aws

Entwicklerhandbuch

Amazon Glacier

API-Version 2012-06-01

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Glacier: Entwicklerhandbuch

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Marken und Handelsmarken von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, die geeignet ist, Kunden irrezuführen oder Amazon in irgendeiner Weise herabzusetzen oder zu diskreditieren. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

..... x
Was ist Amazon Glacier? 1
Verwenden Sie derzeit Amazon Glacier? 1
Datenmodell 3
Vault 3
Archiv 4
Aufgabe 5
Benachrichtigungskonfiguration 6
Unterstützte Vorgänge 7
Tresor-Operationen 7
Archivoperationen 7
Jobs 7
Zugriff auf Amazon Glacier 8
Regionen und Endpunkte 8
Erste Schritte 9
Schritt 1: Bevor Sie beginnen 10
Richten Sie einen ein AWS-Konto 10
Laden Sie das entsprechende SDK AWS herunter 12
Schritt 2: Erstellen eines Tresors 13
Schritt 3: Hochladen eines Archivs in einen Tresor 15
Hochladen eines Archivs mit Java 16
Hochladen eines Archivs mithilfe von .NET 21
Schritt 4: Herunterladen eines Archivs aus einem Tresor 23
Herunterladen eines Archivs mit Java 24
Herunterladen eines Archivs mit .NET 26
Schritt 5: Löschen eines Archivs aus einem Tresor 28
Zugehörige Abschnitte 29
Löschen eines Archivs mit Java 29
Löschen eines Archivs mit .NET 30
Löschen eines Archivs mit dem AWS CLI 31
Schritt 6: Löschen eines Tresors 34
Wie geht es weiter? 36
Arbeiten mit Tresoren 37
Tresorvorgänge in Amazon Glacier 38

Erstellen und Löschen von Tresoren ... 38
Abrufen von Tresormetadaten ... 38
Herunterladen eines Tresorbestands ... 39
Konfigurieren von Tresorbenachrichtigungen ... 39
Erstellen eines Tresors ... 40
Erstellen eines Tresors mit Java ... 41
Erstellen eines Tresors mit .NET ... 44
Erstellen eines Tresors mithilfe von REST ... 49
Erstellen eines Tresors mit der Konsole ... 49
Erstellen eines Tresors mit dem AWS CLI ... 49
Abrufen von Tresormetadaten ... 50
Abrufen von Tresor-Metadaten mit Java ... 51
Abrufen von Tresor-Metadaten mit .NET ... 54
Abrufen von Tresormetadaten mithilfe von REST ... 57
Abrufen von Tresor-Metadaten mit dem AWS CLI ... 57
Herunterladen eines Tresorbestands ... 58
Über das Inventar ... 60
Herunterladen einer Tresorinventur mit Java ... 61
Herunterladen eines Tresorbestands mit .NET ... 69
Herunterladen eines Tresorbestands mit REST ... 76
Herunterladen eines Tresor-Inventars mithilfe des AWS CLI ... 77
Konfigurieren von Tresorbenachrichtigungen ... 80
Allgemeine Konzepte ... 80
Konfigurieren von Tresorbenachrichtigungen mithilfe von Java ... 82
Konfigurieren von Tresorbenachrichtigungen mithilfe von .NET ... 85
Konfigurieren von Tresorbenachrichtigungen mithilfe der REST-API ... 88
Konfigurieren von Tresorbenachrichtigungen mithilfe der Konsole ... 89
Konfigurieren von Tresorbenachrichtigungen mithilfe der CLI ... 92
Löschen eines Tresors ... 93
Löschen eines Tresors mit Java ... 94
Löschen eines Tresors mithilfe von .NET ... 95
Löschen eines Tresors mithilfe von REST ... 97
Löschen eines leeren Tresors mithilfe der Konsole ... 97
Löschen eines Tresors mit dem AWS CLI ... 98
Markieren von Tresoren ... 102
Kennzeichnen von Tresoren mithilfe der Amazon Glacier-Konsole ... 102

Taggen von Tresoren mithilfe von AWS CLI ... 104
Taggen von Tresoren mithilfe der Amazon Glacier-API ... 105
Zugehörige Abschnitte ... 105
Vault Lock ... 106
Übersicht über Tresorverriegelung ... 106
Verriegeln eines Tresors mithilfe der API ... 107
Verriegeln eines Tresors mithilfe der CLI ... 109
Verriegeln eines Tresors mithilfe der Konsole ... 111
Arbeiten mit Archiven ... 114
Archivoperationen ... 115
Hochladen eines Archivs ... 115
Suchen eines Archivs ... 115
Herunterladen eines Archivs ... 115
Löschen eines Archivs ... 116
Aktualisieren eines Archivs ... 116
Pflegen von clientseitigen Archivmetadaten ... 116
Hochladen eines Archivs ... 117
Optionen zum Hochladen eines Archivs ... 117
Hochladen eines Archivs in einer einzigen Operation ... 118
Hochladen von großen Archiven in Teilen ... 129
Herunterladen eines Archivs ... 148
Abrufen von Archiven ... 148
Herunterladen eines Archivs mit Java ... 153
Herunterladen eines Archivs mit .NET ... 170
Ein großes Archiv mit Python herunterladen ... 187
Herunterladen eines Archivs mit REST ... 195
Herunterladen eines Archivs mit dem AWS CLI ... 196
Löschen eines Archivs ... 199
Löschen eines Archivs mit Java ... 200
Löschen eines Archivs mit .NET ... 202
Löschen eines Archivs mit REST ... 206
Löschen eines Archivs mit dem AWS CLI ... 206
Mit dem AWS SDKs ... 210
AWS SDK-Bibliotheken für Java und .NET ... 210
Was ist die Low-Level-&API? ... 210
Was ist die High-Level-&API? ... 211

Situationen für die Verwendung der High-Level- und der Low-Level-API ... 211
Arbeitet mit AWS SDKs ... 212
Mit dem AWS SDK für Java ... 213
Verwenden der Low-Level-API ... 213
Verwenden der High-Level-API ... 214
Ausführung von Java-Beispielen unter Verwendung von Eclipse ... 215
Festlegen des Endpunkts ... 216
Mit dem AWS SDK für .NET ... 217
Verwenden der Low-Level-API ... 217
Verwenden der High-Level-API ... 218
Ausführen der NET-Beispiele ... 219
Festlegen des Endpunkts ... 220
Codebeispiele ... 221
Grundlagen ... 222
Hallo Amazon Glacier ... 223
Aktionen ... 224
Szenarien ... 290
Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen ... 291
Abrufen von Archivinhalten und Löschen des Archivs ... 297
Sicherheit ... 303
Datenschutz ... 304
Datenverschlüsselung ... 304
Schlüsselverwaltung ... 305
Richtlinie für den Datenverkehr zwischen Netzwerken ... 305
Identitäts- und Zugriffsverwaltung ... 306
Zielgruppe ... 306
Authentifizierung mit Identitäten ... 307
Verwalten des Zugriffs mit Richtlinien ... 308
So funktioniert Amazon Glacier mit IAM ... 310
Beispiele für identitätsbasierte Richtlinien ... 317
Beispiele für eine ressourcenbasierte Richtlinie ... 324
Fehlerbehebung ... 327
Amazon Glacier API-Berechtigungsreferenz ... 329
Protokollieren und Überwachen ... 337
Compliance-Validierung ... 338
Ausfallsicherheit ... 340

Infrastruktursicherheit 341
VPC-Endpunkte 341
Richtlinien für den Datenabruf 342
Auswahl einer Amazon Glacier-Richtlinie für den Datenabruf 342
Richtlinie „Free Tier Only” 343
Richtlinie „Max Retrieval Rate” 344
Keine Richtlinie für den Abruf von Daten 344
Verwenden der Amazon Glacier-Konsole zum Einrichten einer Richtlinie für den Datenabruf ... 345
Verwenden der Amazon Glacier-API zum Einrichten einer Datenabruf-Richtlinie 345
Verwenden der Amazon Glacier REST-API zum Einrichten einer Datenabruf-Richtlinie 346
Verwenden von AWS SDKs So richten Sie eine Datenabruf-Richtlinie ein 346
-Ressourcen markieren 347
Grundlagen des Markierens 347
Tag-Einschränkungen 348
Kosten mithilfe von Tags verfolgen 348
Verwalten der Zugriffskontrolle mit Markieren 349
Zugehörige Abschnitte 349
Audit-Protokollierung mit AWS CloudTrail 350
Informationen zu Amazon Glacier in CloudTrail 350
Grundlegendes zu Amazon Glacier-Protokolldateieinträgen 351
API-Referenz 355
Häufig verwendete Anforderungsheader 356
Allgemeine Antwort-Header 360
Signieren von Anforderungen 361
Signatur-Berechnungsbeispiel 362
Berechnen von Signaturen für Streaming-Operationen 364
Berechnen von Prüfsummen 367
Struktur-Hash-Beispiel 1: Hochladen eines Archivs in einer einzelnen Anfrage 368
Struktur-Hash-Beispiel 2: Hochladen eines Archivs mit einem mehrteiligen Upload 369
Berechnen des Struktur-Hashs einer Datei 370
Erhalten von Prüfsummen bei Daten-Download 380
Fehlermeldungen 382
Beispiel 1: Beschreiben einer Auftragsanfrage mit einer Auftrags-ID, die nicht existiert 386
Beispiel 2: Anfrage zum Auflisten von Aufträgen mit einem ungültigen Wert für den Anfrageparameter 387
Tresor-Operationen 388

Abort Vault Lock ... 389
Add Tags To Vault ... 392
Create Vault ... 396
Complete Vault Lock ... 399
Delete Vault ... 402
Delete Vault Access Policy ... 405
Delete Vault Notifications ... 408
Describe Vault ... 410
Get Vault Access Policy ... 415
Get Vault Lock ... 418
Get Vault Notifications ... 423
Initiate Vault Lock ... 427
List Tags For Vault ... 431
List Vaults ... 435
Remove Tags From Vault ... 442
Set Vault Access Policy ... 445
Set Vault Notification Configuration ... 449
Archivoperationen ... 453
Delete Archive ... 453
Upload Archive ... 456
Operationen für mehrteiligen Upload ... 462
Abort Multipart Upload ... 463
Abschließen eines mehrteiligen Uploads ... 466
Initiieren eines mehrteiligen Uploads ... 471
Teile auflisten ... 476
List Multipart Uploads ... 484
Upload Part ... 491
Auftragsoperationen ... 498
Describe Job ... 498
Get Job Output ... 509
Initiate Job ... 520
Aufträge auflisten ... 532
In Auftragsoperationen verwendete Datentypen ... 542
CSVInput ... 543
CSVOutput ... 544
Verschlüsselung ... 545

GlacierJobDescription 546
Gewährung 550
Grantee 551
InputSerialization 552
InventoryRetrievalJobInput 552
jobParameters 554
OutputLocation 557
OutputSerialization 557
S3Location 558
SelectParameters 560
Datenabrufoperationen 561
Get Data Retrieval Policy 561
List Provision Capacity 565
Purchase Provisioned Capacity 569
Set Data Retrieval Policy 572
Dokumentverlauf 578
Frühere Aktualisierungen 579
AWS Glossar 582

Diese Seite ist nur für Bestandskunden des Amazon Glacier-Service bestimmt, die Vaults und die ursprüngliche REST-API von 2012 verwenden.

Wenn Sie nach Archivspeicherlösungen suchen, empfehlen wir die Verwendung der Amazon Glacier-Speicherklassen in Amazon S3, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive. Weitere Informationen zu diesen Speicheroptionen finden Sie unter Amazon Glacier-Speicherklassen.

Amazon Glacier (ursprünglicher eigenständiger, auf Tresor basierender Service) akzeptiert keine Neukunden mehr. Amazon Glacier ist ein eigenständiger Service APIs , der Daten in Tresoren speichert und sich von den Speicherklassen Amazon S3 und Amazon S3 Glacier unterscheidet. Ihre vorhandenen Daten bleiben in Amazon Glacier auf unbestimmte Zeit sicher und zugänglich. Es ist keine Migration erforderlich. Für kostengünstige, langfristige Archivierungsspeicherung AWS empfiehlt sich die Amazon S3 Glacier-Speicherklasse, die mit S3-Bucket-Basis, voller AWS-Region Verfügbarkeit APIs, geringeren Kosten und AWS Serviceintegration ein hervorragendes Kundenerlebnis bieten. Wenn Sie erweiterte Funktionen wünschen, sollten Sie eine Migration zu Amazon S3 Glacier-Speicherklassen in Betracht ziehen, indem Sie unseren AWS Lösungsleitfaden für die Übertragung von Daten aus Amazon Glacier-Tresoren in Amazon S3 Glacier-Speicherklassen verwenden.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.

Was ist Amazon Glacier?

Wenn Sie derzeit den Amazon Glacier-Service nutzen und mehr erfahren möchten, finden Sie die Informationen, die Sie benötigen, in diesem Handbuch. Amazon Glacier ist ein sicherer und langlebiger Service für kostengünstige Datenarchivierung und langfristige Sicherung mithilfe von Tresoren. Weitere Informationen zu den Amazon Glacier-Servicepreisen finden Sie unter Amazon Glacier-Preise.

Themen

- Verwenden Sie derzeit Amazon Glacier?
- Amazon Glacier-Datenmodell
- Unterstützte Vorgänge in Amazon Glacier
- Zugriff auf Amazon Glacier

Verwenden Sie derzeit Amazon Glacier?

> **Note**
>
> In diesem Abschnitt geht es um den Amazon Glacier-Service. Wenn Sie derzeit die Amazon S3 Glacier-Speicherklassen (S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive) verwenden, finden Sie weitere Informationen unter Speicherklassen für die Archivierung von Objekten im Amazon S3 S3-Benutzerhandbuch.

Wenn Sie derzeit den Amazon Glacier-Service nutzen und mehr erfahren möchten, empfehlen wir Ihnen, zunächst die folgenden Abschnitte zu lesen:

- Was ist Amazon Glacier — Der Rest dieses Abschnitts beschreibt das zugrunde liegende Datenmodell, die unterstützten Operationen und die, AWS SDKs die Sie für die Interaktion mit dem Service verwenden können.
- Erste Schritte – Im Abschnitt Erste Schritte mit Amazon Glacier erfahren Sie, wie Sie einen Tresor erstellen, Archive hochladen, Aufträge zum Herunterladen von Archiven erstellen, die Auftragsausgabe abrufen und Archive löschen.

> **Important**
>
> Amazon Glacier bietet eine Konsole. Für jeden Archivierungsvorgang, wie Hochladen, Herunterladen oder Löschen, müssen Sie jedoch AWS Command Line Interface (AWS CLI) verwenden oder Code schreiben. Es steht keine Konsolenunterstützung für Archivvorgänge zur Verfügung. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über AWS SDKs. Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie in der AWS CLI Referenz für Amazon Glacier. Beispiele für die Verwendung von AWS CLI zum Hochladen von Archiven auf Amazon Glacier finden Sie unter Verwenden von Amazon Glacier mit dem AWS Command Line Interface.

Neben dem Abschnitt Erste Schritte möchten Sie wahrscheinlich mehr über den Betrieb von Amazon Glacier erfahren. Die folgenden Abschnitte enthalten detaillierte Informationen zur Arbeit mit Amazon Glacier mithilfe der REST-API und der AWS SDKs für Java und Microsoft.NET:

- Verwenden von AWS SDKs mit Amazon Glacier

  Dieser Abschnitt bietet einen Überblick über die in verschiedenen Codebeispielen in diesem Handbuch AWS SDKs verwendeten. Die Kenntnis dieses Abschnitts wird beim Lesen der folgenden Abschnitte hilfreich sein. Er enthält einen Überblick über die High-Level- und APIs Low-Level-Funktionen, deren Einsatzmöglichkeiten und allgemeine Schritte zur Ausführung der in diesem Handbuch enthaltenen Codebeispiele. SDKs

- Arbeiten mit Tresoren in Amazon Glacier

  Dieser Abschnitt enthält Details zu verschiedenen Tresor-Operationen, z. B. das Erstellen eines Tresors, das Abrufen von Tresormetadaten, das Abrufen von Tresorbeständen mithilfe von Aufträgen und das Konfigurieren von Tresorbenachrichtigungen. Zusätzlich zur Verwendung der Amazon Glacier-Konsole können Sie die AWS SDKs für verschiedene Tresorvorgänge verwenden. In diesem Abschnitt wird die API beschrieben und Arbeitsbeispiele mithilfe von AWS SDK für Java und der bereitgestellt AWS SDK für .NET.

