Länge. Das Standardlimit ist ein Zeichen. Das maximale Zeichenlimit beträgt 2.147.483.647.	`'xyz'`
float	8-Byte-Gleitkommazahl	`CAST(0.456 AS FLOAT)`
decimal, numeric	Zahl mit der Basis 10 mit einer maximalen Genauigkeit von 38 (d. h. maximale Anzahl signifikanter Ziffern) und einer Größe zwischen -2^{31} und $2^{31}-1$ (d. h. der Exponent der Basis 10).	`123.456`
timestamp	Zeitstempel stellen einen bestimmten Zeitpunkt dar, enthalten immer einen lokalen Versatz und können beliebig genau sein. Im Textformat befolgen Zeitstempel den W3C-Hinweis zu Datums- und Zeitformaten, müssen jedoch mit dem Buchstaben "T" enden, wenn die Präzision nicht mindestens ein ganzer Tag ist. Sekundenbruchteile mit mindestens einer Stelle bis zu beliebig vielen Stellen sind zulässig. Der lokale Zeitversatz kann entweder	`CAST('2007-04-05T14:30Z' AS TIMESTAMP)`

Name	Beschreibung	Beispiele
	als Versatz im Format Stunde:Minute von UTC-Zeit oder als Buchstabe "Z" zur Angabe einer lokalen UTC-Zeit angegeben werden. Er ist für Zeitstempel mit Uhrzeit erforderlich, für reine Datumswerte jedoch nicht zulässig.	

Operatoren

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden Operatoren.

Logische Operatoren

- `AND`
- `NOT`
- `OR`

Vergleichsoperatoren

- `<`
- `>`
- `<=`
- `>=`
- `=`
- `<>`
- `!=`
- `BETWEEN`
- `IN` – Beispiel: `IN ('a', 'b', 'c')`

Mustervergleichsoperatoren

- `LIKE`
- _ (Vergleicht ein beliebiges Zeichen)
- % (Vergleicht eine beliebige Zeichenreihenfolge)

Unäre Operatoren

- IS NULL
- IS NOT NULL

Mathematische Operatoren

Es werden Addition, Subtraktion, Multiplikation, Division und Modulo unterstützt.

- `+`

- –
- *
- %

Rangfolge der Operatoren

Die folgende Tabelle zeigt die Rangfolge der Operatoren in absteigender Reihenfolge.

Operator/ Element	Assoziativität	Erforderlich
–	rechts	unär minus
*, /, %	links	Multiplikation, Division und Modulo
+, -	links	Addition und Subtraktion
IN		Mitgliedschaft festlegen
BETWEEN		in Bereich enthalten
LIKE		Zeichenfolgenübereinstimmung
<>		kleiner als, größer als
=	rechts	Gleichheit, Zuweisung
NOT	rechts	logische Negation
AND	links	logische Verbindung
ODER	links	logische Disjunktion

Reservierte Schlüsselwörter

Nachstehend finden Sie eine Liste der reservierten Schlüsselwörter für Amazon S3 Select und S3 Glacier Select. Dazu zählen Funktionsnamen, Datentypen, Operatoren etc., die zur Ausführung der SQL-Ausdrücke benötigt werden, mit denen Objektinhalte abgefragt werden.

```
absolute
action
add
all
allocate
alter
```

```
and
any
are
as
asc
assertion
at
authorization
avg
bag
begin
between
bit
bit_length
blob
bool
boolean
both
by
cascade
cascaded
case
cast
catalog
char
char_length
character
character_length
check
clob
close
coalesce
collate
collation
column
commit
connect
connection
constraint
constraints
continue
convert
corresponding
count
create
cross
current
current_date
current_time
current_timestamp
current_user
cursor
date
day
deallocate
dec
decimal
declare
default
deferrable
deferred
delete
desc
describe
descriptor
diagnostics
```

```
disconnect
distinct
domain
double
drop
else
end
end-exec
escape
except
exception
exec
execute
exists
external
extract
false
fetch
first
float
for
foreign
found
from
full
get
global
go
goto
grant
group
having
hour
identity
immediate
in
indicator
initially
inner
input
insensitive
insert
int
integer
intersect
interval
into
is
isolation
join
key
language
last
leading
left
level
like
limit
list
local
lower
match
max
min
minute
missing
```

```
module
month
names
national
natural
nchar
next
no
not
null
nullif
numeric
octet_length
of
on
only
open
option
or
order
outer
output
overlaps
pad
partial
pivot
position
precision
prepare
preserve
primary
prior
privileges
procedure
public
read
real
references
relative
restrict
revoke
right
rollback
rows
schema
scroll
second
section
select
session
session_user
set
sexp
size
smallint
some
space
sql
sqlcode
sqlerror
sqlstate
string
struct
substring
sum
symbol
```

```
system_user
table
temporary
then
time
timestamp
timezone_hour
timezone_minute
to
trailing
transaction
translate
translation
trim
true
tuple
union
unique
unknown
unpivot
update
upper
usage
user
using
value
values
varchar
varying
view
when
whenever
where
with
work
write
year
zone
```

SQL-Funktionen

Amazon S3 Select und S3 Glacier Select unterstützen mehrere SQL-Funktionen.

Themen

Aggregationsfunktionen (nur Amazon S3 Select)

Amazon S3 Select unterstützt die folgenden Aggregationsfunktionen.

> **Note**
>
> S3 Glacier Select unterstützt keine Aggregationsfunktionen.

Funktion	Argumenttyp	Typ der Rückgabe
`AVG(expression)`	INT, FLOAT, DECIMAL	DECIMAL für ein INT-Argument, FLOAT für ein Gleitkomma-Argument; andernfalls gleich dem Argumentdatentyp.
`COUNT`	–	INT
`MAX(expression)`	INT, DECIMAL	Entspricht dem Argumenttyp.
`MIN(expression)`	INT, DECIMAL	Entspricht dem Argumenttyp.
`SUM(expression)`	INT, FLOAT, DOUBLE, DECIMAL	INT für ein INT-Argument, FLOAT für ein Gleitkomma-Argument; andernfalls gleich dem Argumentdatentyp.

Konditionale Funktionen

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden konditionalen Funktionen.

Themen

COALESCE

Wertet die Argumente nacheinander aus und gibt das erste unbekannte zurück, das heißt, das erste, das nicht null oder nicht fehlend ist. Null und fehlend werden von der Funktion nicht übernommen.

Syntax

```
COALESCE ( expression, expression, ... )
```

Parameter

Ausdruck

Der Zielausdruck, der von der Funktion verwendet wird.

Beispiele

```
COALESCE(1)                -- 1
COALESCE(null)             -- null
COALESCE(null, null)       -- null
COALESCE(missing)          -- null
COALESCE(missing, missing) -- null
COALESCE(1, null)          -- 1
COALESCE(null, null, 1)    -- 1
COALESCE(null, 'string')   -- 'string'
COALESCE(missing, 1)       -- 1
```

NULLIF

Gibt bei zwei Ausdrücken, die das gleiche Auswertungsergebnis haben, NULL zurück. Andernfalls wird das Auswertungsergebnis für den ersten Ausdruck zurückgegeben.

Syntax

```
NULLIF ( expression1, expression2 )
```

Parameter

expression1, expression2

Die Zielausdrücke, die von der Funktion verwendet werden.

Beispiele

```
NULLIF(1, 1)             -- null
NULLIF(1, 2)             -- 1
NULLIF(1.0, 1)           -- null
NULLIF(1, '1')           -- 1
NULLIF([1], [1])         -- null
NULLIF(1, NULL)          -- 1
NULLIF(NULL, 1)          -- null
NULLIF(null, null)       -- null
NULLIF(missing, null)    -- null
NULLIF(missing, missing) -- null
```

Konvertierungsfunktionen

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden Konvertierungsfunktionen.

Themen

CAST

Die `CAST`-Funktion konvertiert ein Element, z. B. einen Ausdruck zur Auswertung eines einzelnen Werts, von einem Typ in einen anderen.

Syntax

```
CAST ( expression AS data_type )
```

Parameter

Ausdruck

Eine Kombination von Werten, Operatoren und SQL-Funktionen, die zu einem Wert ausgewertet werden können

data_type

Der Zieldatentyp, z. B. `INT`, in den der Ausdruck umgewandelt werden soll Eine Liste der unterstützten Datentypen finden Sie unter Datentypen (p. 855).

Beispiele

```
CAST('2007-04-05T14:30Z' AS TIMESTAMP)
CAST(0.456 AS FLOAT)
```

Datumsfunktionen

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden Datumsfunktionen.

Themen

DATE_ADD

Gibt bei einem Datumsteil, einer Menge und einem Zeitstempel einen aktualisierten Zeitstempel zurück, indem der Datumsteil anhand der Menge modifiziert wird.

Syntax

```
DATE_ADD( date_part, quantity, timestamp )
```

Parameter

date_part

Gibt den zu modifizierenden Teil des Datums an. Dabei kann es sich um einen der folgenden Werte handeln:

- Jahr
- Monat
- Tag
- Stunde
- Minute
- Sekunde

quantity

Der Wert, der auf den aktualisierten Zeitstempel anzuwenden ist. Positive "quantity"-Werte werden zum "date_part"-Wert des Zeitstempels addiert, negative Werte werden subtrahiert.

timestamp

Der Zielzeitstempel, der von der Funktion verwendet wird.

Beispiele

```
DATE_ADD(year, 5, `2010-01-01T`)                -- 2015-01-01 (equivalent to 2015-01-01T)
DATE_ADD(month, 1, `2010T`)                     -- 2010-02T (result will add precision as
 necessary)
DATE_ADD(month, 13, `2010T`)                    -- 2011-02T
DATE_ADD(day, -1, `2017-01-10T`)                -- 2017-01-09 (equivalent to 2017-01-09T)
DATE_ADD(hour, 1, `2017T`)                      -- 2017-01-01T01:00-00:00
DATE_ADD(hour, 1, `2017-01-02T03:04Z`)          -- 2017-01-02T04:04Z
DATE_ADD(minute, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:05:05.006Z
DATE_ADD(second, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:04:06.006Z
```

DATE_DIFF

Gibt bei einem Datumsteil und zwei gültigen Zeitstempeln die Differenz in Datumsteilen an. Sofern der `date_part`-Wert von `timestamp1` größer ist als der `date_part`-Wert von `timestamp2`, wird eine negative Ganzzahl zurückgegeben. Wenn der `date_part`-Wert von `timestamp1` kleiner ist als der `date_part`-Wert von `timestamp2`, wird eine positive Ganzzahl zurückgegeben.

Syntax

```
DATE_DIFF( date_part, timestamp1, timestamp2 )
```

Parameter

date_part

Gibt den zu vergleichenden Teil der Zeitstempel an. Die Definition von `date_part` finden Sie unter DATE_ADD (p. 864).

timestamp1

Der erste Zeitstempel für den Vergleich.

timestamp2

Der zweite Zeitstempel für den Vergleich.

Beispiele

```
DATE_DIFF(year, `2010-01-01T`, `2011-01-01T`)           -- 1
DATE_DIFF(year, `2010T`, `2010-05T`)                    -- 4 (2010T is equivalent to
 2010-01-01T00:00:00.000Z)
DATE_DIFF(month, `2010T`, `2011T`)                      -- 12
DATE_DIFF(month, `2011T`, `2010T`)                      -- -12
DATE_DIFF(day, `2010-01-01T23:00T`, `2010-01-02T01:00T`) -- 0 (need to be at least 24h
 apart to be 1 day apart)
```

EXTRACT

Gibt bei einem Datumsteil und einem Zeitstempel den Datumsteilwert des Zeitstempels zurück.

Syntax

```
EXTRACT( date_part FROM timestamp )
```

Parameter

date_part

Gibt den zu extrahierenden Teil der Zeitstempel an. Dabei kann es sich um einen der folgenden Werte handeln:

- Jahr
- Monat
- Tag
- Stunde
- Minute
- Sekunde
- Zeitzone_Stunde
- Zeitzone_Minute

timestamp

Der Zielzeitstempel, der von der Funktion verwendet wird.

Beispiele

```
EXTRACT(YEAR FROM `2010-01-01T`)                            -- 2010
EXTRACT(MONTH FROM `2010T`)                                 -- 1 (equivalent to
 2010-01-01T00:00:00.000Z)
EXTRACT(MONTH FROM `2010-10T`)                              -- 10
EXTRACT(HOUR FROM `2017-01-02T03:04:05+07:08`)              -- 3
EXTRACT(MINUTE FROM `2017-01-02T03:04:05+07:08`)            -- 4
EXTRACT(TIMEZONE_HOUR FROM `2017-01-02T03:04:05+07:08`)     -- 7
EXTRACT(TIMEZONE_MINUTE FROM `2017-01-02T03:04:05+07:08`)   -- 8
```

TO_STRING

Gibt bei einem Zeitstempel und einem Formatmuster eine Zeichenfolgendarstellung des Zeitstempels im angegebenen Format zurück.

Syntax

```
TO_STRING ( timestamp time_format_pattern )
```

Parameter

timestamp

Der Zielzeitstempel, der von der Funktion verwendet wird.

time_format_pattern

Eine Zeichenfolge mit folgenden speziellen Zeichenbedeutungen:

Format	Beispiel	Beschreibung
`yy`	`69`	Jahreszahl mit 2 Ziffern
`y`	`1969`	Jahreszahl mit 4 Ziffern
`yyyy`	`1969`	Jahreszahl mit 4 Ziffern, mit Nullen aufgefüllt
`M`	`1`	Monatsname
`MM`	`01`	Monatsname, mit Nullen aufgefüllt
`MMM`	`Jan`	Abkürzung des Monatsnamens
`MMMM`	`January`	Vollständiger Monatsnamen
`MMMMM`	`J`	Erster Buchstabe des Monatsnamens (HINWEIS: kann nicht mit der Funktion to_timestamp verwendet werden)
`d`	`2`	Monatstag (1-31)
`dd`	`02`	Monatstag, mit Nullen aufgefüllt (01-31)
`a`	`AM`	AM oder PM
`h`	`3`	Stunde (1-12)
`hh`	`03`	Stunde, mit Nullen aufgefüllt (01-12)
`H`	`3`	Stunde (0-23)
`HH`	`03`	Stunde, mit Nullen aufgefüllt (00-23)
`m`	`4`	Minute (0-59)
`mm`	`04`	Minute, mit Nullen aufgefüllt (00-59)

Format	Beispiel	Beschreibung
`s`	`5`	Sekunde (0-59)
`ss`	`05`	Sekunde, mit Nullen aufgefüllt (00-59)
`S`	`0`	Sekundenbruchteil (Genauigkeit: 0,1, Bereich: 0,0-0,9)
`SS`	`6`	Sekundenbruchteil (Genauigkeit: 0,01, Bereich: 0,0-0,99)
`SSS`	`60`	Sekundenbruchteil (Genauigkeit: 0,001, Bereich: 0,0-0,999)
…	…	…
`SSSSSSSSS`	`60000000`	Sekundenbruchteil (max. Genauigkeit: 1 Nanosekunde, Bereich: 0,0-0,999999999)
`n`	`60000000`	Nanosekunde
`X`	`+07 or Z`	Offset in Stunden oder "Z" bei Offset = 0
`XX or XXXX`	`+0700 or Z`	Offset in Stunden und Minuten oder "Z" bei Offset = 0
`XXX or XXXXX`	`+07:00 or Z`	Offset in Stunden und Minuten oder "Z" bei Offset = 0
`x`	`7`	Offset in Stunden
`xx or xxxx`	`700`	Offset in Stunden und Minuten

Format	Beispiel	Beschreibung
`xxx or xxxxx`	`+07:00`	Offset in Stunden und Minuten

Beispiele

```
TO_STRING(`1969-07-20T20:18Z`,  'MMMM d, y')                    -- "July 20, 1969"
TO_STRING(`1969-07-20T20:18Z`, 'MMM d, yyyy')                   -- "Jul 20, 1969"
TO_STRING(`1969-07-20T20:18Z`, 'M-d-yy')                        -- "7-20-69"
TO_STRING(`1969-07-20T20:18Z`, 'MM-d-y')                        -- "07-20-1969"
TO_STRING(`1969-07-20T20:18Z`, 'MMMM d, y h:m a')               -- "July 20, 1969 8:18 PM"
TO_STRING(`1969-07-20T20:18Z`, 'y-MM-dd''T''H:m:ssX')           -- "1969-07-20T20:18:00Z"
TO_STRING(`1969-07-20T20:18+08:00Z`, 'y-MM-dd''T''H:m:ssX')     -- "1969-07-20T20:18:00Z"
TO_STRING(`1969-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXX')   --
 "1969-07-20T20:18:00+0800"
TO_STRING(`1969-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXXX')  --
 "1969-07-20T20:18:00+08:00"
```

TO_TIMESTAMP

Wandelt eine Zeichenfolge in einen Zeitstempel um. Das ist der umgekehrte Vorgang von TO_STRING.

Syntax

```
TO_TIMESTAMP ( string )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
TO_TIMESTAMP('2007T')                          -- `2007T`
TO_TIMESTAMP('2007-02-23T12:14:33.079-08:00') -- `2007-02-23T12:14:33.079-08:00`
```

UTCNOW

Gibt die aktuelle Zeit in UTC als Zeitstempel zurück.

Syntax

```
UTCNOW()
```

Parameter

Keine

Beispiele

```
UTCNOW() -- 2017-10-13T16:02:11.123Z
```

Funktionen für Zeichenfolgen

Amazon S3 Select und S3 Glacier Select unterstützen die folgenden Zeichenfolgenfunktionen.

Themen

CHAR_LENGTH, CHARACTER_LENGTH

Zählt die Anzahl der Zeichen in der angegebenen Zeichenfolge.

> **Note**
>
> `CHAR_LENGTH` und `CHARACTER_LENGTH` sind Synonyme.

Syntax

```
CHAR_LENGTH ( string )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
CHAR_LENGTH('')          -- 0
CHAR_LENGTH('abcdefg')   -- 7
```

LOWER

Wandelt alle Großbuchstaben einer Zeichenfolge in Kleinbuchstaben um. Alle Zeichen, die keine Großbuchstaben sind, bleiben unverändert.

Syntax

```
LOWER ( string )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
LOWER('AbCdEfG!@#$') -- 'abcdefg!@#$'
```

SUBSTRING

Gibt bei einer Zeichenfolge, einem Startindex und (optional) einer Länge die Teilzeichenfolge vom Startindex bis zum Ende der Zeichenfolge oder bis zur angegebenen Länge zurück.

> Note
>
> Das erste Zeichen der Eingabezeichenfolge hat den Index 1. Wenn `start` den Wert < 1, aufweist, wird er auf 1 gesetzt.

Syntax

```
SUBSTRING( string FROM start [ FOR length ] )
```

Parameter

string

> Die Zielzeichenfolge, die von der Funktion verwendet wird.

start

> Die Startposition der Zeichenfolge.

length

> Die Länge der zurückzugebenden Teilzeichenfolge. Falls nicht angegeben, wird bis zum Ende der Zeichenfolge fortgefahren.