- Arbeiten mit Archiven in Amazon Glacier

Dieser Abschnitt enthält Details zu Archivoperationen, z. B. das Hochladen eines Archivs in einer einzigen Anforderung oder das Hochladen großer Archive in Teilen mithilfe eines mehrteiligen Uploads. In diesem Abschnitt wird auch das Erstellen von Aufträgen zum asynchronen Download von Archiven erläutert. Der Abschnitt liefert Beispiele, die das AWS SDK für Java und AWS SDK für .NET verwenden.

- API-Referenz für Amazon Glacier

  Amazon Glacier ist ein RESTful Service. Dieser Abschnitt beschreibt die REST-Operationen, einschließlich der Syntax, und Beispielanfragen und -antworten für alle Operationen. Die AWS SDK-Bibliotheken umfassen diese API und vereinfachen so Ihre Programmieraufgaben.

Amazon Glacier-Datenmodell

Zu den Kernkomponenten des Amazon Glacier-Datenmodells gehören Tresore und Archive. Amazon Glacier ist ein REST-basierter Webservice. Im Hinblick auf REST sind Tresore und Archive die Ressourcen. Darüber hinaus umfasst das Amazon Glacier-Datenmodell Ressourcen zur Auftrags- und Benachrichtigungskonfiguration. Diese Ressourcen vervollständigen die Kernressourcen.

Themen

- Vault
- Archiv
- Aufgabe
- Benachrichtigungskonfiguration

Vault

In Amazon Glacier ist ein Tresor ein Container zum Speichern von Archiven. Ein Tresor ist mit einem Amazon-S3-Bucket vergleichbar. Wenn Sie einen Tresor erstellen, geben Sie einen Namen an und wählen einen AWS-Region Ort aus, an dem Sie den Tresor erstellen möchten.

Jede Tresor-Ressource verfügt über eine eindeutige Adresse. Die allgemeine Form ist:

```
https://region-specific-endpoint/account-id/vaults/vault-name
```

Ein Beispiel: Sie erstellen in Ihrem Konto mit der ID 111122223333 einen Tresor (`examplevault`) in der Region „USA West (Oregon)“. Dieser Tresor kann dann mit dem folgenden URI adressiert werden:

```
https://glacier.us-west-2.amazonaws.com/111122223333/vaults/examplevault
```

Die verschiedenen Komponenten des URI bedeuten Folgendes:

- `glacier.us-west-2.amazonaws.com` identifiziert die Region „USA West (Oregon)“.
- `111122223333`ist die AWS-Konto ID, der der Tresor gehört.
- `vaults` bezieht sich auf die Sammlung von Tresoren, die zu dem AWS-Konto gehören.
- identifiziert `examplevault` einen bestimmten Tresor in der Tresorsammlung.

Und AWS-Konto kann Tresore in jedem unterstützten AWS-Region Format erstellen. Eine Liste der unterstützten Programme finden Sie AWS-Regionen unterZugriff auf Amazon Glacier. Innerhalb einer Region muss ein Konto eindeutige Tresornamen verwenden. An AWS-Konto kann Tresore mit demselben Namen in verschiedenen Regionen erstellen.

Sie können eine unbegrenzte Anzahl von Archiven in einem Tresor speichern. Je nach den Anforderungen Ihres Unternehmens oder Ihrer Anwendung können Sie diese Archive in einem Tresor oder in mehreren Tresoren speichern.

Amazon Glacier unterstützt verschiedene Tresoroperationen. Tresor-Operationen sind regionsspezifisch. Wenn Sie beispielsweise einen Tresor erstellen, so erstellen Sie ihn in einer spezifischen Region. Wenn Sie eine Tresorliste anfordern, fordern Sie sie von einer bestimmten Region an. Die resultierende Liste enthält nur Tresore AWS-Region, die in dieser bestimmten Region erstellt wurden.

Archiv

Bei einem Archiv kann es sich um beliebige Daten handeln, z. B: Fotos, Videos oder Dokumente. Ein Archiv ähnelt einem Amazon S3 S3-Objekt und ist die Basisspeichereinheit in Amazon Glacier. Jedes Archiv besitzt eine eindeutige ID und optional eine Beschreibung. Sie können diese optionale Beschreibung nur beim Hochladen eines Archivs angeben. Amazon Glacier weist dem Archiv eine ID zu, die in der Art, AWS-Region in der das Archiv gespeichert ist, eindeutig ist.

Jedes Archiv hat eine eindeutige Adresse. Die grundlegende Form sieht wie folgt aus:

```
https://region-specific-endpoint/account-id/vaults/vault-name/archives/archive-id
```

Folgender URI ist ein Beispiel-URI eines Archivs, das im Tresor `examplevault` in der Region „USA West (Oregon)“ im Konto 111122223333 gespeichert ist:

```
https://glacier.us-west-2.amazonaws.com/111122223333/vaults/
examplevault/archives/NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
```

Sie können eine unbegrenzte Anzahl von Archiven in einem Tresor speichern.

Aufgabe

Ein Amazon Glacier-Auftrag kann ein Archiv abrufen oder ein Inventar eines Tresors abrufen.

Das Abrufen von Archiven und Tresorinventaren (Archivlisten) ist ein asynchroner Vorgang in Amazon Glacier, bei dem Sie zuerst einen Auftrag initiieren und dann die Auftragsausgabe herunterladen, nachdem Amazon Glacier den Auftrag abgeschlossen hat.

> **Note**
>
> Amazon Glacier bietet eine Lösung zur Archivierung von Daten im Kühlraum. Wenn Ihre Anwendung eine Speicherlösung benötigt, die Datenabruf in Echtzeit erfordert, sollten Sie die Nutzung von Amazon S3 in Betracht ziehen. Weitere Infrmationen finden Sie unter Amazon Simple Storage Service (Amazon S3).

Wenn Sie einen Auftrag zur Tresorinventur initiieren möchten, geben Sie einen Tresornamen an. Für Archivabrufaufträge müssen Sie sowohl den Tresornamen als auch die Archiv-ID angeben. Sie können optional auch eine Beschreibung des Auftrags angeben, um diesen einfacher identifizieren zu können.

Aufträge zum Abrufen von Archiven und Aufträge zur Tresorinventur sind einem Tresor zugeordnet. Ein Tresor kann jederzeit mehrere Aufträge in Bearbeitung haben. Wenn Sie eine Jobanfrage senden (einen Job initiieren), gibt Amazon Glacier Ihnen eine Job-ID zurück, mit der Sie den Job verfolgen können. Jeder Auftrag wird eindeutig identifiziert durch einen URI der Form:

```
https://region-specific-endpoint/account-id/vaults/vault-name/jobs/job-id
```

Nachfolgend finden Sie ein Beispiel für einen Auftrag im Zusammenhang mit dem Tresor `examplevault` in der Region „USA West (Oregon)“ im Konto „111122223333“.

```
https://glacier.us-west-2.amazonaws.com/111122223333/vaults/examplevault/jobs/
HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

Für jeden Auftrag verwaltet Amazon Glacier Informationen wie Auftragstyp, Beschreibung, Erstellungsdatum, Abschlussdatum und Auftragsstatus. Sie können Informationen über einen bestimmten Auftrag oder eine Liste aller Aufträge, die einem Tresor zugeordnet sind, abrufen. Die Liste der Jobs, die Amazon Glacier zurückgibt, umfasst alle laufenden und kürzlich abgeschlossenen Jobs.

Benachrichtigungskonfiguration

Da die Ausführung von Aufträgen einige Zeit in Anspruch nimmt, unterstützt Amazon Glacier einen Benachrichtigungsmechanismus, der Sie benachrichtigt, wenn ein Auftrag abgeschlossen ist. Sie können einen Tresor so konfigurieren, dass er eine Benachrichtigung an ein Amazon Simple Notification Service (Amazon SNS)-Thema sendet, wenn ein Auftrag abgeschlossen wurde. In der Benachrichtigungskonfiguration können Sie ein Amazon-SNS-Thema pro Tresor angeben.

Amazon Glacier speichert die Benachrichtigungskonfiguration als JSON-Dokument. Es folgt ein Beispiel für eine Tresorbenachrichtigungskonfiguration:

```
{
   "Topic": "arn:aws:sns:us-west-2:111122223333:mytopic",
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

Benachrichtigungskonfigurationen sind Tresoren zugeordnet. Sie können eine für jeden Tresor festlegen. Jede Benachrichtigungskonfigurations-Ressource wird eindeutig identifiziert durch einen URI der Form:

```
https://region-specific-endpoint/account-id/vaults/vault-name/notification-
configuration
```

Amazon Glacier unterstützt Vorgänge zum Einrichten, Abrufen und Löschen einer Benachrichtigungskonfiguration. Wenn Sie eine Benachrichtigungskonfiguration löschen, werden

keine Benachrichtigungen gesendet, wenn eine Datenabruf-Operation für den Tresor abgeschlossen wurde.

Unterstützte Vorgänge in Amazon Glacier

Für die Arbeit mit Tresoren und Archiven (siheAmazon Glacier-Datenmodell) unterstützt Amazon Glacier eine Reihe von Vorgängen. Von allen unterstützten Operationen sind nur folgende asynchron:

- Abrufen eines Archivs
- Abrufen eines Tresorbestands (Liste von Archiven)

Diese Operationen erfordern, dass Sie zuerst einen Auftrag initiieren und dann die Auftragsausgabe herunterladen. In den folgenden Abschnitten werden die Abläufe von Amazon Glacier zusammengefasst.

Tresor-Operationen

Amazon Glacier bietet Funktionen zum Erstellen und Löschen von Tresoren. Sie können eine Tresorbeschreibung für einen bestimmten Tresor oder für alle Tresore in einer AWS-Region abrufen. Die Tresorbeschreibung enthält Informationen wie das Erstellungsdatum, die Anzahl der Archive im Tresor, die Gesamtgröße in Byte, die von allen Archiven im Tresor verwendet wird, und das Datum, an dem Amazon Glacier das Tresorinventar generiert hat. Amazon Glacier bietet auch Funktionen zum Einrichten, Abrufen und Löschen einer Benachrichtigungskonfiguration im Tresor. Weitere Informationen finden Sie unter Arbeiten mit Tresoren in Amazon Glacier.

Archivoperationen

Amazon Glacier bietet Funktionen zum Hochladen und Löschen von Archiven. Sie können ein bestehendes Archiv nicht aktualisieren, Sie müssen das bestehende Archiv löschen und ein neues Archiv hochladen. Jedes Mal, wenn Sie ein Archiv hochladen, generiert Amazon Glacier eine neue Archiv-ID. Weitere Informationen finden Sie unter Arbeiten mit Archiven in Amazon Glacier.

Jobs

Sie können einen Amazon Glacier-Auftrag initiieren, um ein Archiv abzurufen oder ein Inventar eines Tresors abzurufen.

Im Folgenden sind die Arten von Amazon Glacier-Aufträgen aufgeführt:

- `archive-retrieval`: Ein Archiv abrufen.

  Weitere Informationen finden Sie unter Ein Archiv in Amazon Glacier herunterladen.
- `inventory-retrieval`: Bestand eines Tresors erfassen.

  Weitere Informationen finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

Zugriff auf Amazon Glacier

Amazon Glacier ist ein RESTful Webservice, der HTTP und HTTPS als Transportprotokoll und JavaScript Object Notation (JSON) als Nachrichtenserialisierungsformat verwendet. Ihr Anwendungscode kann Anfragen direkt an die Amazon Glacier Webservice-API stellen. Bei direkter Verwendung der REST-API müssen Sie den benötigten Code zum Signieren und Authentifizieren der Anforderungen schreiben. Weitere Informationen zur API finden Sie unter API-Referenz für Amazon Glacier.

Alternativ können Sie die Anwendungsentwicklung vereinfachen, indem Sie AWS SDKs die REST-API-Aufrufe von Amazon Glacier verwenden. Sie geben Ihre Anmeldeinformationen an und diese Bibliotheken übernehmen die Authentifizierung und signieren die Anforderungen. Weitere Informationen zur Verwendung der AWS SDKs finden Sie unter Verwenden von AWS SDKs mit Amazon Glacier.

Amazon Glacier bietet auch eine Konsole. Für alle Archivierungs- und Auftragsvorgänge müssen Sie jedoch Code schreiben und Anfragen stellen, indem Sie entweder direkt die REST-API oder die AWS SDK-Wrapper-Bibliotheken verwenden. Um auf die Amazon Glacier-Konsole zuzugreifen, gehen Sie zu Amazon Glacier Console.

Regionen und Endpunkte

Sie erstellen einen Tresor in einem bestimmten AWS-Region. Sie senden Ihre Amazon Glacier-Anfragen immer an einen Endpunkt, der für einen spezifisch ist AWS-Region. Eine Liste der von Amazon Glacier AWS-Regionen unterstützten Endpunkte und Kontingente finden Sie unter Amazon Glacier-Endpunkte und Kontingente in der AWS Allgemeinen Referenz.

Erste Schritte mit Amazon Glacier

Sie können mit Amazon Glacier (Amazon Glacier) beginnen, indem Sie mit Tresoren und Archiven arbeiten. Ein Tresor ist ein Container, in dem Archive gespeichert werden, und ein Archiv ist jedes Objekt (z. B. ein Foto, Video oder Dokument), das Sie in einem Tresor speichern. Die Basiseinheit des Speichers in Amazon Glacier ist ein Archiv. In dieser Übung „Erste Schritte“ erfahren Sie, wie Sie sich mit den grundlegenden Amazon Glacier-Vorgängen in Tresoren und Archiven vertraut machen. Weitere Informationen zu diesen Ressourcen finden Sie im Abschnitt Amazon Glacier-Datenmodell.

Die ersten Schritte bestehen darin, dass Sie einen Tresor erstellen, ein Archiv hochladen, es herunterladen und dann das Archiv und den Tresor löschen. Alle diese Vorgänge können programmatisch ausgeführt werden. In der Übung „Erste Schritte“ wird jedoch die Amazon Glacier-Managementkonsole verwendet, um einen Tresor zu erstellen und zu löschen. Für das Hoch- und Herunterladen eines Archivs verwendet dieser Abschnitt „Erste Schritte“ die High-Level-API für AWS SDK für Java und die AWS SDK für .NET. Die High-Level-API bietet eine vereinfachte Programmiererfahrung bei der Arbeit mit Amazon Glacier. Weitere Informationen zur Verwendung der High-Level-API mit dem AWS SDKs finden Sie unterVerwenden von AWS SDKs mit Amazon Glacier.

> **Important**
>
> Amazon Glacier bietet eine Konsole. Für jeden Archivierungsvorgang, wie Hochladen, Herunterladen oder Löschen, müssen Sie jedoch die AWS Command Line Interface (CLI) verwenden oder Code schreiben. Es steht keine Konsolenunterstützung für Archivvorgänge zur Verfügung. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, indem Sie entweder die REST-API direkt verwenden oder indem Sie die AWS SDKs.
>
> Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie in der AWS CLI Referenz für Amazon Glacier. Beispiele für die Verwendung von AWS CLI zum Hochladen von Archiven auf Amazon Glacier finden Sie unter Verwenden von Amazon Glacier mit dem AWS Command Line Interface.

Die Übung „Erste Schritte“ enthält Codebeispiele in Java und C#, die Sie zum Hochladen und Herunterladen eines Archivs verwenden können. Der letzte Abschnitt der Übung „Erste Schritte“

enthält Schritte, in denen Sie mehr über die Entwicklererfahrung mit Amazon Glacier erfahren können.

Themen

- Schritt 1: Bevor Sie mit Amazon Glacier beginnen
- Schritt 2: Einen Tresor in Amazon Glacier erstellen
- Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch
- Schritt 4: Laden Sie ein Archiv aus einem Tresor in Amazon Glacier herunter
- Schritt 5: Löschen eines Archivs aus einem Tresor in Amazon Glacier
- Schritt 6: Löschen eines Tresors in Amazon Glacier
- Wie geht es weiter?

Schritt 1: Bevor Sie mit Amazon Glacier beginnen

Bevor Sie mit dieser Übung beginnen können, müssen Sie sich für eine anmelden AWS-Konto (falls Sie noch keine haben) und dann eine davon herunterladen AWS SDKs. Detaillierte Informationen finden Sie in den folgenden Abschnitten.

Themen

- Richten Sie einen Benutzer AWS-Konto und einen Administratorbenutzer ein
- Laden Sie das entsprechende SDK AWS herunter

Richten Sie einen Benutzer AWS-Konto und einen Administratorbenutzer ein

Falls Sie dies noch nicht getan haben, müssen Sie sich für ein Konto registrieren AWS-Konto und einen Administratorbenutzer für das Konto erstellen.

Folgen Sie den Anweisungen in den folgenden Themen, um die Einrichtung abzuschließen.

Richten Sie einen Administratorbenutzer ein AWS-Konto und erstellen Sie einen

Melden Sie sich an für AWS

Wenn Sie sich für Amazon Web Services (AWS) registrieren, AWS-Konto ist Ihr Konto automatisch für alle Services angemeldet AWS, einschließlich Amazon Glacier. Berechnet werden Ihnen aber nur

die Services, die Sie nutzen. Weitere Informationen zu den Amazon Glacier-Nutzungstarifen finden Sie auf der Seite mit den Amazon Glacier-Preisen.

Wenn Sie bereits eine haben AWS-Konto, fahren Sie mit fortLaden Sie das entsprechende SDK AWS herunter. Wenn Sie noch keine haben AWS-Konto, gehen Sie wie folgt vor, um eine zu erstellen.

Wenn Sie noch keine haben AWS-Konto, führen Sie die folgenden Schritte aus, um eine zu erstellen.

Um sich für eine anzumelden AWS-Konto

1. Öffnen Sie https://portal.aws.amazon.com/billing/die Anmeldung.
2. Folgen Sie den Online-Anweisungen.

   Während der Anmeldung erhalten Sie einen Telefonanruf oder eine Textnachricht und müssen einen Verifizierungscode über die Telefontasten eingeben.

   Wenn Sie sich für eine anmelden AWS-Konto, Root-Benutzer des AWS-Kontoswird eine erstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Als bewährte Sicherheitsmethode weisen Sie einem Benutzer Administratorzugriff zu und verwenden Sie nur den Root-Benutzer, um Aufgaben auszuführen, die Root-Benutzerzugriff erfordern.

Wählen Sie zum Erstellen eines Administratorbenutzers eine der folgenden Optionen aus.

Wählen Sie eine Möglichkeit zur Verwaltung Ihres Administrators aus.	Bis	Von	Sie können auch
Im IAM Identity Center	Verwendung von kurzfristigen Anmeldeinformationen für den Zugriff auf AWS.	Beachtung der Anweisungen unter Erste Schritte im AWS IAM Identity Center - Benutzerhandbuch.	Konfigurieren Sie den programmatischen Zugriff, indem Sie AWS CLI die Konfiguration für die Verwendung AWS

Wählen Sie eine Möglichkeit zur Verwaltung Ihres Administrators aus.	Bis	Von	Sie können auch
(Empfohlen)	Dies steht im Einklang mit den bewährten Methoden für die Sicherheit. Weitere Informationen zu bewährten Methoden finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.		IAM Identity Center im AWS Command Line Interface Benutzerhandbuch vornehmen.
In IAM (Nicht empfohlen)	Verwendung von langfristigen Anmeldeinformationen für den Zugriff auf AWS.	Folgen Sie den Anleitungen unter IAM-Benutzer für den Notfallzugriff erstellen im IAM-Benutzerhandbuch.	Sie konfigurieren den programmgesteuerten Zugriff unter Verwendung der Informationen unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im IAM-Benutzerhandbuch.

Laden Sie das entsprechende SDK AWS herunter

Um die Übung „Erste Schritte“ auszuprobieren, müssen Sie entscheiden, welche Programmiersprache Sie verwenden möchten, und dann das entsprechende AWS SDK für Ihre Entwicklungsplattform herunterladen.

Die Übung „Erste Schritte“ enthält Beispiele in Java und C#.

Das wird heruntergeladen AWS SDK für Java

Sie benötigen AWS SDK für Java, um die Java-Beispiele in diesem Entwicklerhandbuch zu testen. Für das Herunterladen stehen Ihnen die folgenden Optionen zur Verfügung:

- Wenn Sie Eclipse verwenden, können Sie das über die AWS Toolkit for Eclipse Update-Site http://aws.amazon.com/eclipse/ herunterladen und installieren. Weitere Informationen finden Sie unter AWS Toolkit for Eclipse.
- Wenn Sie eine andere IDE zum Erstellen Ihrer Anwendung verwenden, laden Sie AWS SDK für Java herunter.

Das wird heruntergeladen AWS SDK für .NET

Sie benötigen AWS SDK für .NET, um die C#-Beispiele in diesem Entwicklerhandbuch zu testen. Für das Herunterladen stehen Ihnen die folgenden Optionen zur Verfügung:

- Wenn Sie Visual Studio verwenden, können Sie AWS SDK für .NET sowohl den als auch den installieren AWS Toolkit for Visual Studio. Das Toolkit bietet AWS Explorer für Visual Studio und Projektvorlagen, die Sie für die Entwicklung verwenden können. Um das herunterzuladen AWS SDK für .NET, gehen Sie zu http://aws.amazon.com/sdkfornet. Standardmäßig installiert das Installationsskript sowohl das AWS SDK als auch das AWS Toolkit for Visual Studio. Weitere Informationen zum Toolkit finden Sie im Benutzerhandbuch für AWS Toolkit for Visual Studio.
- Wenn Sie eine andere IDE zum Erstellen Ihrer Anwendung nutzen, können Sie den für den vorherigen Schritt bereitgestellten Link verwenden und nur AWS SDK für .NET installieren.

Schritt 2: Einen Tresor in Amazon Glacier erstellen

Ein Tresor ist ein Container für das Speichern von Archiven. Ihr erster Schritt besteht darin, einen Tresor in einem der unterstützten Systeme zu erstellen AWS-Regionen. Eine Liste der AWS-Regionen von Amazon Glacier unterstützten Endpunkte und Kontingente finden Sie unter Amazon Glacier-Endpunkte und Kontingente in der AWS Allgemeinen Referenz.

Sie können Tresore programmgesteuert oder mithilfe der Amazon Glacier-Konsole erstellen. In diesem Abschnitt erstellen wir einen Tresor über die Konsole.

So erstellen Sie einen Tresor

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie im linken Navigationsbereich Tresore aus.
3. Wählen Sie Tresor erstellen aus.

   Die Seite Tresor erstellen wird geöffnet.
4. Wählen Sie unter Region auswählen eine Region AWS-Region aus der Regionsauswahl aus. Ihr Tresor wird in der ausgewählten Region erstellt.
5. Geben Sie unter Tresorname einen Namen für Ihren Tresor ein.

   Folgende Anforderungen müssen Sie beim Benennen von Tresoren beachten:

   - Ein Tresorname muss innerhalb des AWS-Konto und des Tresors, AWS-Region in dem der Tresor erstellt wurde, eindeutig sein.
   - Tresornamen müssen zwischen 1 und 255 Zeichen lang sein.
   - Ein Tresorname darf nur die folgenden Zeichen enthalten: a–z, A–Z, 0–9, _ (Unterstrich), - (Bindestrich) und . (Punkt).
6. Wählen Sie unter Ereignis-Benachrichtigungen eine der folgenden Einstellungen aus, um für einen Tresor Benachrichtigungen bei Abschluss eines Auftrags zu aktivieren oder zu deaktivieren:

   - Benachrichtigungen deaktivieren: Benachrichtigungen sind deaktiviert und es werden bei Abschluss eines Auftrags keine Benachrichtigungen an ein Amazon Simple Notification Service (Amazon SNS)-Thema gesendet.

   - Benachrichtigungen aktivieren: Benachrichtigungen sind aktiviert und bei Abschluss eines bestimmten Auftrags werden Benachrichtigungen an das angegebene Amazon-SNS-Thema gesendet.

     Wenn Sie Benachrichtigungen aktivieren ausgewählt haben, finden Sie weitere Informationen unter Konfiguration von Vault-Benachrichtigungen mithilfe der Amazon Glacier-Konsole.
7. Wenn der AWS-Region und der Tresorname korrekt sind, wählen Sie „Tresor erstellen“.

Ihr neuer Tresor ist jetzt auf der Seite Tresore in der Amazon Glacier-Konsole aufgeführt.

Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch

In diesem Schritt laden Sie ein Beispielarchiv in den Tresor hoch, den Sie im vorherigen Schritt erstellt haben (siehe Schritt 2: Einen Tresor in Amazon Glacier erstellen). Klicken Sie am Ende dieses Abschnitts auf den Link für die verwendete Entwicklungsplattform.

> **Important**
>
> Für alle Archivvorgänge (Hochladen, Herunterladen oder Löschen) müssen Sie die AWS Command Line Interface (CLI) verwenden oder Code schreiben. Es steht keine Konsolenunterstützung für Archivvorgänge zur Verfügung. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über die AWS SDKs.
> Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie unter AWS CLI Referenz für Amazon Glacier. Beispiele für die Verwendung von AWS CLI zum Hochladen von Archiven auf Amazon Glacier finden Sie unter Verwenden von Amazon Glacier mit dem AWS Command Line Interface.

Als Archiv wird jedes beliebige Objekt bezeichnet, wie z. B. ein Foto, Video oder Dokument, das Sie in einem Tresor speichern. Die Basiseinheit des Speichers in Amazon Glacier ist ein Archiv. Sie können ein Archiv mittels einer einzigen Anforderung hochladen. Für große Archive bietet Amazon Glacier einen mehrteiligen Upload-API-Vorgang, mit dem Sie ein Archiv in Teilen hochladen können.

In diesem einführenden Abschnitt laden Sie ein Musterarchiv mittels einer einzigen Anforderung hoch. Bei dieser Übung geben Sie eine Datei kleineren Umfangs an. Für größere Dateien ist ein mehrteiliger Upload angemessen. Weitere Informationen finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

Themen

- Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch, indem Sie AWS SDK für Java
- Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch, indem Sie AWS SDK für .NET

Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch, indem Sie AWS SDK für Java

Das folgende Java-Codebeispiel verwendet die High-Level-API von AWS SDK für Java , um ein Beispielarchiv in den Tresor hochzuladen. Beachten Sie im Codebeispiel Folgendes:

- Im Beispiel wird eine Instance der `AmazonGlacierClient`-Klasse erstellt.
- Das Beispiel verwendet den API-Vorgang `upload` der `ArchiveTransferManager`-Klasse aus der High-Level-API des AWS SDK für Java.
- In diesem Beispiel wird die Region „USA West (Oregon)“ (`us-west-2`) verwendet.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt mit dem Namen des hochzuladenden Archivs aktualisieren.

> **Note**
>
> Amazon Glacier führt ein Inventar aller Archive in Ihren Tresoren. Wenn Sie das Archiv im folgenden Beispiel hochladen, wird es auf der Managementkonsole erst nach der Aktualisierung des Tresorbestands im Tresor angezeigt. Diese Aktualisierung wird in der Regel einmal täglich durchgeführt.

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.UploadArchiveRequest;
import software.amazon.awssdk.services.glacier.model.UploadArchiveResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import java.io.File;
import java.nio.file.Path;
```

```
import java.nio.file.Paths;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class UploadArchive {

    static final int ONE_MB = 1024 * 1024;

    public static void main(String[] args) {
        final String usage = """

                Usage:   <strPath> <vaultName>\s

                Where:
                   strPath - The path to the archive to upload (for example, C:\\AWS
\\test.pdf).
                   vaultName - The name of the vault.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String strPath = args[0];
        String vaultName = args[1];
        File myFile = new File(strPath);
        Path path = Paths.get(strPath);
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String archiveId = uploadContent(glacier, path, vaultName, myFile);
        System.out.println("The ID of the archived item is " + archiveId);
```

```
        glacier.close();
    }

    public static String uploadContent(GlacierClient glacier, Path path, String
vaultName, File myFile) {
        // Get an SHA-256 tree hash value.
        String checkVal = computeSHA256(myFile);
        try {
            UploadArchiveRequest uploadRequest = UploadArchiveRequest.builder()
                    .vaultName(vaultName)
                    .checksum(checkVal)
                    .build();

            UploadArchiveResponse res = glacier.uploadArchive(uploadRequest, path);
            return res.archiveId();

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    private static String computeSHA256(File inputFile) {
        try {
            byte[] treeHash = computeSHA256TreeHash(inputFile);
            System.out.printf("SHA-256 tree hash = %s\n", toHex(treeHash));
            return toHex(treeHash);

        } catch (IOException ioe) {
            System.err.format("Exception when reading from file %s: %s", inputFile,
ioe.getMessage());
            System.exit(-1);

        } catch (NoSuchAlgorithmException nsae) {
            System.err.format("Cannot locate MessageDigest algorithm for SHA-256:
%s", nsae.getMessage());
            System.exit(-1);
        }
        return "";
    }

    public static byte[] computeSHA256TreeHash(File inputFile) throws IOException,
            NoSuchAlgorithmException {
```

```
        byte[][] chunkSHA256Hashes = getChunkSHA256Hashes(inputFile);
        return computeSHA256TreeHash(chunkSHA256Hashes);
    }

    /**
     * Computes an SHA256 checksum for each 1 MB chunk of the input file. This
     * includes the checksum for the last chunk, even if it's smaller than 1 MB.
     */
    public static byte[][] getChunkSHA256Hashes(File file) throws IOException,
            NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        long numChunks = file.length() / ONE_MB;
        if (file.length() % ONE_MB > 0) {
            numChunks++;
        }

        if (numChunks == 0) {
            return new byte[][] { md.digest() };
        }

        byte[][] chunkSHA256Hashes = new byte[(int) numChunks][];
        FileInputStream fileStream = null;

        try {
            fileStream = new FileInputStream(file);
            byte[] buff = new byte[ONE_MB];

            int bytesRead;
            int idx = 0;

            while ((bytesRead = fileStream.read(buff, 0, ONE_MB)) > 0) {
                md.reset();
                md.update(buff, 0, bytesRead);
                chunkSHA256Hashes[idx++] = md.digest();
            }

            return chunkSHA256Hashes;

        } finally {
            if (fileStream != null) {
                try {
                    fileStream.close();
```

```
                } catch (IOException ioe) {
                    System.err.printf("Exception while closing %s.\n %s",
file.getName(),
                            ioe.getMessage());
                }
            }
        }
    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1 MB chunk
     * checksums.
     */
    public static byte[] computeSHA256TreeHash(byte[][] chunkSHA256Hashes)
            throws NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        byte[][] prevLvlHashes = chunkSHA256Hashes;
        while (prevLvlHashes.length > 1) {
            int len = prevLvlHashes.length / 2;
            if (prevLvlHashes.length % 2 != 0) {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];
            int j = 0;
            for (int i = 0; i < prevLvlHashes.length; i = i + 2, j++) {

                // If there are at least two elements remaining.
                if (prevLvlHashes.length - i > 1) {

                    // Calculate a digest of the concatenated nodes.
                    md.reset();
                    md.update(prevLvlHashes[i]);
                    md.update(prevLvlHashes[i + 1]);
                    currLvlHashes[j] = md.digest();

                } else { // Take care of the remaining odd chunk
                    currLvlHashes[j] = prevLvlHashes[i];
                }
            }

            prevLvlHashes = currLvlHashes;
        }
```

```
        return prevLvlHashes[0];
    }

    /**
     * Returns the hexadecimal representation of the input byte array
     */
    public static String toHex(byte[] data) {
        StringBuilder sb = new StringBuilder(data.length * 2);
        for (byte datum : data) {
            String hex = Integer.toHexString(datum & 0xFF);

            if (hex.length() == 1) {
                // Append leading zero.
                sb.append("0");
            }
            sb.append(hex);
        }
        return sb.toString().toLowerCase();
    }
}
```

- Einzelheiten zur API finden Sie UploadArchivein der AWS SDK for Java 2.x API-Referenz.

Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch, indem Sie AWS SDK für .NET

Das folgende C#-Codebeispiel verwendet die High-Level-API von AWS SDK für .NET , um ein Beispielarchiv in den Tresor hochzuladen. Beachten Sie im Codebeispiel Folgendes:

- Das Beispiel erstellt eine Instanz der `ArchiveTransferManager` Klasse für den angegebenen Endpunkt der Amazon Glacier-Region.
- In diesem Beispiel wird die Region „USA West (Oregon)“ (`us-west-2`) verwendet.
- Im Beispiel wird der API-Vorgang `Upload` der `ArchiveTransferManager`-Klasse zum Hochladen des Archivs verwendet. Bei kleinen Archiven lädt dieser Vorgang das Archiv direkt auf Amazon Glacier hoch. Bei größeren Archiven verwendet dieser Vorgang den mehrteiligen Upload-API-Vorgang in Amazon Glacier, um den Upload zur besseren Fehlerbehebung in mehrere Teile aufzuteilen, falls beim Streamen der Daten zu Amazon Glacier Fehler auftreten.

step-by-stepAnweisungen zur Ausführung des folgenden Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit dem Namen Ihres Tresors und dem Namen des hochzuladenden Archivs aktualisieren.

> **Note**
>
> Amazon Glacier führt ein Inventar aller Archive in Ihren Tresoren. Wenn Sie das Archiv im folgenden Beispiel hochladen, wird es in der Managementkonsole erst nach der Aktualisierung des Tresorbestands im Tresor angezeigt. Diese Aktualisierung wird in der Regel einmal täglich durchgeführt.

Example— Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
    class ArchiveUploadHighLevel_GettingStarted
    {
        static string vaultName = "examplevault";
        static string archiveToUpload = "*** Provide file name (with full path) to
 upload ***";

        public static void Main(string[] args)
        {
            try
            {
                var manager = new
 ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
                // Upload an archive.
                string archiveId = manager.Upload(vaultName, "getting started archive
 test", archiveToUpload).ArchiveId;
                Console.WriteLine("Copy and save the following Archive ID for the next
 step.");
                Console.WriteLine("Archive ID: {0}", archiveId);
                Console.WriteLine("To continue, press Enter");
                Console.ReadKey();
            }
```

```
            catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
            catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
            catch (Exception e) { Console.WriteLine(e.Message); }
            Console.WriteLine("To continue, press Enter");
            Console.ReadKey();
        }
    }
}
```

Schritt 4: Laden Sie ein Archiv aus einem Tresor in Amazon Glacier herunter

In diesem Schritt laden Sie das Beispielarchiv herunter, das Sie zuvor in Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch hochgeladen haben.