Beispiele

```
SUBSTRING("123456789", 0)      -- "123456789"
SUBSTRING("123456789", 1)      -- "123456789"
SUBSTRING("123456789", 2)      -- "23456789"
SUBSTRING("123456789", -4)     -- "123456789"
SUBSTRING("123456789", 0, 999) -- "123456789"
SUBSTRING("123456789", 1, 5)   -- "12345"
```

TRIM

Kürzt vorangestellte oder nachgestellte Zeichen aus einer Zeichenfolge. Standardmäßig wird das Zeichen ' ' entfernt.

Syntax

```
TRIM ( [[LEADING | TRAILING | BOTH remove_chars] FROM] string )
```

Parameter

string

> Die Zielzeichenfolge, die von der Funktion verwendet wird.

LEADING | TRAILING | BOTH

> Gib an, ob vorangestellte oder nachgestellte Zeichen oder vorangestellte und nachgestellte Zeichen gekürzt werden sollen.

remove_chars

Die zu entfernenden Zeichen. Beachten Sie, dass `remove_chars` eine Zeichenfolge mit einer Länge von > 1 sein kann. Diese Funktion gibt die Zeichenfolge mit einem beliebigen Zeichen von `remove_chars` zurück, der am Anfang oder am Ende der entfernten Zeichenfolge stand.

Beispiele

```
TRIM('       foobar         ')               -- 'foobar'
TRIM('      \tfoobar\t         ')            -- '\tfoobar\t'
TRIM(LEADING FROM '       foobar         ')  -- 'foobar         '
TRIM(TRAILING FROM '       foobar         ') -- '       foobar'
TRIM(BOTH FROM '       foobar         ')     -- 'foobar'
TRIM(BOTH '12' FROM '1112211foobar22211122') -- 'foobar'
```

UPPER

Wandelt alle Kleinbuchstaben einer Zeichenfolge in Großbuchstaben um. Alle Zeichen, die keine Kleinbuchstaben sind, bleiben unverändert.

Syntax

```
UPPER ( string )
```

Parameter

string

Die Zielzeichenfolge, die von der Funktion verwendet wird.

Beispiele

```
UPPER('AbCdEfG!@#$') -- 'ABCDEFG!@#$'
```

Dokumentverlauf

- Letzte Aktualisierung der Dokumentation: 18. September 2019
- Aktuelle API-Version: 2006-03-01

In der folgenden Tabelle werden die wichtigen Änderungen in den einzelnen Versionen des Entwicklerhandbuch für Amazon Simple Storage Service ab dem 19. Juni 2018 beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

update-history-change	update-history-description	update-history-date
Job-Tags für S3 Stapeloperationen (p. 873)	Sie können Tags zu Ihren S3 Stapeloperationen-Aufträgen hinzufügen, um diese Aufträge zu steuern und zu bezeichnen. Weitere Informationen finden Sie unter Tags für S3 Stapeloperationen-Aufträge.	March 16, 2020
Amazon S3 Access Points (p. 873)	Amazon S3 Access Points vereinfachen das skalierbare Verwalten des Datenzugriffs für freigegebene Datasets in S3. Zugriffspunkte sind benannte Netzwerkendpunkte, die Buckets zugeordnet sind, mit denen Sie S3-Objektoperationen ausführen können. Weitere Informationen finden Sie unter Verwalten des Datenzugriffs mit Amazon S3 Access Points.	December 2, 2019
Access Analyzer für Amazon S3 (p. 873)	Access Analyzer für Amazon S3 macht Sie auf S3-Buckets aufmerksam, die so konfiguriert sind, dass jedem im Internet oder anderen AWS-Konten, einschließlich AWS-Konten außerhalb Ihrer Organisation, Zugriff gewährt wird. Weitere Informationen finden Sie unter Verwenden von Access Analyzer für Amazon S3.	December 2, 2019
S3 Replication Time Control (S3 RTC) (p. 873)	S3 Replication Time Control (S3 RTC) repliziert die meisten Objekte, die Sie zu Amazon S3 hochladen, in Sekunden und 99,99 Prozent dieser Objekte innerhalb von 15 Minuten. Weitere Informationen finden Sie unter Replizieren von Objekten	November 20, 2019

	mit S3 Replication Time Control (S3 RTC).	
Same-Region Replication (SRR, Replikation in derselben Region) (p. 873)	Mit SRR können Objekte in verschiedene Amazon S3-Buckets derselben Region kopiert werden. Informationen zur regionsübergreifenden Replikation und zur Replikation innerhalb derselben Region finden Sie unter Replikation.	September 18, 2019
Unterstützung der regionsübergreifenden Replikation für S3 Objektsperre (p. 873)	Regionsübergreifende Replikation unterstützt jetzt Objektsperre. Weitere Informationen finden Sie unter Regionsübergreifende Replikation und Was repliziert Amazon S3?.	May 28, 2019
S3 Stapeloperationen (p. 873)	Mit S3 Stapeloperationen können Sie Stapeloperationen für S3 Stapeloperationen-Objekte im großen Umfang ausführen. Amazon S3 kann eine einzelne Operation für von Ihnen angegebene Listen von Objekten ausführen. Ein einziger Auftrag kann die festgelegte Operation auf Milliarden von Objekten mit mehreren Exabytes an Daten durchführen. Weitere Informationen finden Sie unter Ausführen von S3 Stapeloperationen.	April 30, 2019
Region Asien-Pazifik (Hongkong) (p. 873)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Hongkong) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	April 24, 2019
Hinzufügung eines neuen Feldes zu den Serverzugriffsprotokollen (p. 873)	Amazon S3 fügte das folgende neue Feld zu den Serverzugriffsprotokollen hinzu: Transport Layer Security(TLS)-Version. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokoll-Format.	March 28, 2019
Neue Archivspeicherklasse (p. 873)	Amazon S3bietet eine neue Archivspeicherklasse, DEEP_ARCHIVE, für die Speicherung selten benötigter Objekte. Weitere Informationen finden Sie unter Speicherklassen.	March 27, 2019

Hinzufügung neuer Felder zu den Serverzugriffsprotokollen (p. 873)	Amazon S3 fügte die folgenden neuen Felder zu den Serverzugriffsprotokollen hinzu: Signaturversion, Cipher Suite, Authentifizierungstyp und Host Header. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokoll-Format.	March 5, 2019
Unterstützung für Parquet-formatierte Amazon S3-Bestandsdateien (p. 873)	Amazon S3 unterstützt nun für Bestandsausgabedateien das Format Apache Parquet (Parquet) zusätzlich zu den Dateiformaten Apache Optimized Row Columnar (ORC) und Comma-Separated Values (CSV). Weitere Informationen finden Sie unter Amazon S3-Bestand.	December 4, 2018
Wiederherstellungsgeschwindigkeit-Upgrade (p. 873)	Mittels des Amazon S3-Wiederherstellungsgeschwindigkeits-Upgrades können Sie während der Ausführung einer Wiederherstellung aus der Speicherklasse S3 Glacier die Geschwindigkeit der Wiederherstellung in eine höhere Geschwindigkeit ändern. Weitere Informationen finden Sie unter Wiederherstellen archivierter Objekte.	November 26, 2018
Wiederherstellen von Ereignisbenachrichtigungen (p. 873)	Amazon S3-Wiederherstellungsbenachrichtigungen unterstützen jetzt Initiierungs- und Abschlussereignisse, wenn Objekte aus der Speicherklasse S3 Glacier wiederhergestellt werden. Weitere Informationen finden Sie unter Ereignisbenachrichtigungen.	November 26, 2018
S3 Objektsperre (p. 873)	Amazon S3 stellt nun Objektsperre-Funktionalität bereit, die Write Once Read Many-Schutz für Amazon S3-Objekte bereitstellt. Weitere Informationen finden Sie unter Sperren von Objekten.	November 26, 2018

PUT direkt zur S3 Glacier-Speicherklasse (p. 873)	Die Amazon S3-PUT-Operation unterstützt nun beim Erstellen von Objekten die Angabe von S3 Glacier als Speicherklasse. Zuvor mussten Sie Objekte aus anderen Amazon S3-Speicherklassen in die Speicherklasse S3 Glacier überführen. Beim Verwenden der regionsübergreifenden Replikation (Cross Region Replication, CRR) in S3 können Sie für replizierte Objekte nun S3 Glacier als Speicherklasse angeben. Weitere Informationen zur Speicherklasse S3 Glacier finden Sie unter Speicherklassen. Weitere Informationen zum Angeben der Speicherklasse für replizierte Objekte finden Sie unter Übersicht über die Replikationskonfiguration. Weitere Informationen zum direkten PUT zu S3 Glacier-REST API-Änderungen finden Sie unter Dokumentverlauf: PUT direkt zu S3 Glacier.	November 26, 2018
Neue Speicherklasse (p. 873)	Amazon S3 stellt nun eine neue Speicherklasse mit dem Namen INTELLIGENT_TIERING bereit, die für Langzeitdaten mit veränderlichen oder unbekannten Zugriffsmustern entworfen wurde. Weitere Informationen finden Sie unter Speicherklassen.	November 26, 2018
Blockieren des öffentlichen Zugriffs in Amazon S3 (p. 873)	Amazon S3 bietet nun die Möglichkeit, den öffentlichen Zugriff auf Buckets und Objekte auf Bucket- oder Kontobasis zu blockieren. Weitere Informationen finden Sie unter Verwenden des Blockierens des öffentlichen Zugriffs in Amazon S3.	November 15, 2018