> **Important**
>
> Amazon Glacier bietet eine Konsole. Für jeden Archivierungsvorgang, wie Hochladen, Herunterladen oder Löschen, müssen Sie jedoch die AWS Command Line Interface (CLI) verwenden oder Code schreiben. Es steht keine Konsolenunterstützung für Archivvorgänge zur Verfügung. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, indem Sie entweder die REST-API direkt verwenden oder indem Sie die AWS SDKs.
>
> Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie unter AWS CLI Referenz für Amazon Glacier. Beispiele für die Verwendung von AWS CLI zum Hochladen von Archiven auf Amazon Glacier finden Sie unter Verwenden von Amazon Glacier mit dem AWS Command Line Interface.

Im Allgemeinen erfolgt das Abrufen Ihrer Daten von Amazon Glacier in zwei Schritten:

1. Initiieren Sie einen Abrufauftrag.
2. Laden Sie die Datenbytes herunter, nachdem der Auftrag abgeschlossen wurde.

Um ein Archiv von Amazon Glacier abzurufen, initiieren Sie zunächst einen Job. Laden Sie die Daten herunter, nachdem der Auftrag abgeschlossen wurde. Weitere Informationen zu Archivabrufen finden Sie unter Amazon Glacier-Archive abrufen.

Die Zugriffszeit für Ihre Anforderung hängt davon ab, welche Abrufoption Sie wählen: „Expedited“, „Standard“ oder „Bulk“. Daten, die unter Verwendung von „Expedited“ abgerufen werden, stehen normalerweise innerhalb von 1 bis 5 Minuten zur Verfügung, außer wenn es sich um sehr große Archive (250 MB und mehr) handelt. Archive, die mit der Option „Standard“ abgerufen werden, sind in der Regel innerhalb von 3 bis 5 Stunden verfügbar. Bulk-Abrufe werden in der Regel innerhalb von 5 bis 12 Stunden ausgeführt. Weitere Informationen zu den verschiedenen Abruf-Optionen finden Sie in den häufig gestellten Fragen zu Amazon Glacier. Informationen zu den Gebühren für den Datenabruf finden Sie auf der Preisseite von Amazon Glacier.

Mit dem Beispielcode in den folgenden Abschnitten wird der Auftrag initiiert, danach wird gewartet, bis der Auftrag abgeschlossen ist, und abschließend werden die Archivdaten heruntergeladen.

Themen

- Laden Sie ein Archiv aus einem Tresor in Amazon Glacier herunter, indem Sie AWS SDK für Java
- Laden Sie ein Archiv aus einem Tresor in Amazon Glacier herunter, indem Sie AWS SDK für .NET

Laden Sie ein Archiv aus einem Tresor in Amazon Glacier herunter, indem Sie AWS SDK für Java

Das folgende Java-Codebeispiel verwendet die High-Level-API von AWS SDK für Java , um das Archiv herunterzuladen, das Sie im vorherigen Schritt hochgeladen haben. Beachten Sie im Codebeispiel Folgendes:

- Im Beispiel wird eine Instance der `AmazonGlacierClient`-Klasse erstellt.
- Im Code wird die Region „USA West (Oregon)“ (`us-west-2`) verwendet, damit die Region dem Standort entspricht, an dem der Tresor in Schritt 2: Einen Tresor in Amazon Glacier erstellen erstellt wurde.
- Das Beispiel verwendet den API-Vorgang `download` der `ArchiveTransferManager`-Klasse aus der High-Level-API des AWS SDK für Java. Im Beispiel werden ein Amazon Simple Notification Service (Amazon SNS)-Thema und eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange erstellt, die dieses Thema abonniert hat. Wenn Sie wie unter beschrieben einen AWS Identity and Access Management (IAM-) Administratorbenutzer erstellt habenSchritt 1: Bevor

Sie mit Amazon Glacier beginnen, verfügt Ihr Benutzer über die erforderlichen IAM-Berechtigungen für die Erstellung und Verwendung des Amazon SNS SNS-Themas und der Amazon SQS SQS-Warteschlange.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unter. Ausführen von Java-Beispielen für Amazon Glacier mit Eclipse Sie müssen den Code wie gezeigt mit der Archiv-ID der in Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch hochgeladenen Datei aktualisieren.

Example— Herunterladen eines Archivs mit dem AWS SDK für Java

```
import java.io.File;
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.transfer.ArchiveTransferManager;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sqs.AmazonSQSClient;

public class AmazonGlacierDownloadArchive_GettingStarted {
    public static String vaultName = "examplevault";
    public static String archiveId = "*** provide archive ID ***";
    public static String downloadFilePath  = "*** provide location to download archive
 ***";

    public static AmazonGlacierClient glacierClient;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

    public static void main(String[] args) throws IOException {


     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        glacierClient = new AmazonGlacierClient(credentials);
        sqsClient = new AmazonSQSClient(credentials);
        snsClient = new AmazonSNSClient(credentials);

        glacierClient.setEndpoint("glacier.us-west-2.amazonaws.com");
        sqsClient.setEndpoint("sqs.us-west-2.amazonaws.com");
        snsClient.setEndpoint("sns.us-west-2.amazonaws.com");
```

```
        try {
            ArchiveTransferManager atm = new ArchiveTransferManager(glacierClient,
 sqsClient, snsClient);

            atm.download(vaultName, archiveId, new File(downloadFilePath));

        } catch (Exception e)
        {
            System.err.println(e);
        }
    }
}
```

Laden Sie ein Archiv aus einem Tresor in Amazon Glacier herunter, indem Sie AWS SDK für .NET

Das folgende C#-Codebeispiel verwendet die High-Level-API von AWS SDK für .NET , um das Archiv herunterzuladen, in das Sie zuvor hochgeladen haben. Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch, indem Sie AWS SDK für .NET Beachten Sie im Codebeispiel Folgendes:

- Das Beispiel erstellt eine Instanz der `ArchiveTransferManager` Klasse für den angegebenen Endpunkt der Amazon Glacier-Region.
- Im Codebeispiel wird die Region „USA West (Oregon)“ (`us-west-2`) verwendet, damit die Region dem Standort entspricht, an dem der Tresor zuvor in Schritt 2: Einen Tresor in Amazon Glacier erstellen erstellt wurde.
- Im Beispiel wird der API-Vorgang `Download` der `ArchiveTransferManager`-Klasse zum Herunterladen Ihres Archivs verwendet. Im Beispiel werden ein Amazon Simple Notification Service (Amazon SNS)-Thema und eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange erstellt, die dieses Thema abonniert hat. Wenn Sie wie unter beschrieben einen AWS Identity and Access Management (IAM-) Administratorbenutzer erstellt habenSchritt 1: Bevor Sie mit Amazon Glacier beginnen, verfügt Ihr Benutzer über die erforderlichen IAM-Berechtigungen für die Erstellung und Verwendung des Amazon SNS SNS-Themas und der Amazon SQS SQS-Warteschlange.
- Das Beispiel initiiert daraufhin den Abrufauftrag für das Archiv und fragt die Warteschlange für das Archiv nach der Verfügbarkeit ab. Wenn das Archiv verfügbar ist, beginnt der Download. Weitere Informationen zu Abrufzeiten finden Sie unter Archiv-Abrufoptionen.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unter. Ausführen von Codebeispielen Sie müssen den Code wie gezeigt mit der Archiv-ID der in Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch hochgeladenen Datei aktualisieren.

Example— Laden Sie ein Archiv herunter, indem Sie die High-Level-API des AWS SDK für .NET

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
    class ArchiveDownloadHighLevel_GettingStarted
    {
        static string vaultName = "examplevault";
        static string archiveId = "*** Provide archive ID ***";
        static string downloadFilePath = "*** Provide the file name and path to where
 to store the download ***";

        public static void Main(string[] args)
        {
            try
            {
                var manager = new
 ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);

                var options = new DownloadOptions();
                options.StreamTransferProgress +=
 ArchiveDownloadHighLevel_GettingStarted.progress;
                // Download an archive.
                Console.WriteLine("Intiating the archive retrieval job and then polling
 SQS queue for the archive to be available.");
                Console.WriteLine("Once the archive is available, downloading will
 begin.");
                manager.Download(vaultName, archiveId, downloadFilePath, options);
                Console.WriteLine("To continue, press Enter");
                Console.ReadKey();
            }
            catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
            catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
            catch (Exception e) { Console.WriteLine(e.Message); }
            Console.WriteLine("To continue, press Enter");
```

```
            Console.ReadKey();
        }

        static int currentPercentage = -1;
        static void progress(object sender, StreamTransferProgressArgs args)
        {
            if (args.PercentDone != currentPercentage)
            {
                currentPercentage = args.PercentDone;
                Console.WriteLine("Downloaded {0}%", args.PercentDone);
            }
        }
    }
}
```

Schritt 5: Löschen eines Archivs aus einem Tresor in Amazon Glacier

In diesem Schritt löschen Sie das Beispielarchiv, das Sie zuvor entsprechend Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch hochgeladen haben.

> **Important**
>
> Sie können ein Archiv nicht mithilfe der Amazon Glacier-Konsole löschen. Für jeden Archivierungsvorgang, wie Hochladen, Herunterladen oder Löschen, müssen Sie die AWS Command Line Interface (CLI) verwenden oder Code schreiben. Um Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über die AWS SDKs.
>
> Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie unter AWS CLI Referenz für Amazon Glacier. Beispiele für die Verwendung von AWS CLI zum Hochladen von Archiven auf Amazon Glacier finden Sie unter Verwenden von Amazon Glacier mit dem AWS Command Line Interface.

Löschen Sie das Beispielarchiv, indem Sie einer der folgenden SDKs Methoden folgen oder AWS CLI:

- Löschen Sie ein Archiv aus einem Tresor in Amazon Glacier mithilfe der AWS SDK für Java
- Löschen Sie ein Archiv aus einem Tresor in Amazon Glacier mit dem AWS SDK für .NET
- Löschen Sie ein Archiv in Amazon Glacier mit dem AWS CLI

Zugehörige Abschnitte

- Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch
- Löschen eines Archivs in Amazon Glacier

Löschen Sie ein Archiv aus einem Tresor in Amazon Glacier mithilfe der AWS SDK für Java

Das folgende Codebeispiel verwendet den AWS SDK für Java , um das Archiv zu löschen. Beachten Sie im Code Folgendes:

- Das `DeleteArchiveRequest`-Objekt beschreibt die Löschanforderung, einschließlich des Namens von dem Tresor, in dem sich das Archiv befindet, und der Archiv-ID.
- Der `deleteArchive` API-Vorgang sendet die Anfrage an Amazon Glacier, um das Archiv zu löschen.
- In diesem Beispiel wird die Region „USA West (Oregon)“ (`us-west-2`) verwendet.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt mit der Archiv-ID der in Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch hochgeladenen Datei aktualisieren.

Example— Löschen eines Archivs mit dem AWS SDK für Java

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.DeleteArchiveRequest;

public class AmazonGlacierDeleteArchive_GettingStarted {
```

```
    public static String vaultName = "examplevault";
    public static String archiveId = "*** provide archive ID***";
    public static AmazonGlacierClient client;

    public static void main(String[] args) throws IOException {


     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

        try {

            // Delete the archive.
            client.deleteArchive(new DeleteArchiveRequest()
                .withVaultName(vaultName)
                .withArchiveId(archiveId));

            System.out.println("Deleted archive successfully.");

        } catch (Exception e) {
            System.err.println("Archive not deleted.");
            System.err.println(e);
        }
    }
}
```

Löschen Sie ein Archiv aus einem Tresor in Amazon Glacier mit dem AWS SDK für .NET

Das folgende C#-Codebeispiel verwendet die High-Level-API von AWS SDK für .NET , um das Archiv zu löschen, das Sie im vorherigen Schritt hochgeladen haben. Beachten Sie im Codebeispiel Folgendes:

- Das Beispiel erstellt eine Instanz der `ArchiveTransferManager` Klasse für den angegebenen Endpunkt der Amazon Glacier-Region.
- In diesem Beispiel wird die Region „USA West (Oregon)“ (`us-west-2`) verwendet.
- Das Beispiel verwendet den API-Vorgang `Delete` der `ArchiveTransferManager`-Klasse, die als Teil der High-Level-API des AWS SDK für .NET bereitgestellt wird.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit der Archiv-ID der in Schritt 3: Laden Sie ein Archiv in einen Tresor in Amazon Glacier hoch hochgeladenen Datei aktualisieren.

Example— Löschen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDeleteHighLevel_GettingStarted
  {
    static string vaultName = "examplevault";
    static string archiveId = "*** Provide archive ID ***";

    public static void Main(string[] args)
    {
      try
      {
        var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
        manager.DeleteArchive(vaultName, archiveId);
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

Löschen Sie ein Archiv in Amazon Glacier mit dem AWS CLI

Sie können Archive in Amazon Glacier mit AWS Command Line Interface (AWS CLI) löschen.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Löschen eines Archivs mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface
2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

     ```
     aws configure list
     ```

Beispiel: Löschen eines Archivs mit dem AWS CLI

1. Verwenden Sie den `initiate-job`-Befehl, um eine Lagerabrufaufgabe zu starten. Weitere Informationen zum `initiate-job`-Befehl finden Sie unter Initiate Job.

   ```
   aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
   job-parameters "{\"Type\": \"inventory-retrieval\"}"
   ```

   Erwartete Ausgabe:

   ```
   {
       "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
       "jobId": "*** jobid ***"
   ```

```
}
```

2. Verwenden Sie den `describe-job`-Befehl, um den Status des vorherigen Abrufauftrags zu überprüfen. Weitere Informationen zum `describe-job`-Befehl finden Sie unter Describe Job.

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

Erwartete Ausgabe:

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. Warten Sie, bis der Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie beim Initiieren des Jobs eine Benachrichtigungskonfiguration für den Tresor festgelegt oder ein Amazon Simple Notification Service (Amazon SNS) -Thema angegeben haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an das Thema.

   Sie können Benachrichtigungskonfiguration für bestimmte Ereignisse auf dem Tresor einrichten. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier. Amazon Glacier sendet bei jedem Eintreten eines bestimmten Ereignisses eine Nachricht an das angegebene Amazon SNS SNS-Thema.

4. Nach Abschluss des Auftrags laden Sie den Abrufauftrag mit dem `get-job-output`-Befehl in die Datei „`output.json`“ herunter. Weitere Informationen zum `get-job-output`-Befehl finden Sie unter Get Job Output.

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output.json
```

Dieser Befehl erzeugt eine Datei mit den folgenden Feldern.

```
{
"VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
"InventoryDate":"*** job completion date ***",
"ArchiveList":[{
{"ArchiveId":"*** archiveid ***",
"ArchiveDescription":"*** archive description (if set) ***",
"CreationDate":"*** archive creation date ***",
"Size":"*** archive size (in bytes) ***",
"SHA256TreeHash":"*** archive hash ***"
}],
"ArchiveId": 123456789

}
```

5. Verwenden Sie den `delete-archive`-Befehl, um jedes Archiv aus einem Tresor zu löschen, bis kein Archiv übrig bleibt.

```
aws glacier delete-archive --vault-name awsexamplevault --account-id 111122223333
 --archive-id="*** archiveid ***"
```

Weitere Informationen zum `delete-archive`-Befehl finden Sie unter Delete Archive.

Schritt 6: Löschen eines Tresors in Amazon Glacier

Ein Tresor ist ein Container für das Speichern von Archiven. Um einen Amazon Glacier-Tresor zu löschen, müssen Sie zunächst alle vorhandenen Archive im Tresor löschen, die seit dem letzten Inventar, das Amazon Glacier berechnet hat, vorhanden sind.

Sie können einen Tresor programmgesteuert oder mithilfe der Amazon Glacier-Konsole löschen. Informationen zum programmatischen Löschen eines Tresors finden Sie unter Löschen eines Tresors in Amazon Glacier.

> **Important**
>
> Wenn Sie innerhalb der letzten 24 Stunden ein Archiv in einen Tresor hochgeladen oder ein Archiv aus einem Tresor gelöscht haben, müssen Sie warten, bis der letzte Tresorbestand

aktualisiert wurde, um die neuesten Informationen wiederzugeben. Amazon Glacier bereitet regelmäßig eine Inventur für jeden Tresor vor, alle 24 Stunden.

So löschen Sie einen leeren Tresor

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie im Menü „Region auswählen“ den AWS-Region für den Tresor aus, den Sie löschen möchten.

   In dieser Übung für die ersten Schritte befindet sich Ihr Beispieltresor in der Region „USA West (Oregon)“.
3. Aktivieren Sie die Optionsschaltfläche neben dem leeren Tresor, den Sie löschen möchten. Wenn der Tresor nicht leer ist, müssen Sie alle Archive löschen, bevor Sie den Tresor löschen. Weitere Informationen finden Sie unter Löschen eines Archivs in Amazon Glacier.

   > **Important**
   >
   > Das Löschen eines Tresors kann nicht rückgängig gemacht werden.
4. Wählen Sie Löschen aus.
5. Das Dialogfeld Tresor löschen wird angezeigt. Wählen Sie Löschen aus.

So löschen Sie einen Tresor, der nicht leer ist

1. Wenn Sie einen Tresor löschen möchten, der nicht leer ist, müssen Sie zunächst alle vorhandenen Archive löschen, bevor Sie den Tresor löschen. Sie können dies tun, indem Sie Code schreiben, um eine Anfrage zum Löschen eines Archivs zu stellen, indem Sie entweder die REST-API, die AWS SDK für Java, die AWS SDK für .NET oder die verwenden AWS CLI. Informationen zum Löschen von Archiven finden Sie unter Schritt 5: Löschen eines Archivs aus einem Tresor in Amazon Glacier.
2. Wenn der Tresor leer ist, gehen Sie wie im vorherigen Verfahren beschrieben vor, um einen leeren Tresor zu löschen.

Wie geht es weiter?

Nachdem Sie die Übung „Erste Schritte“ ausprobiert haben, können Sie in den folgenden Abschnitten mehr über Amazon Glacier erfahren.

- Arbeiten mit Tresoren in Amazon Glacier
- Arbeiten mit Archiven in Amazon Glacier

Arbeiten mit Tresoren in Amazon Glacier

Ein Tresor ist ein Container für das Speichern von Archiven. Wenn Sie einen Tresor erstellen, geben Sie einen Tresornamen und den Namen an, AWS-Region in dem Sie den Tresor erstellen möchten. Eine Liste der von Amazon Glacier AWS-Regionen unterstützten Endpunkte und Kontingente finden Sie unter Amazon Glacier-Endpunkte und Kontingente in der AWS Allgemeinen Referenz.

Sie können eine unbegrenzte Anzahl von Archiven in einem Tresor speichern.

> **Important**
>
> Amazon Glacier bietet eine Konsole. Für jeden Archivierungsvorgang, wie Hochladen, Herunterladen oder Löschen, müssen Sie jedoch AWS Command Line Interface (AWS CLI) verwenden oder Code schreiben. Es steht keine Konsolenunterstützung für Archivvorgänge zur Verfügung. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über AWS SDKs. Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie in der AWS CLI Referenz für Amazon Glacier. Beispiele für die Verwendung von AWS CLI zum Hochladen von Archiven auf Amazon Glacier finden Sie unter Verwenden von Amazon Glacier mit dem AWS Command Line Interface.

Themen

- Tresorvorgänge in Amazon Glacier
- Einen Tresor in Amazon Glacier erstellen
- Abrufen von Tresor-Metadaten in Amazon Glacier
- Ein Tresor-Inventar in Amazon Glacier herunterladen
- Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier
- Löschen eines Tresors in Amazon Glacier
- Kennzeichnen Ihrer Amazon Glacier-Tresore
- Amazon Glacier Tresorschloss

Tresorvorgänge in Amazon Glacier

Amazon Glacier unterstützt verschiedene Tresoroperationen. Tresorvorgänge sind spezifisch für eine bestimmte AWS-Regionen. Mit anderen Worten, wenn Sie einen Tresor erstellen, dann erstellen Sie ihn in einer bestimmten AWS-Region. Wenn Sie Tresore auflisten, gibt Amazon Glacier die Tresorliste aus der Liste zurück AWS-Region , die Sie in der Anfrage angegeben haben.

Erstellen und Löschen von Tresoren

An AWS-Konto kann bis zu 1.000 Tresore pro Person erstellen. AWS-Region Eine Liste der von Amazon Glacier AWS-Regionen unterstützten Endpunkte und Kontingente finden Sie unter Amazon Glacier-Endpunkte und Kontingente in der AWS Allgemeinen Referenz.

Sie können einen Tresor nur löschen, wenn sich zum Zeitpunkt der letzten Inventarisierung, die Amazon Glacier berechnet hat, keine Archive im Tresor befinden und wenn seit der letzten Inventur keine Schreibvorgänge in den Tresor stattgefunden haben.

> **Note**
>
> Amazon Glacier bereitet regelmäßig eine Inventur für jeden Tresor vor, alle 24 Stunden. Da das Inventar möglicherweise nicht die neuesten Informationen enthält, stellt Amazon Glacier sicher, dass der Tresor tatsächlich leer ist, indem überprüft wird, ob seit der letzten Tresorinventur Schreibvorgänge stattgefunden haben.

Weitere Informationen erhalten Sie unter Einen Tresor in Amazon Glacier erstellen und Löschen eines Tresors in Amazon Glacier.

Abrufen von Tresormetadaten

Sie können Informationen über einen Tresor abrufen, u. a. das Erstellungsdatum, die Anzahl der Archive und die Gesamtgröße aller Archive im Tresor. Amazon Glacier stellt API-Aufrufe bereit, mit denen Sie diese Informationen für einen bestimmten Tresor oder alle Tresore in einem bestimmten Tresor AWS-Region in Ihrem Konto abrufen können. Weitere Informationen finden Sie unter Abrufen von Tresor-Metadaten in Amazon Glacier.

Herunterladen eines Tresorbestands

Ein Tresorbestand bezeichnet die Liste der Archive in einem Tresor. Für jedes Archiv in der Liste enthält das Inventar Archivinformationen wie die Archiv-ID, das Erstellungsdatum und die Größe. Amazon Glacier aktualisiert das Tresorinventar einmal täglich, beginnend mit dem Tag, an dem das erste Archiv in den Tresor hochgeladen wird. Ein Tresorbestand muss vorhanden sein, damit Sie ihn herunterladen können.

Das Herunterladen eines Tresorbestands ist eine asynchrone Operation. Sie müssen zuerst einen Auftrag initiieren, um das Inventar herunterzuladen. Nach Erhalt der Jobanfrage bereitet Amazon Glacier Ihr Inventar für den Download vor. Sie können die Inventardaten herunterladen, nachdem der Auftrag abgeschlossen wurde.

Aufgrund der asynchronen Natur des Auftrags können Sie Amazon Simple Notification Service (Amazon SNS)-Benachrichtigungen verwenden, damit Sie informiert werden, wenn der Auftrag abgeschlossen wurde. Sie können ein Amazon-SNS-Thema für jede einzelne Auftragsanforderung angeben oder Ihren Tresor so konfigurieren, dass eine Benachrichtigung gesendet wird, wenn bestimmte Tresorereignisse eintreten.

Amazon Glacier bereitet regelmäßig eine Inventur für jeden Tresor vor, alle 24 Stunden. Wenn dem Tresor seit der letzten Inventur keine Archive hinzugefügt oder aus ihm gelöscht wurden, wird das Inventurdatum nicht aktualisiert.

Wenn Sie einen Job für ein Tresorinventar starten, gibt Amazon Glacier das zuletzt generierte Inventar zurück, bei dem es sich um einen point-in-time Snapshot und nicht um Echtzeitdaten handelt. Möglicherweise finden Sie es nicht nützlich, einen Tresorbestand für jeden Archiv-Upload abzurufen. Nehmen wir jedoch an, dass Sie auf der Clientseite eine Datenbank verwalten, die Metadaten enthält, die mit den Archiven verknüpft sind, die Sie auf Amazon Glacier hochladen. Dann finden Sie es möglicherweise nützlich, wenn die Tresorinventur die Informationen in Ihrer Datenbank mit dem aktuellen Tresorbestand abgleicht.

Weitere Informationen zum Abrufen eines Tresorbestands finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

Konfigurieren von Tresorbenachrichtigungen

Das Abrufen von Daten aus Amazon Glacier, z. B. eines Archivs aus einem Tresor oder einem Tresorbestand, erfolgt in zwei Schritten. Zunächst initiieren Sie einen Auftrag. Sie können die

Ausgabe herunterladen, nachdem der Auftrag abgeschlossen wurde. Um zu erfahren, wann Ihr Job abgeschlossen ist, können Sie Amazon Glacier-Benachrichtigungen verwenden. Amazon Glacier sendet Benachrichtigungen an ein von Ihnen bereitgestelltes Amazon Simple Notification Service (Amazon SNS) -Thema.

Sie können Benachrichtigungen für einen Tresor konfigurieren und Tresorereignisse sowie das Amazon-SNS-Thema angeben, an das bei Eintritt des Ereignisses eine Benachrichtigung gesendet wird. Jedes Mal, wenn das Tresorereignis eintritt, sendet Amazon Glacier eine Benachrichtigung an das angegebene Amazon SNS SNS-Thema. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier.

Einen Tresor in Amazon Glacier erstellen

Das Erstellen eines Tresors fügt der Reihe von Tresoren in Ihrem Konto einen Tresor hinzu. An AWS-Konto kann bis zu 1.000 Tresore pro AWS Region erstellen. Eine Liste der von Amazon Glacier (Amazon Glacier) unterstützten AWS Regionen finden Sie in der AWS allgemeinen Referenz unter Regionen und Endpunkte.

Wenn Sie einen Tresor erstellen, müssen Sie einen Tresornamen angeben. Folgende Bedingungen müssen Sie beim Benennen von Tresoren beachten:

- Namen können zwischen 1 und 255 Zeichen lang sein.
- Gültige Zeichen sind a-z, A-Z, 0–9, „_“ (Unterstrich), „-“ (Bindestrich) und „.“ (Punkt).

Tresornamen müssen innerhalb eines Kontos und der AWS Region, in der der Tresor erstellt wird, eindeutig sein. Das heißt, ein Konto kann Tresore mit demselben Namen in verschiedenen AWS Regionen, aber nicht in derselben AWS Region erstellen.

Themen

- Erstellen eines Tresors in Amazon Glacier mit dem AWS SDK für Java
- Erstellen eines Tresors in Amazon Glacier mit dem AWS SDK für .NET
- Einen Tresor in Amazon Glacier mithilfe der REST-API erstellen
- Einen Tresor mit der Amazon Glacier Console erstellen
- Erstellen eines Tresors in Amazon Glacier mit dem AWS Command Line Interface

Erstellen eines Tresors in Amazon Glacier mit dem AWS SDK für Java

Die Low-Level-API bietet Methoden für alle Tresorvorgänge, einschließlich des Erstellens und Löschens von Tresoren, des Abrufs einer Tresorbeschreibung und des Abrufs einer Liste von Tresoren, die in einem bestimmten Bereich erstellt wurden. AWS-Region Im Folgenden werden die Schritte zum Erstellen eines Tresors mit AWS SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen einen Tresor angeben, AWS-Region in dem Sie einen Tresor erstellen möchten. Alle Operationen, die Sie mit diesem Client durchführen, gelten für diese AWS-Region.

2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `CreateVaultRequest`-Klasse.

   Amazon Glacier (Amazon Glacier) verlangt von Ihnen die Angabe eines Tresornamens und Ihrer Konto-ID. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anfrage verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier.

3. Führen Sie die `createVault`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Die Antwort, die Amazon Glacier zurückgibt, ist im `CreateVaultResult` Objekt verfügbar.

Der folgende Java-Codeausschnitt veranschaulicht die vorherigen Schritte. Der Ausschnitt erstellt einen Tresor in der Region `us-west-2`. `Location`Es druckt die relative URI des Tresors, die Ihre Konto-ID AWS-Region, den und den Tresornamen enthält.

```
AmazonGlacierClient client = new AmazonGlacierClient(credentials);
client.setEndpoint("https://glacier.us-west-2.amazonaws.com");

CreateVaultRequest request = new CreateVaultRequest()
    .withVaultName("*** provide vault name ***");
CreateVaultResult result = client.createVault(request);

System.out.println("Created vault successfully: " + result.getLocation());
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Create Vault (PUT vault).

Beispiel: Erstellen eines Tresors mit dem AWS SDK für Java

Das folgende Java-Codebeispiel erstellt einen Tresor in der `us-west-2` Region (weitere Informationen dazu finden Sie unterZugriff auf Amazon Glacier). AWS-Regionen Darüber hinaus ruft das Codebeispiel die Tresorinformationen ab, listet alle Tresore im selben AWS-Region Verzeichnis auf und löscht dann den erstellten Tresor.

step-by-stepAnweisungen zur Ausführung des folgenden Beispiels finden Sie unter. Ausführen von Java-Beispielen für Amazon Glacier mit Eclipse

Example

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

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.CreateVaultRequest;
import com.amazonaws.services.glacier.model.CreateVaultResult;
import com.amazonaws.services.glacier.model.DeleteVaultRequest;
import com.amazonaws.services.glacier.model.DescribeVaultOutput;
import com.amazonaws.services.glacier.model.DescribeVaultRequest;
import com.amazonaws.services.glacier.model.DescribeVaultResult;
import com.amazonaws.services.glacier.model.ListVaultsRequest;
import com.amazonaws.services.glacier.model.ListVaultsResult;


public class AmazonGlacierVaultOperations {

    public static AmazonGlacierClient client;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
```

```
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        String vaultName = "examplevaultfordelete";

        try {
            createVault(client, vaultName);
            describeVault(client, vaultName);
            listVaults(client);
            deleteVault(client, vaultName);

        } catch (Exception e) {
            System.err.println("Vault operation failed." + e.getMessage());
        }
    }

    private static void createVault(AmazonGlacierClient client, String vaultName) {
        CreateVaultRequest createVaultRequest = new CreateVaultRequest()
            .withVaultName(vaultName);
        CreateVaultResult createVaultResult = client.createVault(createVaultRequest);

        System.out.println("Created vault successfully: " +
createVaultResult.getLocation());
    }

    private static void describeVault(AmazonGlacierClient client, String vaultName) {
        DescribeVaultRequest describeVaultRequest = new DescribeVaultRequest()
            .withVaultName(vaultName);
        DescribeVaultResult describeVaultResult  =
client.describeVault(describeVaultRequest);

        System.out.println("Describing the vault: " + vaultName);
        System.out.print(
                "CreationDate: " + describeVaultResult.getCreationDate() +
                "\nLastInventoryDate: " + describeVaultResult.getLastInventoryDate() +
                "\nNumberOfArchives: " + describeVaultResult.getNumberOfArchives() +
                "\nSizeInBytes: " + describeVaultResult.getSizeInBytes() +
                "\nVaultARN: " + describeVaultResult.getVaultARN() +
                "\nVaultName: " + describeVaultResult.getVaultName());
    }

    private static void listVaults(AmazonGlacierClient client) {
        ListVaultsRequest listVaultsRequest = new ListVaultsRequest();
        ListVaultsResult listVaultsResult = client.listVaults(listVaultsRequest);
```