Filtern von Erweiterungen in Regeln für die regionsübergreifende Replikation (CRR) (p. 873)	Sie können nun in einer CRR-Regelkonfiguration einen Objektfilter angeben, um eine Untermenge von Objekten auszuwählen, auf die die Regel angewendet werden soll. Zuvor konnten Sie ausschließlich nach Objektschlüsselpräfixen filtern. In dieser Version können Sie nach Objektschlüsselpräfixen, einem oder mehreren Objekt-Tags oder beidem filtern. Weitere Informationen finden Sie unter CRR-Einrichtung: Übersicht über die Replikationskonfiguration.	September 19, 2018
Neue Amazon S3 Select-Funktionen (p. 873)	Amazon S3 Select unterstützt jetzt Apache Parquet-Eingaben, Abfragen für verschachtelte JSON-Objekte und zwei neue Amazon CloudWatch-Überwachungsmetriken (`SelectScannedBytes` und `SelectReturnedBytes`).	September 5, 2018
Updates jetzt über RSS verfügbar (p. 873)	Sie können jetzt einen RSS-Feed abonnieren, um Benachrichtigungen über Updates für das Entwicklerhandbuch für Amazon Simple Storage Service zu erhalten.	June 19, 2018

Frühere Aktualisierungen

In der folgenden Tabelle werden die wichtigen Änderungen in den einzelnen Versionen des Entwicklerhandbuch für Amazon Simple Storage Service vor dem 19. Juni 2018 beschrieben.

Änderung	Beschreibung	Datum
Aktualisierung von Codebeispielen	Aktualisierte Codebeispiele: • C# – Alle Beispiele wurden für die Verwendung des aufgabenbasierten asynchronen Musters aktualisiert. Weitere Informationen finden Sie unter Asynchrone Amazon Web Services-APIs für .NET im AWS SDK für .NET-Entwicklerhandbuch. Codebeispiele sind nun mit Version 3 des AWS SDK für .NET konform. • Java – Alle Beispiele wurden für die Verwendung des Client Builder-Modells aktualisiert. Weitere Informationen zum Client Builder-Modell finden Sie unter Erstellen von Service-Clients. • PHP – Die Beispiele wurden für die Verwendung mit AWS SDK für PHP 3.0 aktualisiert. Weitere Informationen zu AWS SDK für PHP 3.0 finden Sie unter AWS SDK for PHP.	30. April 2018

Änderung	Beschreibung	Datum
	• Ruby – Der Beispiel-Code wurde für die Verwendung mit AWS SDK für Ruby Version 3 aktualisiert.	
Amazon S3 meldet jetzt die Speicherklassen S3 Glacier und ONEZONE_IA an Amazon CloudWatch Logs-Speichermetriken.	Diese Speichermetriken melden nicht nur tatsächliche Bytes, sondern beinhalten auch Overhead-Bytes pro Objekt für entsprechende Speicherklassen (ONEZONE_IA, STANDARD_IA und S3 Glacier): • Für Objekte der Speicherklassen ONEZONE_IA und STANDARD_IA meldet Amazon S3 Objekte kleiner als 128 KB als 128 KB. Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122). • Für Objekte der Speicherklasse S3 Glacier melden die Speichermetriken die folgenden Overheads: • Ein Overhead von 32 KB pro Objekt zu dem Preis, der für Speicherklasse S3 Glacier berechnet wird • Ein Overhead von 8 KB pro Objekt zu dem Preis, der Speicherklasse STANDARD berechnet wird Weitere Informationen finden Sie unter Übergang von Objekten mit dem Amazon S3-Lebenszyklus (p. 141). Weitere Informationen zu Speichermetriken finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).	30. April 2018
Neue Speicherklasse	Amazon S3 bietet jetzt eine neue Speicherklasse, ONEZONE_IA (IA für "Infrequent Access" (seltener Zugriff)), um Objekte zu speichern. Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122).	4. April 2018
Amazon S3 Select	Amazon S3 unterstützt nun den Abruf von Objektinhalten auf Basis eines SQL-Ausdrucks. Weitere Informationen finden Sie unter Auswählen von Inhalten aus Objekten (p. 268).	4. April 2018
Region Asien-Pazifik (Osaka – regional)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Osaka-Lokal) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference. Important Sie können die Region Asien-Pazifik (Osaka-Lokal) ausschließlich in Verbindung mit der Region Asien-Pazifik (Tokio) verwenden. Um Zugang zur Region Asien-Pazifik (Osaka-Lokal) zu erhalten, wenden Sie sich an Ihren Vertriebsmitarbeiter.	12. Februar 2018
Zeitstempel der Amazon S3-Bestandserstellung	Der Amazon S3-Bestand enthält nun einen Zeitstempel mit Datum und Startzeit der Erstellung des Amazon S3-Bestandsberichts. Mithilfe des Zeitstempels können Sie Änderungen in Ihrem Amazon S3-Speicher ab dem Startzeitpunkt, zu dem der Bestandsbericht erstellt wurde, ermitteln.	16. Januar 2018

Änderung	Beschreibung	Datum
Region Europa (Paris)	Amazon S3 ist jetzt in der Region Europa (Paris) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	18. Dezember 2017
Region „China (Ningxia)“	Amazon S3 ist jetzt in der Region China (Ningxia) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	11. Dezember 2017
Abfragen von Archiven mit SQL	Amazon S3 unterstützt jetzt das Abfragen von S3 Glacier-Datenarchiven mit SQL. Weitere Informationen finden Sie unter Abfrage von archivierten Objekten (p. 277).	29. November 2017
Unterstützung für ORC-formatierte Amazon S3-Bestandsdateien	Amazon S3 unterstützt jetzt für Bestandsausgabedateien zusätzlich zum Comma-Separated Values (CSV)-Dateiformat das Format Apache Optimized Row Columnar (ORC). Sie können nun den Amazon S3-Bestand auch mittels Standard-SQL abfragen, indem Sie Amazon Athena, Amazon Redshift Spectrum und weitere Tools wie Presto, Apache Hive und Apache Spark verwenden. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503).	17. November 2017
Standard-Verschlüsselung für S3-Buckets	Die Amazon S3-Standardverschlüsselung bietet eine Methode zum Festlegen des Verhaltens der Standardverschlüsselung für einen S3-Bucket. Sie können die Standardverschlüsselung in einem Bucket festlegen, sodass alle Objekte beim Speichern im Bucket verschlüsselt werden. Die Objekte werden unter Verwendung von serverseitiger Verschlüsselung entweder mit Amazon S3-verwalteten Schlüsseln (SSE-S3) oder AWS KMS-verwalteten Schlüsseln (SSE-KMS) verschlüsselt. Weitere Informationen finden Sie unter Amazon S3-Standardverschlüsselung für S3-Buckets (p. 72).	06. November 2017
Verschlüsselungsstatus in Amazon S3-Bestand	Amazon S3 unterstützt jetzt den Einschluss des Verschlüsselungsstatus in den Amazon S3-Bestand, sodass Sie zu Audit- oder anderen Zwecken die Art der Verschlüsselung Ihrer Objekte im Ruhezustand anzeigen können. Sie können auch so konfigurieren, dass S3-Bestand mit serverseitiger Verschlüsselung (SSE) oder SSE-KMS verschlüsselt wird, damit alle Bestandsdateien entsprechend verschlüsselt werden. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503).	06. November 2017

Änderung	Beschreibung	Datum
Erweiterungen der regionsübergreifenden Replikation (CRR)	Die regionsübergreifende Replikation unterstützt jetzt Folgendes: • In einem kontoübergreifenden Szenario können Sie eine CRR-Konfiguration hinzufügen, um die Replikateigentümerschaft dem AWS-Konto zuzuweisen, das den Ziel-Bucket besitzt. Weitere Informationen finden Sie unter Ändern des Replikateigentümers (p. 690). • Amazon S3 repliziert standardmäßig keine Objekte in Ihrem Quell-Bucket, die unter Verwendung serverseitiger Verschlüsselung mit in AWS KMS gespeicherten Schlüsseln erstellt werden. In Ihrer CRR-Konfigurationsdatei können Sie Amazon S3 nun so einrichten, dass es diese Objekte repliziert. Weitere Informationen finden Sie unter Replizieren von Objekten, die mit der serverseitigen Verschlüsselung (SSE) unter Verwendung von in AWS KMS gespeicherten Verschlüsselungsschlüsseln erstellt wurden (p. 693).	06. November 2017
Region Europa (London)	Amazon S3 ist jetzt in der Region Europa (London) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	13. Dezember 2016
Region Kanada (Zentral)	Amazon S3 ist jetzt in der Region Kanada (Zentral) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	8. Dezember 2016