```
        List<DescribeVaultOutput> vaultList = listVaultsResult.getVaultList();
        System.out.println("\nDescribing all vaults (vault list):");
        for (DescribeVaultOutput vault : vaultList) {
            System.out.println(
                    "\nCreationDate: " + vault.getCreationDate() +
                    "\nLastInventoryDate: " + vault.getLastInventoryDate() +
                    "\nNumberOfArchives: " + vault.getNumberOfArchives() +
                    "\nSizeInBytes: " + vault.getSizeInBytes() +
                    "\nVaultARN: " + vault.getVaultARN() +
                    "\nVaultName: " + vault.getVaultName());
        }
    }

    private static void deleteVault(AmazonGlacierClient client, String vaultName) {
        DeleteVaultRequest request = new DeleteVaultRequest()
            .withVaultName(vaultName);
        client.deleteVault(request);
        System.out.println("Deleted vault: " + vaultName);
    }

}
```

Erstellen eines Tresors in Amazon Glacier mit dem AWS SDK für .NET

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for .NET APIs bereitgestellt werden, bieten eine Methode zum Erstellen eines Tresors.

Themen

- Erstellen eines Tresors mithilfe der High-Level-API von AWS SDK für .NET
- Erstellen eines Tresors mithilfe der Low-Level-API von AWS SDK für .NET

Erstellen eines Tresors mithilfe der High-Level-API von AWS SDK für .NET

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `CreateVault`-Methode, mit der Sie einen Tresor in einer AWS -Region erstellen können.

Beispiel: Tresoroperationen unter Verwendung der High-Level-API von AWS SDK für .NET

Mit dem folgenden C#-Beispielcode wird ein Tresor in der Region „USA West (Oregon)“ erstellt und gelöscht. Eine Liste, AWS-Regionen in denen Sie Tresore erstellen können, finden Sie unterZugriff auf Amazon Glacier.

step-by-stepAnweisungen zur Ausführung des folgenden Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit einem Tresornamen aktualisieren.

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class VaultCreateDescribeListVaultsDeleteHighLevel
  {
    static string vaultName = "*** Provide vault name ***";

    public static void Main(string[] args)
    {
      try
      {
          var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
          manager.CreateVault(vaultName);
          Console.WriteLine("Vault created. To delete the vault, press Enter");
          Console.ReadKey();
          manager.DeleteVault(vaultName);
          Console.WriteLine("\nVault deleted. To continue, press Enter");
          Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

Erstellen eines Tresors mithilfe der Low-Level-API von AWS SDK für .NET

Die Low-Level-API bietet Methoden für alle Tresorvorgänge, darunter das Erstellen und Löschen von Tresoren, das Abrufen einer Tresorbeschreibung und das Abrufen einer Liste von Tresoren, die in einem bestimmten Bereich erstellt wurden. AWS-Region Im Folgenden werden die Schritte zum Erstellen eines Tresors mit AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen einen Tresor angeben, AWS-Region in dem Sie einen Tresor erstellen möchten. Alle Operationen, die Sie mit diesem Client durchführen, gelten für diese AWS-Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `CreateVaultRequest`-Klasse.

   Amazon Glacier (Amazon Glacier) verlangt von Ihnen die Angabe eines Tresornamens und Ihrer Konto-ID. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für .NET mit Amazon Glacier.
3. Führen Sie die `CreateVault`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Die Antwort, die Amazon Glacier zurückgibt, ist im `CreateVaultResponse` Objekt verfügbar.

Beispiel: Vault-Operationen unter Verwendung der Low-Level-API des AWS SDK für .NET

Im folgenden C#-Codebeispiel werden die vorstehenden Schritte veranschaulicht. Das Beispiel erstellt einen Tresor in der Region „USA West (Oregon)“. Darüber hinaus ruft das Codebeispiel die Tresorinformationen ab, listet alle Tresore im selben AWS-Region Verzeichnis auf und löscht dann den erstellten Tresor. `Location`Gedruckt wird der relative URI des Tresors, der Ihre Konto-ID AWS-Region, den und den Tresornamen enthält.

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Create Vault (PUT vault).

step-by-stepAnweisungen zur Ausführung des folgenden Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit einem Tresornamen aktualisieren.

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Model;
```

```
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class VaultCreateDescribeListVaultsDelete
  {
    static string vaultName = "*** Provide vault name ***";
    static AmazonGlacierClient client;

    public static void Main(string[] args)
    {
       try
      {
         using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Creating a vault.");
          CreateAVault();
          DescribeVault();
          GetVaultsList();
          Console.WriteLine("\nVault created. Now press Enter to delete the vault...");
          Console.ReadKey();
          DeleteVault();
        }
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static void CreateAVault()
    {
      CreateVaultRequest request = new CreateVaultRequest()
      {
        VaultName = vaultName
      };
      CreateVaultResponse response = client.CreateVault(request);
      Console.WriteLine("Vault created: {0}\n", response.Location);
    }

    static void DescribeVault()
    {
      DescribeVaultRequest describeVaultRequest = new DescribeVaultRequest()
```

```
      {
        VaultName = vaultName
      };

      DescribeVaultResponse describeVaultResponse =
client.DescribeVault(describeVaultRequest);
      Console.WriteLine("\nVault description...");
      Console.WriteLine(
        "\nVaultName: " + describeVaultResponse.VaultName +
        "\nVaultARN: " + describeVaultResponse.VaultARN +
        "\nVaultCreationDate: " + describeVaultResponse.CreationDate +
        "\nNumberOfArchives: " + describeVaultResponse.NumberOfArchives +
        "\nSizeInBytes: " + describeVaultResponse.SizeInBytes +
        "\nLastInventoryDate: " + describeVaultResponse.LastInventoryDate
        );
    }

    static void GetVaultsList()
    {
      string lastMarker = null;
      Console.WriteLine("\n List of vaults in your account in the specific
region ...");
      do
      {
        ListVaultsRequest request = new ListVaultsRequest()
        {
          Marker = lastMarker
        };
        ListVaultsResponse response = client.ListVaults(request);

        foreach (DescribeVaultOutput output in response.VaultList)
        {
          Console.WriteLine("Vault Name: {0} \tCreation Date: {1} \t #of archives:
{2}",
                            output.VaultName, output.CreationDate,
output.NumberOfArchives);
        }
        lastMarker = response.Marker;
      } while (lastMarker != null);
    }

    static void DeleteVault()
    {
      DeleteVaultRequest request = new DeleteVaultRequest()
```

```
      {
        VaultName = vaultName
      };
      DeleteVaultResponse response = client.DeleteVault(request);
    }
  }
}
```

Einen Tresor in Amazon Glacier mithilfe der REST-API erstellen

Zum Erstellen eines Tresors mit der REST-API siehe Create Vault (PUT vault).

Einen Tresor mit der Amazon Glacier Console erstellen

Informationen zum Erstellen eines Tresors mit der Amazon Glacier (Amazon Glacier) -Konsole finden Sie Schritt 2: Einen Tresor in Amazon Glacier erstellen im Tutorial Erste Schritte.

Erstellen eines Tresors in Amazon Glacier mit dem AWS Command Line Interface

Gehen Sie wie folgt vor, um mit dem () einen Tresor in Amazon Glacier (Amazon Glacier AWS CLI) zu erstellen. AWS Command Line Interface

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Erstellen eines Tresors mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface
2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

- Versuchen Sie, den Befehl help zu verwenden.

```
aws help
```

- Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

```
aws glacier list-vaults --account-id 123456789012
```

- Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

```
aws configure list
```

Beispiel: Erstellen eines Tresors mit dem AWS CLI

1. Verwenden Sie den `create-vault` Befehl, um einen Tresor mit dem Namen *awsexamplevault* Account zu erstellen*111122223333*.

```
aws glacier create-vault --vault-name awsexamplevault --account-id 111122223333
```

   Erwartete Ausgabe:

```
{
    "location": "/111122223333/vaults/awsexamplevault"
}
```

2. Überprüfen Sie die Erstellung mithilfe des `describe-vault`-Befehls.

```
aws glacier describe-vault --vault-name awsexamplevault --account-id 111122223333
```

Abrufen von Tresor-Metadaten in Amazon Glacier

Sie können Informationen über einen Tresor abrufen, u. a. das Erstellungsdatum, die Anzahl der Archive und die Gesamtgröße aller Archive im Tresor. Amazon Glacier (Amazon Glacier) stellt API-

Aufrufe bereit, mit denen Sie diese Informationen für einen bestimmten Tresor oder alle Tresore in einer bestimmten AWS Region in Ihrem Konto abrufen können.

Wenn Sie eine Tresorliste abrufen, gibt Amazon Glacier die Liste sortiert nach den ASCII-Werten der Tresornamen zurück. Die Liste enthält bis zu 1 000 Tresore. Sie sollten immer das die Antwort auf eine Markierung überprüfen, bei der die Liste fortgesetzt werden soll. Wenn keine Elemente mehr vorhanden sind, ist der Wert des Markierungsfelds gleich `null`. Sie können die Anzahl der in der Antwort zurückgegebenen Tresore begrenzen. Wenn mehr Tresore vorhanden sind, als in der Antwort zurückgegeben werden, wird das Ergebnis paginiert. Sie müssen zusätzliche Anfragen senden, um den nächsten Satz Tresore abzurufen.

Themen

- Abrufen von Tresor-Metadaten in Amazon Glacier mithilfe der AWS SDK für Java
- Abrufen von Tresor-Metadaten in Amazon Glacier mithilfe der AWS SDK für .NET
- Abrufen von Tresormetadaten mithilfe der REST-API
- Abrufen von Tresor-Metadaten in Amazon Glacier mithilfe der AWS Command Line Interface

Abrufen von Tresor-Metadaten in Amazon Glacier mithilfe der AWS SDK für Java

Themen

- Abrufen von Tresor-Metadaten für einen Tresor
- Abrufen von Tresor-Metadaten für alle Tresore in einer Region
- Beispiel: Abrufen von Tresormetadaten mit dem Amazon-SDK für Java

Abrufen von Tresor-Metadaten für einen Tresor

Sie können Metadaten für einen bestimmten Tresor oder alle Tresore in einer bestimmten AWS Region abrufen. Nachfolgend werden die Schritte zum Abrufen der Tresormetadaten für einen bestimmten Tresor mithilfe der Low-Level-API des Amazon-SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der sich der Tresor befindet. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.

2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `DescribeVaultRequest`-Klasse.

   Amazon Glacier (Amazon Glacier) verlangt von Ihnen die Angabe eines Tresornamens und Ihrer Konto-ID. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier.
3. Führen Sie die `describeVault`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Die Tresor-Metadateninformationen, die Amazon Glacier zurückgibt, sind im `DescribeVaultResult` Objekt verfügbar.

Der folgende Java-Codeausschnitt veranschaulicht die vorherigen Schritte.

```
DescribeVaultRequest request = new DescribeVaultRequest()
 .withVaultName("*** provide vault name***");

DescribeVaultResult result = client.describeVault(request);

System.out.print(
        "\nCreationDate: " + result.getCreationDate() +
        "\nLastInventoryDate: " + result.getLastInventoryDate() +
        "\nNumberOfArchives: " + result.getNumberOfArchives() +
        "\nSizeInBytes: " + result.getSizeInBytes() +
        "\nVaultARN: " + result.getVaultARN() +
        "\nVaultName: " + result.getVaultName());
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Describe Vault (GET vault).

Abrufen von Tresor-Metadaten für alle Tresore in einer Region

Sie können die `listVaults` Methode auch verwenden, um Metadaten für alle Tresore in einer bestimmten AWS Region abzurufen.

Mit dem folgenden Java-Code-Snippet wird eine Liste der Tresore in der Region `us-west-2` abgerufen. Die Anzahl der bei der Anforderung zurückgegebenen Tresore ist auf 5 beschränkt. Der Codeausschnitt führt dann eine Reihe von `listVaults` Aufrufen durch, um die gesamte Tresorliste aus der Region abzurufen. AWS

```
AmazonGlacierClient client;
client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

String marker = null;
do {
    ListVaultsRequest request = new ListVaultsRequest()
        .withLimit("5")
        .withMarker(marker);
    ListVaultsResult listVaultsResult = client.listVaults(request);

    List<DescribeVaultOutput> vaultList = listVaultsResult.getVaultList();
    marker = listVaultsResult.getMarker();
    for (DescribeVaultOutput vault : vaultList) {
        System.out.println(
                "\nCreationDate: " + vault.getCreationDate() +
                "\nLastInventoryDate: " + vault.getLastInventoryDate() +
                "\nNumberOfArchives: " + vault.getNumberOfArchives() +
                "\nSizeInBytes: " + vault.getSizeInBytes() +
                "\nVaultARN: " + vault.getVaultARN() +
                "\nVaultName: " + vault.getVaultName());
    }
} while (marker != null);
```

Wenn Sie im vorherigen Codesegment den `Limit` Wert in der Anfrage nicht angeben, gibt Amazon Glacier bis zu 10 Tresore zurück, wie von der Amazon Glacier-API festgelegt. Wenn mehr aufzulistende Tresore vorhanden sind, enthält das Antwortfeld `marker` den Amazon-Ressourcennamen (ARN) des Tresors, bei dem die Liste mit einer neuen Anforderung fortzusetzen ist. Anderenfalls hat das Feld `marker` den Wert null.

Beachten Sie, dass die Informationen, die in der Liste für jeden Tresor zurückgegeben werden, mit den Informationen übereinstimmen, die Sie durch Aufrufen der `describeVault`-Methode für einen bestimmten Tresor erhalten.

> **Note**
>
> Durch die `listVaults`-Methode wird die zugrunde liegende REST-API aufgerufen (siehe List Vaults (GET vaults)).

Beispiel: Abrufen von Tresormetadaten mit dem Amazon-SDK für Java

Ein funktionierendes Codebeispiel finden Sie unter Beispiel: Erstellen eines Tresors mit dem AWS SDK für Java. Der folgende Java-Beispielcode erstellt einen Tresor und ruft die Tresor-Metadaten ab.

Abrufen von Tresor-Metadaten in Amazon Glacier mithilfe der AWS SDK für .NET

Themen

- Abrufen von Tresor-Metadaten für einen Tresor
- Abrufen von Tresor-Metadaten für alle Tresore in einer Region
- Beispiel: Abrufen von Tresor-Metadaten mithilfe der Low-Level-API von AWS SDK für .NET

Abrufen von Tresor-Metadaten für einen Tresor

Sie können Metadaten für einen bestimmten Tresor oder alle Tresore in einer bestimmten Region abrufen. AWS Nachfolgend werden die Schritte zum Abrufen der Tresor-Metadaten für einen bestimmten Tresor mithilfe der Low-Level-API von AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der sich der Tresor befindet. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `DescribeVaultRequest`-Klasse.

   Amazon Glacier (Amazon Glacier) verlangt von Ihnen die Angabe eines Tresornamens und Ihrer Konto-ID. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für .NET mit Amazon Glacier.

3. Führen Sie die `DescribeVault`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Die Tresor-Metadateninformationen, die Amazon Glacier zurückgibt, sind im `DescribeVaultResult` Objekt verfügbar.

Der folgende C#-Codeausschnitt veranschaulicht die vorherigen Schritte. Im Codeausschnitt werden Metadateninformationen eines vorhandenen Tresors in der Region „USA West (Oregon)“ abgerufen.

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);

DescribeVaultRequest describeVaultRequest = new DescribeVaultRequest()
{
  VaultName = "*** Provide vault name ***"
};
DescribeVaultResponse describeVaultResponse =
 client.DescribeVault(describeVaultRequest);
Console.WriteLine("\nVault description...");
Console.WriteLine(
   "\nVaultName: " + describeVaultResponse.VaultName +
   "\nVaultARN: " + describeVaultResponse.VaultARN +
   "\nVaultCreationDate: " + describeVaultResponse.CreationDate +
   "\nNumberOfArchives: " + describeVaultResponse.NumberOfArchives +
   "\nSizeInBytes: " + describeVaultResponse.SizeInBytes +
   "\nLastInventoryDate: " + describeVaultResponse.LastInventoryDate
   );
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Describe Vault (GET vault).

Abrufen von Tresor-Metadaten für alle Tresore in einer Region

Sie können die `ListVaults` Methode auch verwenden, um Metadaten für alle Tresore in einer bestimmten AWS Region abzurufen.

Mit dem folgenden C#-Codeausschnitt wird eine Liste der Tresore in der Region „USA West (Oregon)“ abgerufen. Die Anzahl der bei der Anforderung zurückgegebenen Tresore ist auf 5 beschränkt. Der Codeausschnitt führt dann eine Reihe von `ListVaults` Aufrufen durch, um die gesamte Tresorliste aus der Region abzurufen. AWS

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);
string lastMarker = null;
Console.WriteLine("\n List of vaults in your account in the specific AWS Region ...");
do
{
  ListVaultsRequest request = new ListVaultsRequest()
  {
    Limit = 5,
    Marker = lastMarker
  };
  ListVaultsResponse response = client.ListVaults(request);

  foreach (DescribeVaultOutput output in response.VaultList)
  {
    Console.WriteLine("Vault Name: {0} \tCreation Date: {1} \t #of archives: {2}",
                      output.VaultName, output.CreationDate, output.NumberOfArchives);
  }
  lastMarker = response.Marker;
} while (lastMarker != null);
```

Wenn Sie im vorherigen Codesegment den `Limit` Wert in der Anfrage nicht angeben, gibt Amazon Glacier bis zu 10 Tresore zurück, wie von der Amazon Glacier-API festgelegt.