Änderung	Beschreibung	Datum
Markieren von Objekten	Amazon S3 unterstützt jetzt das Markieren von Objekten. Das Markieren von Objekten ermöglicht Ihnen, Speicher zu kategorisieren. Präfixe für Objektschlüsselnamen ermöglichen Ihnen außerdem, die Speicherung zu kategorisieren. Die Markierung von Objekten fügt ihr eine weitere Dimension hinzu. Die Markierung bietet weitere Vorteile. Dazu zählen: • Objekt-Tags bieten eine differenzierte Zugriffskontrolle für Berechtigungen (Sie könnten beispielsweise einem IAM-Benutzer die Berechtigungen für schreibgeschützte Objekte mit bestimmten Tags erteilen). • Differenzierte Kontrolle bei der Spezifizierung einer Lebenszykluskonfiguration. Sie können Tags spezifizieren, um eine Untermenge von Objekten auszuwählen, auf welche die Lebenszyklus-Regel zutrifft. • Wenn Sie eine regionsübergreifende Replikation (CRR) konfiguriert haben, kann Amazon S3 die Tags replizieren. Sie müssen der für Amazon S3 erstellten IAM-Rolle die erforderlichen Berechtigungen erteilen, um Objekte zu replizieren. • Sie können außerdem CloudWatch-Metriken und CloudTrail-Ereignisse anpassen, um Informationen nach spezifischen Tag-Filtern anzuzeigen. Weitere Informationen finden Sie unter Markieren von Objekten (p. 130).	29. November 2016
Der Amazon S3-Lebenszyklus unterstützt nun Tag-basierte Filter.	Amazon S3 unterstützt jetzt auf Tags basierende Filter bei der Konfiguration des Lebenszyklus. Sie können jetzt Lebenszyklusregeln angeben, in denen Sie ein Schlüsselpräfix, ein oder mehrere Objekt-Tags oder eine Kombination aus beidem angeben, um eine Untermenge der Objekte auszuwählen, auf welche die Lebenszyklusregel zutrifft. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).	29. November 2016
CloudWatch-Anforderungsmetriken für Buckets	Amazon S3 unterstützt jetzt CloudWatch-Metriken für Anforderungen für Buckets. Wenn Sie diese Metriken für einen Bucket aktivieren, erstellen diese Metriken Berichte in 1-Minuten-Intervallen. Sie können auch konfigurieren, welche Objekte in einem Bucket diese Anforderungsmetriken melden. Weitere Informationen finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).	29. November 2016
Amazon S3-Bestand	Amazon S3 unterstützt jetzt den Speicherbestand. Der Amazon S3-Bestand stellt eine Flat-Datei-Ausgabe Ihrer Objekte und der zugehörigen Metadaten auf täglicher oder wöchentlicher Basis für einen S3-Bucket oder ein gemeinsames Präfix (d. h. für Objekte, deren Namen mit derselben Zeichenfolge beginnen) bereit. Weitere Informationen finden Sie unter Amazon S3-Bestand (p. 503).	29. November 2016

Änderung	Beschreibung	Datum
Amazon S3-Analyse – Speicherklassenanalyse	Die neue analytische Funktion von Amazon S3, die Speicherklassenanalyse, beobachtet Datenzugriffsmuster, anhand derer Sie entscheiden können, wann Sie STANDARD-Speicher mit weniger häufigem Zugriff in die Speicherklasse STANDARD_IA (IA für "infrequent access" (seltener Zugriff)) überführen sollen. Wenn die Speicherklassenanalyse die seltenen Datenzugriffsmuster für eine gefilterte Datenmenge im Laufe der Zeit beobachtet, können Sie Ihre Lebenszyklusrichtlinien unter Verwendung der Analyseergebnisse verbessern. Diese Funktion beinhaltet auch eine detaillierte tägliche Analyse Ihrer Speichernutzung auf der angegebenen Bucket-, Präfix- oder Tag-Ebene, die Sie in einen S3-Bucket exportieren können. Weitere Informationen finden Sie unter Amazon S3-Analyse – Speicherklassenanalyse (p. 282) im Entwicklerhandbuch für Amazon Simple Storage Service.	29. November 2016
Neuer Expedited- und Bulk-Datenabruf bei der Wiederherstellung archivierter Objekte aus S3 Glacier	Amazon S3 unterstützt jetzt beim Wiederherstellen von Objekten, die zu S3 Glacier archiviert wurden, zusätzlich zu Standardabrufen auch Expedited- und Bulk-Datenabrufe. Weitere Informationen finden Sie unter Wiederherstellen archivierter Objekte (p. 272).	21. November 2016
CloudTrail-Objektprotokollierung	CloudTrail unterstützt die Protokollierung von Amazon S3-API-Operationen auf Objektebene, z. B. `GetObject`, `PutObject` und `DeleteObject`. Sie können Ihre Ereignisauswahlen so konfigurieren, dass API-Operationen auf Objektebene protokolliert werden. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail (p. 750).	21. November 2016
Region USA Ost (Ohio)	Amazon S3 ist jetzt in der Region USA Ost (Ohio) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	17. Oktober 2016
IPv6-Unterstützung für Amazon S3 Transfer Acceleration	Amazon S3 unterstützt jetzt Internetprotokollversion 6 (IPv6) für Amazon S3 Transfer Acceleration. Sie können Verbindungen mit Amazon S3 über IPv6 mittels des neuen Dual-Stack-Endpunkts für Transfer Acceleration herstellen. Weitere Informationen finden Sie unter Erste Schritte mit Amazon S3 Transfer Acceleration (p. 76).	6. Oktober 2016
IPv6-Support	Amazon S3 unterstützt jetzt Internetprotokollversion 6 (IPv6). Sie können mittels Dual-Stack-Endpunkten auf Amazon S3 über IPv6 zugreifen. Weitere Informationen finden Sie unter Senden von Anforderungen an Amazon S3 über IPv6 (p. 13).	11. August 2016
Region Asien-Pazifik (Mumbai)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Mumbai) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	27. Juni 2016

Änderung	Beschreibung	Datum
Amazon S3 Transfer Acceleration	Amazon S3 Transfer Acceleration unterstützt schnelle, einfache und sichere Übertragungen von Dateien zwischen Ihrem Client und einem S3-Bucket über große Entfernungen. Transfer Acceleration nutzt die global verteilten Edge-Standorte von Amazon CloudFront. Weitere Informationen finden Sie unter Amazon S3 Transfer Acceleration (p. 75).	19. April 2016
Lebenszyklusunterstützung um abgelaufene Löschmarkierungen für Objekte zu entfernen	Die Lebenszykluskonfigurations-Aktion `Expiration` ermöglicht Ihnen nun, Amazon S3 anzuweisen, abgelaufene Objektlöschungsmarkierungen in versionierten Buckets zu entfernen. Weitere Informationen finden Sie unter Elemente, die Lebenszyklusaktionen beschreiben (p. 151).	16. März 2016
Die Bucket-Lebenszykluskonfiguration unterstützt jetzt eine Aktion, um unvollständige mehrteilige Uploads abzubrechen	Die Bucket-Lebenszykluskonfiguration unterstützt jetzt die Aktion `AbortIncompleteMultipartUpload`. Mit dieser Aktion können Sie Amazon S3 zum Abbrechen mehrteiliger Uploads anweisen, die nicht innerhalb einer bestimmten Anzahl von Tagen nach der Initiierung abgeschlossen werden. Wenn ein mehrteiliger Upload für eine Abbruchoperation in Frage kommt, löscht Amazon S3 alle hochgeladenen Teile und bricht den mehrteiligen Upload ab. Informationen zum Konzept finden Sie in den folgenden Themen im Entwicklerhandbuch für Amazon Simple Storage Service: • Abbruch unvollständiger mehrteiliger Uploads unter Verwendung einer Bucket-Lebenszyklusrichtlinie (p. 201) • Elemente, die Lebenszyklusaktionen beschreiben (p. 151) Die folgenden API-Operationen wurden aktualisiert, um die neue Aktion zu unterstützen: • PUT-Bucket-Lebenszyklus – Die XML-Konfiguration ermöglicht Ihnen jetzt die Angabe der Aktion `AbortIncompleteMultipartUpload` in einer Lebenszykluskonfigurationsregel. • List Parts und Initiate Multipart Upload – Beide API-Operationen geben nun zwei zusätzliche Antwort-Header zurück (`x-amz-abort-date` und `x-amz-abort-rule-id`), wenn der Bucket eine Lebenszyklusregel besitzt, die die Aktion `AbortIncompleteMultipartUpload` angibt. Diese Header in der Antwort geben an, wenn der initiierte mehrteilige Upload für eine Abbruchoperation in Frage kommen, und welche Lebenszyklusregel gilt.	16. März 2016
Region Asien-Pazifik (Seoul)	Amazon S3 ist jetzt in der Region Asien-Pazifik (Seoul) verfügbar. Weitere Informationen zu Amazon S3-Regionen und -Endpunkten finden Sie unter Regionen und Endpunkte in der AWS General Reference.	6. Januar 2016

Änderung	Beschreibung	Datum
Neuer Bedingungsschlüssel und eine Änderung des mehrteiligen Uploads	IAM-Richtlinien unterstützen jetzt einen Amazon S3 `s3:x-amz-storage-class`-Bedingungsschlüssel. Weitere Informationen finden Sie unter Amazon S3-Bedingungsschlüssel (p. 382). Sie müssen nicht mehr der Initiator eines mehrteiligen Uploads sein, um Teile hochzuladen und den Upload abzuschließen. Weitere Informationen finden Sie unter Multipart-Upload-API und Berechtigungen (p. 204).	14. Dezember 2015
Umbenennung der US-Standardregion	Die Regionsnamen-Zeichenfolge hat sich von "USA Standard" in "USA Ost (Nord-Virginia)" geändert. Dies ist nur eine Aktualisierung des Regionsnamens, die Funktionalität bleibt unverändert.	11. Dezember 2015
Neue Speicherklasse	Amazon S3 bietet jetzt eine neue Speicherklasse, STANDARD_IA (IA für "infrequent access" (seltener Zugriff)), um Objekte zu speichern. Diese Speicherklasse ist für langlebige Daten und Daten mit weniger häufigem Zugriff optimiert. Weitere Informationen finden Sie unter Amazon S3-Speicherklassen (p. 122). Die Aktualisierungen der Lebenszykluskonfiguration ermöglichen Ihnen jetzt, Objekte in die Speicherklasse STANDARD_IA zu überführen. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139). Zuvor hat die Funktion der regionsübergreifenden Replikation die Speicherklasse des Quellobjekts für Objektreplikate verwendet. Wenn Sie jetzt eine regionsübergreifende Replikation konfigurieren, können Sie eine Speicherklasse für das im Ziel-Bucket erstellte Objektreplikat spezifizieren. Weitere Informationen finden Sie unter Replikation (p. 669).	16. September 2015
Integration von AWS CloudTrail	Die neue AWS CloudTrail-Integration ermöglicht Ihnen jetzt das Aufzeichnen von Amazon S3-API-Aktivitäten in Ihrem S3-Bucket. Sie können CloudTrail verwenden, um das Erstellen oder Löschen von S3-Buckets nachzuverfolgen, die Zugriffskontrolle oder die Lebenszyklusrichtlinie zu ändern. Weitere Informationen finden Sie unter Protokollieren von Amazon S3-API-Aufrufen mithilfe von AWS CloudTrail (p. 750).	1. September 2015
Erhöhung des Bucket-Limits	Amazon S3 unterstützt jetzt die Erhöhung des Bucket-Limits. In der Standardeinstellung können die Kunden pro AWS-Konto bis zu 100 Buckets erstellen. Kunden, die weitere Buckets benötigen, können dieses Limit erhöhen, indem sie eine Service-Limit-Erhöhung senden. Weitere Informationen zum Erhöhen von Limits für Ihren Bucket finden Sie unter AWS Service Limits in der Allgemeinen AWS-Referenz. Weitere Informationen finden Sie unter Erstellen eines Buckets (p. 57) und Beschränkungen und Einschränkungen von Buckets (p. 64).	4. August 2015

Änderung	Beschreibung	Datum
Aktualisierung des Datenkonsistenzmodells	Amazon S3 unterstützt jetzt in der Region USA Ost (Nord-Virginia) Read-After-Write-Konsistenz für neue Objekte, die Amazon S3 hinzugefügt werden. Vor dieser Aktualisierung unterstützten alle Regionen außer der Region USA Ost (Nord-Virginia) „Read-After-Write-Consistency“ (Lesen-nach-Schreiben-Konsistenz) für neue Objekte, die Amazon S3 hinzugefügt wurden. Mit dieser Erweiterung unterstützt Amazon S3 jetzt die Read-After-Write-Konsistenz für neue Objekte, die Amazon S3 hinzugefügt werden, in allen Regionen. Die „Read-After-Write-Consistency“ (Lesen-nach-Schreiben-Konsistenz) gestattet Ihnen, Objekte unmittelbar nach ihrer Erstellung in Amazon S3 abzurufen. Weitere Informationen finden Sie unter Regionen (p. 4).	4. August 2015
Ereignisbenachrichtigungen	Amazon S3-Ereignisbenachrichtigungen wurden aktualisiert, um Benachrichtigungen hinzuzufügen, wenn Objekte gelöscht wurden, und das Filtern für Objektnamen mit Präfix- und Suffix-Vergleich hinzuzufügen. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).	28. Juli 2015
Integration von Amazon CloudWatch	Aufgrund der neuen Amazon CloudWatch-Integration können Sie jetzt Alarme für Ihre Amazon S3-Nutzung mittels CloudWatch-Metriken für Amazon S3 überwachen und einrichten. Unterstützte Metriken sind unter anderem die Gesamtzahl der Bytes für den Standardspeicher, die Gesamtzahl der Bytes für den Speicher mit reduzierter Redundanz, sowie die Gesamtzahl der Objekte für einen bestimmten S3-Bucket. Weitere Informationen finden Sie unter Überwachung von Metriken mit Amazon CloudWatch (p. 739).	28. Juli 2015
Unterstützung für das Löschen und Leeren nicht leerer Buckets	Amazon S3 unterstützt jetzt das Löschen und Leeren nicht leerer Buckets. Weitere Informationen finden Sie unter Löschen oder Leeren von Buckets (p. 69).	16. Juli 2015
Bucket-Richtlinien für Amazon VPC-Endpunkte	Amazon S3 hat Unterstützung für Bucket-Richtlinien für Amazon Virtual Private Cloud (Amazon VPC)-Endpunkte hinzugefügt. Sie können den Zugriff auf S3-Buckets von bestimmten Amazon VPC-Endpunkten oder bestimmten VPCs über Bucket-Richtlinien steuern. VPC-Endpunkte sind einfach zu konfigurieren, sind höchst zuverlässig und bieten eine sichere Verbindung zu Amazon S3, ohne dass ein Gateway oder eine NAT-Instance erforderlich sind. Weitere Informationen finden Sie unter Beispiel für Bucket-Richtlinien für VPC-Endpunkte für Amazon S3 (p. 454).	29. April 2015
Ereignisbenachrichtigungen	Amazon S3-Ereignisbenachrichtigungen wurden aktualisiert, um den Wechsel zu ressourcenbasierten Berechtigungen für AWS Lambda zu unterstützen. Weitere Informationen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).	9. April 2015

Änderung	Beschreibung	Datum
Regionsübergreifende Replikation	Amazon S3 unterstützt jetzt eine regionsübergreifende Replikation. Die regionsübergreifende Replikation ist ein automatisches, asynchrones und Bucket-übergreifendes Kopieren von Objekten in verschiedenen AWS-Regionen. Weitere Informationen finden Sie unter Replikation (p. 669).	24. März 2015
Ereignisbenachrichtigungen	Amazon S3 unterstützt jetzt neue Ereignistypen und Ziele in einer Bucket-Benachrichtigungskonfiguration. Vor dieser Version hat Amazon S3 ausschließlich den Ereignistyp s3:ReducedRedundancyLostObject und ein Amazon SNS-Thema als Ziel unterstützt. Weitere Informationen zu den neuen Ereignistypen finden Sie unter Konfigurieren von Amazon S3-Ereignisbenachrichtigungen (p. 646).	13. November 2014
Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln	Serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS) Amazon S3 unterstützt nun die serverseitige Verschlüsselung mit AWS KMS. Mit dieser Funktion können Sie den Envelope-Schlüssel über AWS KMS verwalten. Amazon S3 ruft AWS KMS auf, um innerhalb der von Ihnen festgelegten Berechtigungen auf den Envelope-Schlüssel zuzugreifen. Weitere Informationen zur serverseitigen Verschlüsselung mit AWS KMS finden Sie unter Schützen von Daten mithilfe der serverseitigen Verschlüsselung mit dem AWS Key Management Service.	12. November 2014
Region Europa (Frankfurt)	Amazon S3 ist jetzt in der Region Europa (Frankfurt) verfügbar.	23. Oktober 2014
Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln	Amazon S3 unterstützt jetzt die serverseitige Verschlüsselung mit vom Kunden bereitgestellten Verschlüsselungsschlüsseln (SSE-C). Die serverseitige Verschlüsselung ermöglicht Ihnen, Amazon S3 aufzufordern, Ihre ruhenden Daten zu verschlüsseln. Wenn Sie SSE-C verwenden, verschlüsselt Amazon S3 Ihre Objekte mit den von Ihnen bereitgestellten benutzerdefinierten Verschlüsselungsschlüsseln. Amazon S3 übernimmt die Verschlüsselung für Sie, deshalb können Sie von der Verwendung Ihrer eigenen Verschlüsselungsschlüssel profitieren, ohne die Kosten für das Schreiben oder Ausführen Ihres eigenen Verschlüsselungscodes tragen zu müssen. Weitere Informationen zu SSE-C finden Sie unter Serverseitige Verschlüsselung (Verwenden von kundenseitig bereitgestellten Verschlüsselungsschlüsseln).	12. Juni 2014
Lebenszyklus-Unterstützung für das Versioning	Vor diesem Release wurde die Lebenszykluskonfiguration nur für nicht versionsfähige Buckets unterstützt. Jetzt können Sie den Lebenszyklus sowohl für nicht versionsfähige, als auch für versionsfähige Buckets konfigurieren. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).	20. Mai 2014
Überarbeitung der Themen zur Zugriffskontrolle	Überarbeitung der Dokumentation zur Amazon S3-Zugriffskontrolle. Weitere Informationen finden Sie unter Identity and Access Management in Amazon S3 (p. 329).	15. April 2014