Beachten Sie, dass die Informationen, die in der Liste für jeden Tresor zurückgegeben werden, mit den Informationen übereinstimmen, die Sie durch Aufrufen der `DescribeVault`-Methode für einen bestimmten Tresor erhalten.

> **Note**
>
> Durch die `ListVaults`-Methode wird die zugrunde liegende REST-API aufgerufen (siehe List Vaults (GET vaults)).

Beispiel: Abrufen von Tresor-Metadaten mithilfe der Low-Level-API von AWS SDK für .NET

Ein funktionierendes Codebeispiel finden Sie unter Beispiel: Vault-Operationen unter Verwendung der Low-Level-API des AWS SDK für .NET. Der folgende C#-Beispielcode erstellt einen Tresor und ruft die Tresor-Metadaten ab.

Abrufen von Tresormetadaten mithilfe der REST-API

Informationen zum Auflisten von Tresoren mithilfe der REST-API finden Sie unter List Vaults (GET vaults). Um einen Tresor zu beschreiben, siehe Describe Vault (GET vault).

Abrufen von Tresor-Metadaten in Amazon Glacier mithilfe der AWS Command Line Interface

Dieses Beispiel zeigt, wie Tresorinformationen und Metadaten in Amazon Glacier (Amazon Glacier) mithilfe von AWS Command Line Interface (AWS CLI) abgerufen werden.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Abrufen von Tresor-Metadaten mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface
2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

   ```
   aws help
   ```

- Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

```
aws glacier list-vaults --account-id 123456789012
```

- Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

```
aws configure list
```

Beispiel: Abrufen von Tresor-Metadaten mit dem AWS CLI

- Verwenden Sie den `describe-vault` Befehl, um einen Tresor mit dem Namen Account *awsexamplevault* *111122223333* zu beschreiben.

```
aws glacier describe-vault --vault-name awsexamplevault --account-id 111122223333
```

Ein Tresor-Inventar in Amazon Glacier herunterladen

Nachdem Sie Ihr erstes Archiv in Ihren Tresor hochgeladen haben, erstellt Amazon Glacier (Amazon Glacier) automatisch einen Tresorbestand und aktualisiert ihn dann etwa einmal täglich. Nachdem Amazon Glacier das erste Inventar erstellt hat, dauert es in der Regel einen halben Tag und bis zu einem Tag, bis dieses Inventar abgerufen werden kann. Sie können ein Tresorinventar mit den folgenden zwei Schritten von Amazon Glacier abrufen:

1. Initiieren Sie einen Inventarabrufauftrag mit der Operation Initiate Job (POST jobs).

> **Important**
>
> Eine Richtlinie für den Datenabruf kann dazu führen, dass die Anforderung zur Abrufinitiierung zu einer `PolicyEnforcedException`-Ausnahme führt und fehlschlägt. Weitere Informationen zu Datenabrufrichtlinien finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf. Weitere Informationen zur `PolicyEnforcedException`-Ausnahme finden Sie unter Fehlermeldungen.

2. Laden Sie nach Abschluss des Auftrags die Bytes mit der Operation Get Job Output (GET output) herunter.

Um beispielsweise ein Archiv oder einen Tresorbestand abzurufen, müssen Sie zunächst einen Abrufauftrag initiieren. Die Auftragsanforderung wird asynchron ausgeführt. Wenn Sie einen Abrufauftrag initiieren, erstellt Amazon Glacier einen Auftrag und gibt eine Auftrags-ID in der Antwort zurück. Wenn Amazon Glacier den Auftrag abgeschlossen hat, können Sie die Auftragsausgabe, die Archivbytes oder die Tresorbestandsdaten abrufen.

Der Auftrag muss abgeschlossen sein, bevor Sie die Ausgabe abrufen können. Um den Status des Auftrags festzustellen, haben Sie folgende Optionen:

- Auf die Benachrichtigung über den Abschluss des Auftrags warten — Sie können ein Amazon Simple Notification Service (Amazon SNS) -Thema angeben, an das Amazon Glacier nach Abschluss des Auftrags eine Benachrichtigung senden kann. Sie können das Amazon-SNS-Thema mit den folgenden Methoden angeben:
  - Geben Sie ein Amazon-SNS-Thema pro Auftragsbasis an.

    Wenn Sie einen Auftrag initiieren, können Sie optional ein Amazon-SNS-Thema angeben.
  - Richten Sie Benachrichtigungskonfiguration auf dem Tresor ein.

    Sie können die Benachrichtigungskonfiguration für bestimmte Ereignisse im Tresor einrichten (siehe Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier). Amazon Glacier sendet jedes Mal, wenn ein bestimmtes Ereignis eintritt, eine Nachricht an das angegebene SNS-Thema.

  Wenn Sie die Benachrichtigungskonfiguration für den Tresor eingerichtet haben und bei der Initiierung eines Auftrags auch ein Amazon SNS SNS-Thema angeben, sendet Amazon Glacier eine Nachricht über den Abschluss des Auftrags an beide Themen.

  Sie können das SNS-Thema so konfigurieren, dass Sie per E-Mail benachrichtigt werden oder die Nachricht in Amazon Simple Queue Service (Amazon SQS) gespeichert wird, die von Ihrer Anwendung abgefragt werden kann. Wenn eine Nachricht in der Warteschlange erscheint, können Sie überprüfen, ob der Auftrag erfolgreich abgeschlossen wurde, und dann die Auftragsausgabe herunterladen.
- Jobinformationen explizit anfordern — Amazon Glacier bietet auch einen Vorgang zur Beschreibung des Jobs (Describe Job (GET JobID)), mit dem Sie Jobinformationen abfragen

können. Sie können diese Anfrage periodisch senden, um Auftragsinformationen zu erhalten. Es wird allerdings empfohlen, Amazon-SNS-Benachrichtigungen zu verwenden.

> **Note**
>
> Die Informationen, die Sie per SNS-Benachrichtigung erhalten, sind identisch mit den Informationen, die Sie mit der Funktion zur Auftragsbeschreibung abrufen.

Themen

- Über das Inventar
- Herunterladen eines Tresorbestands in Amazon Glacier mithilfe der AWS SDK für Java
- Herunterladen eines Vault-Inventars in Amazon Glacier mithilfe der AWS SDK für .NET
- Herunterladen eines Tresorbestands mit der REST-API
- Herunterladen eines Vault-Inventars in Amazon Glacier mithilfe der AWS Command Line Interface

Über das Inventar

Amazon Glacier aktualisiert ein Tresorinventar mindestens einmal täglich, beginnend an dem Tag, an dem Sie ein Archiv zum ersten Mal in den Tresor hochladen. Wenn dem Tresor seit der letzten Inventur keine Archive hinzugefügt oder aus ihm gelöscht wurden, wird das Inventurdatum nicht aktualisiert. Wenn Sie einen Job für ein Tresorinventar initiieren, gibt Amazon Glacier das zuletzt generierte Inventar zurück. Dabei handelt es sich um einen point-in-time Snapshot und nicht um Echtzeitdaten. Beachten Sie, dass Amazon Glacier der Erstellung des ersten Inventars für den Tresor in der Regel einen halben Tag und bis zu einem Tag dauert, bis dieses Inventar abgerufen werden kann.

Möglicherweise finden Sie es nicht nützlich, einen Tresorbestand für jeden Archiv-Upload abzurufen. Nehmen wir jedoch an, dass Sie auf der Clientseite eine Datenbank verwalten, die Metadaten zu den Archiven verknüpft, die Sie auf Amazon Glacier hochladen. Dann finden Sie es möglicherweise nützlich, wenn die Tresorinventur die Informationen in Ihrer Datenbank nach Bedarf mit dem aktuellen Tresorbestand abgleicht. Sie können die Anzahl der abgerufenen Elemente des Tresorbestands begrenzen, indem Sie nach dem Archiverstellungsdatum filtern oder ein Kontingent festlegen. Weitere Informationen zur Begrenzung des Inventarabrufs finden Sie unter Inventarabfrage für einzelne Bereiche.

Das Inventar kann in zwei Formaten zurückgegeben werden: kommaseparierte Werte (CSV) oder JSON. Sie können das Format optional angeben, wenn Sie den Inventurauftrag initiieren. JSON ist das Standardformat. Weitere Informationen über die Datenfelder, die in bei einer Inventarabfrage zurückgegeben werden, finden Sie unter Antworttext der Get Job Output-API.

Herunterladen eines Tresorbestands in Amazon Glacier mithilfe der AWS SDK für Java

Nachfolgend werden die Schritte zum Abrufen eines Tresorbestands mithilfe der Low-Level-API für AWS SDK für Java beschrieben. Die High-Level-API unterstützt das Abrufen von Tresorbeständen nicht.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der sich der Tresor befindet. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Initiieren Sie durch Ausführen der `initiateJob`-Methode einen Auftrag zum Abrufen des Inventars.

   Führen Sie `initiateJob` aus, indem Sie Auftragsinformationen in einem `InitiateJobRequest`-Objekt bereitstellen.

   > **Note**
   >
   > Beachten Sie, dass wenn noch keine Inventur für den Tresor abgeschlossen wurde, ein Fehler zurückgegeben wurde. Amazon Glacier (Amazon Glacier) erstellt in regelmäßigen Abständen, alle 24 Stunden, eine Bestandsaufnahme für jeden Tresor.

   Amazon Glacier gibt als Antwort eine Job-ID zurück. Die Antwort ist in einer Instance der `InitiateJobResult`-Klasse verfügbar.

```
InitiateJobRequest initJobRequest = new InitiateJobRequest()
    .withVaultName("*** provide vault name ***")
    .withJobParameters(
            new JobParameters()
                .withType("inventory-retrieval")
                .withSNSTopic("*** provide SNS topic ARN ****")
```

```
        );

InitiateJobResult initJobResult = client.initiateJob(initJobRequest);
String jobId = initJobResult.getJobId();
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie entweder eine Benachrichtigungskonfiguration für den Tresor eingerichtet oder bei der Initiierung des Jobs ein Amazon Simple Notification Service (Amazon SNS) -Thema angegeben haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an das Thema.

   Sie können Amazon Glacier auch abfragen, indem Sie die `describeJob` Methode aufrufen, um den Status der Auftragsabwicklung zu ermitteln. Allerdings wird die Verwendung eines Amazon-SNS-Themas für Benachrichtigungen empfohlen. Das Codebeispiel im folgenden Abschnitt verwendet Amazon SNS for Amazon Glacier, um eine Nachricht zu veröffentlichen.

4. Laden Sie die Auftragsausgabe (Tresorbestandsdaten) herunter, indem Sie die `getJobOutput`-Methode ausführen.

   Sie geben Ihre Konto-ID, Auftrags-ID und den Namen des Tresors an, indem Sie eine Instance der Klasse `GetJobOutputRequest` erstellen. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anfrage verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier.

   Die Ausgabe, die Amazon Glacier zurückgibt, ist im `GetJobOutputResult` Objekt verfügbar.

```
GetJobOutputRequest jobOutputRequest = new GetJobOutputRequest()
        .withVaultName("*** provide vault name ***")
        .withJobId("*** provide job ID ***");
GetJobOutputResult jobOutputResult = client.getJobOutput(jobOutputRequest);
// jobOutputResult.getBody(); provides the output stream.
```

> **Note**
>
> Informationen zu der REST-API, die dem Auftrag zugrunde liegt, finden Sie unter Auftragsoperationen.

Beispiel: Abrufen eines Tresorbestands mithilfe des Amazon-SDK für Java

Im folgenden Java-Codebeispiel wird der Tresorbestand vom angegebenen Tresor abgerufen.

Das Beispiel führt die folgenden Aufgaben durch:

- Erstellt ein Amazon Simple Notification Service (Amazon SNS)-Thema.

  Amazon Glacier sendet nach Abschluss des Auftrags eine Benachrichtigung zu diesem Thema.
- Erstellt eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange.

  Im Beispiel wird der Warteschlange eine Richtlinie angefügt, damit das Amazon-SNS-Thema Nachrichten in der Warteschlange veröffentlichen kann.
- Initiiert einen Auftrag zum Herunterladen des angegebenen Archivs.

  In der Jobanfrage ist das Amazon SNS SNS-Thema angegeben, das erstellt wurde, sodass Amazon Glacier nach Abschluss des Jobs eine Benachrichtigung zu dem Thema veröffentlichen kann.
- Prüft die Amazon-SQS-Warteschlange auf eine Nachricht, die die Auftrags-ID enthält.

  Ist eine Nachricht vorhanden, parsen Sie das JSON und prüfen Sie, ob der Auftrag erfolgreich abgeschlossen wurde. Wenn eine Nachricht enthalten ist, laden Sie das Archiv herunter.
- Bereinigt das System durch Löschen des Amazon-SNS-Themas und der erstellten Amazon-SQS-Warteschlange.

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
import com.fasterxml.jackson.core.JsonFactory;
import com.fasterxml.jackson.core.JsonParseException;
import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
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.SQSActions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.GetJobOutputRequest;
import com.amazonaws.services.glacier.model.GetJobOutputResult;
import com.amazonaws.services.glacier.model.InitiateJobRequest;
import com.amazonaws.services.glacier.model.InitiateJobResult;
import com.amazonaws.services.glacier.model.JobParameters;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sns.model.DeleteTopicRequest;
import com.amazonaws.services.sns.model.SubscribeRequest;
import com.amazonaws.services.sns.model.SubscribeResult;
import com.amazonaws.services.sns.model.UnsubscribeRequest;
import com.amazonaws.services.sqs.AmazonSQSClient;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.CreateQueueResult;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesResult;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;


public class AmazonGlacierDownloadInventoryWithSQSPolling {

    public static String vaultName = "*** provide vault name ***";
    public static String snsTopicName = "*** provide topic name ***";
    public static String sqsQueueName = "*** provide queue name ***";
```

```
    public static String sqsQueueARN;
    public static String sqsQueueURL;
    public static String snsTopicARN;
    public static String snsSubscriptionARN;
    public static String fileName = "*** provide file name ***";
    public static String region = "*** region ***";
    public static long sleepTime = 600;
    public static AmazonGlacierClient client;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier." + region + ".amazonaws.com");
        sqsClient = new AmazonSQSClient(credentials);
        sqsClient.setEndpoint("https://sqs." + region + ".amazonaws.com");
        snsClient = new AmazonSNSClient(credentials);
        snsClient.setEndpoint("https://sns." + region + ".amazonaws.com");

        try {
            setupSQS();

            setupSNS();

            String jobId = initiateJobRequest();
            System.out.println("Jobid = " + jobId);

            Boolean success = waitForJobToComplete(jobId, sqsQueueURL);
            if (!success) { throw new Exception("Job did not complete
successfully."); }

            downloadJobOutput(jobId);

            cleanUp();

        } catch (Exception e) {
            System.err.println("Inventory retrieval failed.");
            System.err.println(e);
        }
    }
```

```
    private static void setupSQS() {
        CreateQueueRequest request = new CreateQueueRequest()
            .withQueueName(sqsQueueName);
        CreateQueueResult result = sqsClient.createQueue(request);
        sqsQueueURL = result.getQueueUrl();

        GetQueueAttributesRequest qRequest = new GetQueueAttributesRequest()
            .withQueueUrl(sqsQueueURL)
            .withAttributeNames("QueueArn");

        GetQueueAttributesResult qResult = sqsClient.getQueueAttributes(qRequest);
        sqsQueueARN = qResult.getAttributes().get("QueueArn");

        Policy sqsPolicy =
            new Policy().withStatements(
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueARN)));
        Map<String, String> queueAttributes = new HashMap<String, String>();
        queueAttributes.put("Policy", sqsPolicy.toJson());
        sqsClient.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueURL,
queueAttributes));

    }
    private static void setupSNS() {
        CreateTopicRequest request = new CreateTopicRequest()
            .withName(snsTopicName);
        CreateTopicResult result = snsClient.createTopic(request);
        snsTopicARN = result.getTopicArn();

        SubscribeRequest request2 = new SubscribeRequest()
            .withTopicArn(snsTopicARN)
            .withEndpoint(sqsQueueARN)
            .withProtocol("sqs");
        SubscribeResult result2 = snsClient.subscribe(request2);

        snsSubscriptionARN = result2.getSubscriptionArn();
    }
    private static String initiateJobRequest() {

        JobParameters jobParameters = new JobParameters()
            .withType("inventory-retrieval")
            .withSNSTopic(snsTopicARN);
```

```
        InitiateJobRequest request = new InitiateJobRequest()
            .withVaultName(vaultName)
            .withJobParameters(jobParameters);

        InitiateJobResult response = client.initiateJob(request);

        return response.getJobId();
    }

    private static Boolean waitForJobToComplete(String jobId, String sqsQueueUrl)
throws InterruptedException, JsonParseException, IOException {

        Boolean messageFound = false;
        Boolean jobSuccessful = false;
        ObjectMapper mapper = new ObjectMapper();
        JsonFactory factory = mapper.getFactory();

        while (!messageFound) {
            List<Message> msgs = sqsClient.receiveMessage(
               new
ReceiveMessageRequest(sqsQueueUrl).withMaxNumberOfMessages(10)).getMessages();

            if (msgs.size() > 0) {
                for (Message m : msgs) {
                    JsonParser jpMessage = factory.createJsonParser(m.getBody());
                    JsonNode jobMessageNode = mapper.readTree(jpMessage);
                    String jobMessage = jobMessageNode.get("Message").textValue();

                    JsonParser jpDesc = factory.createJsonParser(jobMessage);
                    JsonNode jobDescNode = mapper.readTree(jpDesc);
                    String retrievedJobId = jobDescNode.get("JobId").textValue();
                    String statusCode = jobDescNode.get("StatusCode").textValue();
                    if (retrievedJobId.equals(jobId)) {
                        messageFound = true;
                        if (statusCode.equals("Succeeded")) {
                            jobSuccessful = true;
                        }
                    }
                }

            } else {
              Thread.sleep(sleepTime * 1000);
            }
```

```
          }
        return (messageFound && jobSuccessful);
    }

    private static void downloadJobOutput(String jobId) throws IOException {

        GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
            .withVaultName(vaultName)
            .withJobId(jobId);
        GetJobOutputResult getJobOutputResult =
 client.getJobOutput(getJobOutputRequest);

        FileWriter fstream = new FileWriter(fileName);
        BufferedWriter out = new BufferedWriter(fstream);
        BufferedReader in = new BufferedReader(new
 InputStreamReader(getJobOutputResult.getBody()));
        String inputLine;
        try {
            while ((inputLine = in.readLine()) != null) {
                out.write(inputLine);
            }
        }catch(IOException e) {
            throw new AmazonClientException("Unable to save archive", e);
        }finally{
            try {in.close();}  catch (Exception e) {}
            try {out.close();}  catch (Exception e) {}
        }
        System.out.println("Retrieved inventory to " + fileName);
    }

    private static void cleanUp() {
        snsClient.unsubscribe(new UnsubscribeRequest(snsSubscriptionARN));
        snsClient.deleteTopic(new DeleteTopicRequest(snsTopicARN));
        sqsClient.deleteQueue(new DeleteQueueRequest(sqsQueueURL));
    }
}
```