Änderung	Beschreibung	Datum
Überarbeitung des Themas zur Protokollierung des Server-Zugriffs	Überarbeitung der Dokumentation zur Protokollierung des Server-Zugriffs. Weitere Informationen finden Sie unter Amazon S3-Serverzugriffsprotokollierung (p. 777).	26. November 2013
Aktualisierung der .NET SDK-Beispiele auf Version 2.0	Die .NET SDK-Beispiele in diesem Handbuch sind jetzt konform zu Version 2.0.	26. November 2013
SOAP-Unterstützung über HTTP veraltet	Die SOAP-Unterstützung über HTTP ist veraltet, steht über HTTPS aber noch zur Verfügung. Die neuen Amazon S3-Funktionen werden unter SOAP nicht unterstützt. Wir empfehlen, entweder die REST API oder die AWS SDKs zu verwenden.	20. September 2013
Support für IAM-Richtlinienvariablen	Die IAM-Zugriffsrichtliniensprache unterstützt nun Variablen. Wenn eine Richtlinie ausgewertet wird, werden alle Richtlinienvariablen durch Werte ersetzt, die von Kontext-basierten Informationen aus der Sitzung des authentifizierten Benutzers bereitgestellt werden. Sie können Richtlinienvariablen zum Definieren von allgemeinen Richtlinien nutzen, ohne explizit alle Komponenten der Richtlinien aufzulisten. Weitere Informationen über Richtlinienvariablen finden Sie unter Übersicht über IAM-Richtlinienvariablen im IAM-Benutzerhandbuch. Beispiele für Richtlinienvariablen in Amazon S3 finden Sie unter Beispiele für Benutzerrichtlinien (p. 456).	3. April 2013
Konsolensupport für die Zahlung durch den Anforderer	Sie können jetzt Ihren Bucket als Bucket mit Zahlung durch den Auftraggeber unter Verwendung der Amazon S3-Konsole konfigurieren. Weitere Informationen finden Sie unter Konfigurieren des Zahlung durch den Anforderer unter Verwendung der Amazon S3-Konsole (p. 83).	31. Dezember 2012
Root-Domänen-Unterstützung für das Website-Hosting	Amazon S3 unterstützt jetzt das Hosting statischer Websites in der Root-Domäne. Besucher Ihrer Website können jetzt von ihrem Browser aus auf Ihre Site zugreifen, ohne das "www" in der Webadresse angeben zu müssen (z. B. "example.com"). Viele Kunden hosten bereits statische Websites auf Amazon S3, die von einer "www"-Subdomain aus zugreifbar sind (z. B. "www.example.com"). Zuvor mussten Sie für eine Unterstützung eines Root-Domänen-Zugriffs Ihren eigenen Webserver betreiben, um die Root-Domänen-Anforderungen über einen Proxy von den Browsern auf Ihre Website auf Amazon S3 weiterzugeben. Der Betrieb eines Web-Servers, um Anforderungen über einen Proxy weiterzugeben, führt zusätzliche Kosten, Betriebsaufwand und einen weitere potenziellen Ausfallpunkt ein. Jetzt können Sie den Vorteil der hohen Verfügbarkeit und Robustheit von Amazon S3 für "www"- und Stammdomänen-Adressen nutzen. Weitere Informationen finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).	27. Dezember 2012

Änderung	Beschreibung	Datum
Überarbeitung der Konsole	Die Amazon S3-Konsole wurde aktualisiert. Die Dokumentationsthemen, die auf die Konsole verweisen, wurden entsprechend überarbeitet.	14. Dezember 2012
Unterstützung der Archivierung von Daten in S3 Glacier	Amazon S3 unterstützt jetzt eine Speicheroption, mit der Sie den kostengünstigen Speicherservice von S3 Glacier für die Datenarchivierung nutzen können. Für die Archivierung von Objekten definieren Sie Archivierungsregeln, die Objekte identifizieren, und eine Timeline, die festlegt, wann Amazon S3 diese Objekte zu S3 Glacier archivieren soll. Sie können die Regeln für einen Bucket leicht mithilfe der Amazon S3-Konsole oder programmgesteuert über die Amazon S3-API oder AWS SDKs festlegen. Weitere Informationen finden Sie unter Verwaltung des Objektlebenszyklus (p. 139).	13. November 2012
Unterstützung für Seitenumleitungen von Websites	Für einen Bucket, der als Website konfiguriert ist, unterstützt Amazon S3 jetzt die Umleitung einer Anforderung für ein Objekt auf ein anderes Objekt im selben Bucket oder zu einer externen URL. Weitere Informationen finden Sie unter (Optional) Konfigurieren einer Webseiten-Umleitung (p. 614). Weitere Informationen zum Hosting von Websites finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).	4. Oktober 2012
Unterstützung von Cross-Origin Resource Sharing (CORS)	Amazon S3 unterstützt jetzt Cross-Origin Resource Sharing (CORS). CORS bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Mithilfe der CORS-Unterstützung in Amazon S3 können Sie umfassende clientseitige Webanwendungen in Amazon S3 erstellen und selektiv den domänenübergreifenden Zugriff auf Ihre Amazon S3-Ressourcen zulassen. Weitere Informationen finden Sie unter Cross-Origin Resource Sharing (CORS) (p. 174).	31. August 2012
Unterstützung von Kostenzuordnungs-Tags	Amazon S3 unterstützt jetzt Kostenzuordnungs-Tags, um S3-Buckets zu beschriften, sodass Sie die Kosten einfacher für Projekte oder andere Kriterien nachverfolgen können. Weitere Informationen zur Verwendung der Markierung von Buckets finden Sie unter Verwenden von Kostenzuordnungs-Tags für S3-Buckets (p. 100).	21. August 2012

Änderung	Beschreibung	Datum
Unterstützung für MFA-geschütztem API-Zugriff in Bucket-Richtlinien	Amazon S3 unterstützt jetzt den MFA-geschützten API-Zugriff. Diese Funktion kann die AWS-Multifaktorauthentifizierung durchsetzen, um beim Zugriff auf Ihre Amazon S3-Ressourcen zusätzlichen Schutz zu bieten. Es handelt sich um eine Sicherheitsfunktion, die die Angabe eines gültigen MFA-Codes von Benutzern erfordert, mit dem das physische Eigentum eines MFA-Geräts belegt wird. Weitere Informationen finden Sie unter AWS-Multifaktor-Authentifizierung. Sie können jetzt die MFA-Authentifizierung für alle Anforderungen fordern, die auf Ihre Amazon S3-Ressourcen zugreifen. Um eine MFA-Authentifizierung durchzusetzen, unterstützt Amazon S3 jetzt den Schlüssel `aws:MultiFactorAuthAge` in Bucket-Richtlinien. Ein Beispiel für eine Bucket-Richtlinie finden Sie in Beispiel Hinzufügen einer Bucket-Richtlinie zur Anforderung einer MFA (p. 451).	10. Juli 2012
Unterstützung des Ablaufens von Objekten	Sie können das Ablaufen von Objekten nutzen, um das automatische Entfernen von Daten nach einem konfigurierten Zeitraum einzuplanen. Sie legen die Ablaufzeit eines Objekts fest, indem Sie einem Bucket eine Lebenszykluskonfiguration hinzufügen.	27. Dezember 2011
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region Südamerika (São Paulo). Weitere Informationen finden Sie unter Zugriff auf einen Bucket (p. 60).	14. Dezember 2011
Multi-Object Delete	Amazon S3 unterstützt jetzt die Multi-Object Delete API, mit der Sie mehrere Objekte innerhalb einer einzigen Anforderung löschen können. Mit dieser Funktion können Sie eine große Anzahl an Objekten aus Amazon S3 schneller als unter Verwendung mehrerer einzelner DELETE-Anforderungen löschen. Weitere Informationen finden Sie unter Löschen von Objekten (p. 250).	7. Dezember 2011
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region USA West (Oregon). Weitere Informationen finden Sie unter Buckets und Regionen (p. 60).	8. November 2011
Aktualisierung der Dokumentation	Korrektur von Fehlern in der Dokumentation	8. November 2011
Aktualisierung der Dokumentation	Neben den Korrekturen von Fehlern in der Dokumentation enthält diese Version auch die folgenden Erweiterungen: • Neue Abschnitte zur serverseitigen Verschlüsselung mit AWS SDK für PHP (siehe Festlegen einer serverseitigen Verschlüsselung mit dem AWS SDK für PHP (p. 303)) und AWS SDK für Ruby (siehe Festlegen einer serverseitigen Verschlüsselung mit der AWS SDK für Ruby (p. 305)). • Neuer Abschnitt zum Erstellen und Testen von Ruby-Beispielen (siehe Verwenden von AWS SDK für Ruby – Version 3 (p. 813)).	17. Oktober 2011

Änderung	Beschreibung	Datum
Unterstützung der serverseitigen Verschlüsselung	Amazon S3 unterstützt jetzt serverseitige Verschlüsselung Es ermöglicht Ihnen, Amazon S3 aufzufordern, Ihre ruhenden Daten zu verschlüsseln, d. h. Ihre Objektdaten zu verschlüsseln, wenn Amazon S3 Ihre Daten in seinen Rechenzentren auf Datenträger schreibt. Neben den REST API-Aktualisierungen bieten AWS SDK for Java und .NET die erforderliche Funktionalität, eine serverseitige Verschlüsselung anzufordern. Sie können auch eine serverseitige Verschlüsselung anfordern, wenn Sie Objekte mit der AWS-Managementkonsole hochladen. Weitere Informationen zur Datenverschlüsselung finden Sie unter Verwenden der Datenverschlüsselung.	4. Oktober 2011
Aktualisierung der Dokumentation	Neben den Korrekturen von Fehlern in der Dokumentation enthält diese Version auch die folgenden Erweiterungen: • Dem Abschnitt Senden von Anforderungen (p. 11) wurden Ruby- und PHP-Beispiele hinzugefügt. • Hinzufügung von Abschnitten, die Generierung und Verwendung vorsignierter URLs beschreiben. Weitere Informationen finden Sie unter Ein Objekt mit anderen teilen (p. 190) und Hochladen von Objekten unter Verwendung vorsignierter URLs (p. 228). • Aktualisierung eines vorhandenen Abschnitts, um AWS Explorer für Eclipse und Visual Studio einzuführen. Weitere Informationen finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).	22. September 2011
Unterstützung von Anforderungen unter Verwendung temporärer Sicherheitsanmeldeinformationen	Zusätzlich zur Verwendung Ihres AWS-Kontos und von IAM-Sicherheitsanmeldeinformationen für Benutzer, um authentifizierte Anforderungen an Amazon S3 zu senden, können Sie jetzt Anforderungen mittels temporärer Sicherheitsanmeldeinformationen senden, die Sie von AWS Identity and Access Management (IAM) erhalten. Sie können die AWS Security Token Service API oder die AWS SDK Wrapper-Bibliotheken verwenden, um diese temporären Anmeldeinformationen von IAM anzufordern. Sie können diese temporären Anmeldeinformationen für Ihre eigene Verwendung erhalten, oder sie verbundenen Benutzern und Anwendungen erteilen. Diese Funktion ermöglicht Ihnen, Ihre Benutzer außerhalb von AWS zu verwalten und ihnen temporäre Sicherheitsanmeldeinformationen für den Zugriff auf Ihre AWS-Ressourcen zu erteilen. Weitere Informationen finden Sie unter Senden von Anforderungen (p. 11). Weitere Informationen zur IAM-Unterstützung für temporäre Sicherheitsanmeldeinformationen finden Sie unter Temporäre Sicherheits-Anmeldeinformationen im IAM-Benutzerhandbuch.	3. August 2011

Änderung	Beschreibung	Datum
Die API für mehrteilige Uploads wurde erweitert, um das Kopieren von Objekten bis zu 5 TB zu ermöglichen	Vor diesem Release hat die Amazon S3 API nur das Kopieren von Objekten von bis zu 5 GB unterstützt. Um das Kopieren von Objekten zu unterstützen, die größer als 5 GB sind, erweitert Amazon S3 jetzt die API für mehrteilige Uploads um eine neue Operation, `Upload Part (Copy)`. Sie können diese Operation für einen mehrteiligen Upload verwenden, um Objekte mit einer Größe von 5 TB zu kopieren. Weitere Informationen finden Sie unter Objekte kopieren (p. 233). Weitere konzeptuelle Informationen über mehrteilige Uploads finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).	21. Juni 2011
SOAP API-Anrufe über HTTP deaktiviert	Um die Sicherheit zu erhöhen, wurden SOAP API-Aufrufe über HTTP deaktiviert. Authentifizierte und anonyme SOAP-Anforderungen müssen mit SSL an Amazon S3 gesendet werden.	6. Juni 2011
IAM ermöglicht eine kontoübergreifende Delegierung	Um auf eine Amazon S3-Ressource zuzugreifen, benötigte ein IAM-Benutzer zuvor Berechtigungen vom übergeordneten AWS-Konto und vom Besitzer der Amazon S3-Ressource. Mit dem kontenübergreifenden Zugriff bracht der IAM-Benutzer jetzt nur die Berechtigung vom Kontoeigentümer. Das bedeutet, wenn der Ressourcen-Eigentümer Zugriff auf ein AWS-Konto gewährt, kann das AWS-Konto seinen IAM-Benutzern Zugriff auf diese Ressourcen gewähren. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen IAM-Benutzer im IAM-Benutzerhandbuch. Weitere Informationen über die Angabe von Prinzipalen in einer Bucket-Richtlinie finden Sie unter Prinzipale (p. 378).	6. Juni 2011
Neuer Link	Die Endpunktinformationen dieses Service befinden sich jetzt in der allgemeinen AWS-Referenz Weitere Informationen finden Sie unter „Regions and Endpoints“ (Regionen und Endpunkte) in der Allgemeinen AWS-Referenz.	1. März 2011
Unterstützung des Hostings statischer Websites in Amazon S3	Amazon S3 führt eine erweiterte Unterstützung des Hostings statischer Websites ein. Dies beinhaltet die Unterstützung von Indexdokumenten und eines benutzerdefinierten Fehlerdokuments. Wenn Sie diese Funktionen verwenden, geben Anforderungen an das Stammverzeichnis Ihres Buckets oder eines Unterordners (z. B. `http://mywebsite.com/subfolder`) Ihr Indexdokument statt der Liste der Objekte in Ihrem Bucket zurück. Wenn ein Fehler auftritt, gibt Amazon S3 eine benutzerdefinierte Fehlermeldung anstelle einer Amazon S3-Fehlermeldung zurück. Weitere Informationen finden Sie unter Hosten einer statischen Website auf Amazon S3 (p. 602).	17. Februar 2011

Änderung	Beschreibung	Datum
Unterstützung der Response Header API	Die GET Object REST API unterstützt jetzt die Änderung der Antwort-Header der REST GET Object-Anforderung für jede Anforderung. Das bedeutet, Sie können Objekt-Metadaten in der Antwort ändern, ohne das eigentliche Objekt zu ändern. Weitere Informationen finden Sie unter Objekte abrufen (p. 184).	14. Januar 2011
Unterstützung für große Objekte	Amazon S3 hat die maximale Größe für ein Objekt, das Sie in einem S3-Bucket speichern können, von 5 GB auf 5 TB erhöht. Wenn Sie die REST API verwenden, können Sie Objekte bis zu einer Größe von 5 GB in einer einzigen PUT-Operation hochladen. Für größere Objekte müssen Sie Multipart Upload REST API zum Hochladen von Objekten in einzelnen Teilen verwenden. Weitere Informationen finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).	9. Dezember 2010
Mehrteiliger Upload	Der mehrteilige Upload unterstützt schnellere, flexiblere Uploads in Amazon S3. Mithilfe dieser Funktion können Sie ein einzelnes Objekt in mehreren Teilen hochladen. Weitere Informationen finden Sie unter Hochladen von Objekten unter Verwendung des API für mehrteilige Uploads (p. 198).	10. November 2010
Unterstützung kanonischer IDs in Bucket-Richtlinien	Sie können jetzt kanonische IDs in Bucket-Richtlinien angeben. Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in Amazon S3 (p. 376)	17. September 2010
Amazon S3 funktioniert mit IAM	Dieser Service ist jetzt in AWS Identity and Access Management (IAM) integriert. Weitere Informationen finden Sie unter AWS-Services, die mit IAM funktionieren im IAM-Benutzerhandbuch.	2. September 2010
Benachrichtigungen	Die Amazon S3-Benachrichtigungsfunktion unterstützt die Konfigurierung eines Buckets, sodass Amazon S3 eine Nachricht zu einem Amazon Simple Notification Service (Amazon SNS)-Thema veröffentlicht, wenn Amazon S3 ein Schlüsselereignis für einen Bucket erkennt. Weitere Informationen finden Sie unter Einrichten von Benachrichtigungen für Bucket-Ereignisse (p. 646).	14. Juli 2010
Bucket-Richtlinien	Bucket-Richtlinien sind ein Zugriffsmanagementsystem, mit dem Sie Zugriffsberechtigungen für Buckets, Objekte und Objektmengen festlegen. Diese Funktionalität ergänzt Zugriffsrichtlinien und ersetzt sie in vielen Fällen. Weitere Informationen finden Sie unter Verwendung von Bucket-Richtlinien und Benutzerrichtlinien (p. 375).	6. Juli 2010
Path-Style-Syntax in allen Regionen verfügbar	Amazon S3 unterstützt jetzt die Path-Style-Syntax für alle Buckets in der Region US Classic, oder wenn sich der Bucket innerhalb derselben Region befindet wie der Endpunkt der Anforderung. Weitere Informationen finden Sie unter Virtuelles Hosting (p. 48).	9. Juni 2010
Neuer Endpunkt für Europa (Irland)	Amazon S3 unterstützt jetzt einen Endpunkt für Europa (Irland): `http://s3-eu-west-1.amazonaws.com`.	9. Juni 2010

Änderung	Beschreibung	Datum
Konsole	Sie können jetzt Amazon S3 über die AWS Management Console verwenden. Informationen zur Amazon S3-Funktionalität in der Konsole finden Sie im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.	9. Juni 2010
Reduzierte Redundanz	Amazon S3 ermöglicht Ihnen jetzt die Senkung Ihrer Speicherkosten, indem Objekte in Amazon S3 mit reduzierter Redundanz gespeichert werden. Weitere Informationen finden Sie unter Reduced Redundancy Storage (p. 7).	12. Mai 2010
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region Asien-Pazifik (Singapur). Weitere Informationen finden Sie unter Buckets und Regionen (p. 60).	28. April 2010
Objekt-Versioning	In dieser Version wird ein Objekt-Versioning eingeführt. Alle Objekte können jetzt einen Schlüssel und eine Version haben. Wenn Sie das Versioning für einen Bucket aktivieren, gibt Amazon S3 allen Objekten, die einem Bucket hinzugefügt werden, eine eindeutige Versions-ID. Diese Funktion ermöglicht Ihnen eine Wiederherstellung nach einem unbeabsichtigtem Überschreiben oder Löschen. Weitere Informationen finden Sie unter Versioning (p. 8) und Verwenden von Versioning (p. 514).	8. Februar 2010
Neue unterstützte Region	Amazon S3 unterstützt jetzt die Region USA West (Nordkalifornien). Der neue Endpunkt für Anforderungen an diese Region ist s3-us-west-1.amazonaws.com. Weitere Informationen finden Sie unter Buckets und Regionen (p. 60).	2. Dezember 2009
AWS SDK für .NET	AWS stellt jetzt Bibliotheken, Beispiel-Code, Tutorials und andere Ressourcen für Softwareentwickler bereit, die es vorziehen, Anwendungen mithilfe .NET sprachspezifischer APIs statt REST oder SOAP zu erstellen. Diese Bibliotheken bieten grundlegende Funktionen (nicht in den REST oder SOAP APIs), wie beispielsweise die Anforderungsauthentifizierung, das erneute Absenden von Anforderungen und die Fehlerbehandlung, damit Ihnen der Einstieg erleichtert wird. Weitere Informationen zu sprachspezifischen Bibliotheken und Ressourcen finden Sie unter Verwenden der AWS SDKs, CLI und Explorer (p. 801).	11. November 2009

AWS-Glossar

Die aktuelle AWS-Terminologie finden Sie im AWS-Glossar im AWS General Reference.