Herunterladen eines Vault-Inventars in Amazon Glacier mithilfe der AWS SDK für .NET

Nachfolgend werden die Schritte zum Abrufen eines Tresorbestands mithilfe der Low-Level-API für AWS SDK für .NET beschrieben. Die High-Level-API unterstützt das Abrufen von Tresorbeständen nicht.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der sich der Tresor befindet. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Initiieren Sie durch Ausführen der `InitiateJob`-Methode einen Auftrag zum Abrufen des Inventars.

   Sie geben Auftragsinformationen in einem `InitiateJobRequest`-Objekt an. Amazon Glacier (Amazon Glacier) gibt als Antwort eine Job-ID zurück. Die Antwort ist in einer Instance der `InitiateJobResponse`-Klasse verfügbar.

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);

InitiateJobRequest initJobRequest = new InitiateJobRequest()
{
  VaultName = vaultName,
  JobParameters = new JobParameters()
  {
    Type = "inventory-retrieval",
    SNSTopic = "*** Provide Amazon SNS topic arn ***",
  }
};
InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
string jobId = initJobResponse.JobId;
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie entweder im Tresor eine Benachrichtigungskonfiguration eingerichtet haben, die ein Amazon Simple Notification Service (Amazon SNS) -Thema identifiziert, oder ein Amazon SNS-Thema angegeben haben, als Sie einen Job initiiert haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an

dieses Thema. Das Codebeispiel im folgenden Abschnitt verwendet Amazon SNS for Amazon Glacier, um eine Nachricht zu veröffentlichen.

Sie können Amazon Glacier auch abfragen, indem Sie die `DescribeJob` Methode aufrufen, um den Status der Auftragsabwicklung zu ermitteln. Allerdings wird die Verwendung eines Amazon-SNS-Themas für Benachrichtigungen empfohlen.

4. Laden Sie die Auftragsausgabe (Tresorbestandsdaten) herunter, indem Sie die `GetJobOutput`-Methode ausführen.

   Sie erstellen eine Instance der `GetJobOutputRequest`-Klasse, um Ihre Konto-ID, den Tresornamen und die Auftrags-ID bereitzustellen. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für .NET mit Amazon Glacier.

   Die Ausgabe, die Amazon Glacier zurückgibt, ist im `GetJobOutputResponse` Objekt verfügbar.

```
GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
{
  JobId = jobId,
  VaultName = vaultName
};

GetJobOutputResponse getJobOutputResponse =
 client.GetJobOutput(getJobOutputRequest);
using (Stream webStream = getJobOutputResponse.Body)
{
   using (Stream fileToSave = File.OpenWrite(fileName))
   {
     CopyStream(webStream, fileToSave);
   }
}
```

> **Note**
>
> Informationen zu der REST-API, die dem Auftrag zugrunde liegt, finden Sie unter Auftragsoperationen.

Beispiel: Abrufen eines Vault-Inventars mithilfe der Low-Level-API von AWS SDK für .NET

Im folgenden C#-Codebeispiel wird der Tresorbestand vom angegebenen Tresor abgerufen.

Das Beispiel führt die folgenden Aufgaben durch:

- Einrichten eines Amazon-SNS-Themas.

  Amazon Glacier sendet nach Abschluss des Auftrags eine Benachrichtigung zu diesem Thema.
- Einrichten einer Amazon-SQS-Warteschlange.

  Im Beispiel wird der Warteschlange eine Richtlinie angefügt, damit das Amazon-SNS-Thema Nachrichten veröffentlichen kann.
- Initiieren eines Auftrags zum Herunterladen des angegebenen Archivs.

  In der Jobanfrage gibt das Beispiel das Amazon SNS SNS-Thema an, sodass Amazon Glacier nach Abschluss des Auftrags eine Nachricht senden kann.
- Regelmäßiges Überprüfen der Amazon-SQS-Warteschlange auf Nachrichten.

  Ist eine Nachricht vorhanden, parsen Sie das JSON und prüfen Sie, ob der Auftrag erfolgreich abgeschlossen wurde. Wenn eine Nachricht enthalten ist, laden Sie das Archiv herunter. Im Codebeispiel wird die JSON.NET-Bibliothek (siehe JSON.NET) zum Analysieren von JSON verwendet.
- Bereinigung durch Löschen des Amazon-SNS-Themas und der erstellten Amazon-SQS-Warteschlange.

Example

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
```

```
using Amazon.SQS;
using Amazon.SQS.Model;
using Newtonsoft.Json;

namespace glacier.amazon.com.docsamples
{
  class VaultInventoryJobLowLevelUsingSNSSQS
  {
    static string topicArn;
    static string queueUrl;
    static string queueArn;
    static string vaultName = "*** Provide vault name ***";
    static string fileName  = "*** Provide file name and path where to store inventory
 ***";
    static AmazonSimpleNotificationServiceClient snsClient;
    static AmazonSQSClient sqsClient;
    const string SQS_POLICY =
        "{" +
        "    \"Version\" : \"2012-10-17\",&TCX5-2025-waiver;" +
        "    \"Statement\" : [" +
        "        {" +
        "            \"Sid\" : \"sns-rule\"," +
        "            \"Effect\" : \"Allow\"," +
        "            \"Principal\" : {\"AWS\" : \"arn:aws:iam::123456789012:root\" },"
 +
        "            \"Action\"    : \"sqs:SendMessage\"," +
        "            \"Resource\"  : \"{QuernArn}\"," +
        "            \"Condition\" : {" +
        "                \"ArnLike\" : {" +
        "                    \"aws:SourceArn\" : \"{TopicArn}\"" +
        "                }" +
        "            }" +
        "        }" +
        "    ]" +
        "}";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
            Console.WriteLine("Setup SNS topic and SQS queue.");
```

```
            SetupTopicAndQueue();
            Console.WriteLine("To continue, press Enter"); Console.ReadKey();

            Console.WriteLine("Retrieve Inventory List");
            GetVaultInventory(client);
        }
        Console.WriteLine("Operations successful.");
        Console.WriteLine("To continue, press Enter"); Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      finally
      {
       // Delete SNS topic and SQS queue.
       snsClient.DeleteTopic(new DeleteTopicRequest() { TopicArn = topicArn });
       sqsClient.DeleteQueue(new DeleteQueueRequest() { QueueUrl = queueUrl });
      }
    }

    static void SetupTopicAndQueue()
    {
      long ticks = DateTime.Now.Ticks;

      // Setup SNS topic.
      snsClient = new
AmazonSimpleNotificationServiceClient(Amazon.RegionEndpoint.USWest2);
      sqsClient = new AmazonSQSClient(Amazon.RegionEndpoint.USWest2);

      topicArn = snsClient.CreateTopic(new CreateTopicRequest { Name =
"GlacierDownload-" + ticks }).TopicArn;
      Console.Write("topicArn: "); Console.WriteLine(topicArn);

      CreateQueueRequest createQueueRequest =  new CreateQueueRequest();
      createQueueRequest.QueueName = "GlacierDownload-" + ticks;
      CreateQueueResponse createQueueResponse =
sqsClient.CreateQueue(createQueueRequest);
      queueUrl = createQueueResponse.QueueUrl;
      Console.Write("QueueURL: "); Console.WriteLine(queueUrl);

      GetQueueAttributesRequest getQueueAttributesRequest = new
GetQueueAttributesRequest();
      getQueueAttributesRequest.AttributeNames = new List<string> { "QueueArn" };
      getQueueAttributesRequest.QueueUrl = queueUrl;
```

```
      GetQueueAttributesResponse response =
sqsClient.GetQueueAttributes(getQueueAttributesRequest);
      queueArn = response.QueueARN;
      Console.Write("QueueArn: ");Console.WriteLine(queueArn);

      // Setup the Amazon SNS topic to publish to the SQS queue.
      snsClient.Subscribe(new SubscribeRequest()
      {
        Protocol = "sqs",
        Endpoint = queueArn,
        TopicArn = topicArn
      });

      // Add the policy to the queue so SNS can send messages to the queue.
      var policy = SQS_POLICY.Replace("{TopicArn}", topicArn).Replace("{QuernArn}",
queueArn);

      sqsClient.SetQueueAttributes(new SetQueueAttributesRequest()
      {
          QueueUrl = queueUrl,
          Attributes = new Dictionary<string, string>
          {
              { QueueAttributeName.Policy, policy }
          }
      });

   }

   static void GetVaultInventory(AmazonGlacierClient client)
   {
     // Initiate job.
     InitiateJobRequest initJobRequest = new InitiateJobRequest()
     {
       VaultName = vaultName,
       JobParameters = new JobParameters()
       {
         Type = "inventory-retrieval",
         Description = "This job is to download a vault inventory.",
         SNSTopic = topicArn,
       }
     };

     InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
     string jobId = initJobResponse.JobId;
```

```
     // Check queue for a message and if job completed successfully, download
inventory.
     ProcessQueue(jobId, client);
   }

   private static void ProcessQueue(string jobId, AmazonGlacierClient client)
   {
     ReceiveMessageRequest receiveMessageRequest = new ReceiveMessageRequest()
{ QueueUrl = queueUrl, MaxNumberOfMessages = 1 };
     bool jobDone = false;
     while (!jobDone)
     {
       Console.WriteLine("Poll SQS queue");
       ReceiveMessageResponse receiveMessageResponse =
sqsClient.ReceiveMessage(receiveMessageRequest);
       if (receiveMessageResponse.Messages.Count == 0)
       {
         Thread.Sleep(10000 * 60);
         continue;
       }
       Console.WriteLine("Got message");
       Message message = receiveMessageResponse.Messages[0];
       Dictionary<string, string> outerLayer =
JsonConvert.DeserializeObject<Dictionary<string, string>>(message.Body);
       Dictionary<string, object> fields =
JsonConvert.DeserializeObject<Dictionary<string, object>>(outerLayer["Message"]);
       string statusCode = fields["StatusCode"] as string;

       if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_SUCCEEDED,
StringComparison.InvariantCultureIgnoreCase))
       {
         Console.WriteLine("Downloading job output");
         DownloadOutput(jobId, client); // Save job output to the specified file
location.
       }
       else if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_FAILED,
StringComparison.InvariantCultureIgnoreCase))
         Console.WriteLine("Job failed... cannot download the inventory.");

       jobDone = true;
       sqsClient.DeleteMessage(new DeleteMessageRequest() { QueueUrl = queueUrl,
ReceiptHandle = message.ReceiptHandle });
     }
```

```
    }

    private static void DownloadOutput(string jobId, AmazonGlacierClient client)
    {
      GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
      {
        JobId = jobId,
        VaultName = vaultName
      };

      GetJobOutputResponse getJobOutputResponse =
 client.GetJobOutput(getJobOutputRequest);
      using (Stream webStream = getJobOutputResponse.Body)
      {
        using (Stream fileToSave = File.OpenWrite(fileName))
        {
          CopyStream(webStream, fileToSave);
        }
      }
    }

    public static void CopyStream(Stream input, Stream output)
    {
      byte[] buffer = new byte[65536];
      int length;
      while ((length = input.Read(buffer, 0, buffer.Length)) > 0)
      {
        output.Write(buffer, 0, length);
      }
    }
  }
}
```

Herunterladen eines Tresorbestands mit der REST-API

Herunterladen eines Tresorbestands mit der REST-API

Das Herunterladen eines Tresorbestands erfolgt in zwei Schritten.

1. Initiieren Sie einen Auftrag des Typs `inventory-retrieval`. Weitere Informationen finden Sie unter Initiate Job (POST jobs).

2. Laden Sie die Inventardaten herunter, nachdem der Auftrag abgeschlossen ist. Weitere Informationen finden Sie unter Get Job Output (GET output).

Herunterladen eines Vault-Inventars in Amazon Glacier mithilfe der AWS Command Line Interface

Gehen Sie wie folgt vor, um mithilfe von () ein Tresorinventar in Amazon Glacier AWS Command Line Interface (Amazon Glacier AWS CLI) herunterzuladen.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Herunterladen eines Tresorinventars mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface

2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

```
aws configure list
```

Beispiel: Herunterladen eines Tresorinventars mit dem AWS CLI

1. Verwenden Sie den `initiate-job`-Befehl, um eine Lagerabrufaufgabe zu starten.

   ```
   aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
   job-parameters='{"Type": "inventory-retrieval"}'
   ```

   Erwartete Ausgabe:

   ```
   {
       "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
       "jobId": "*** jobid ***"
   }
   ```

2. Verwenden Sie den `describe-job`-Befehl, um den Status der vorherigen Abrufaufgabe zu überprüfen.

   ```
   aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
   job-id *** jobid ***
   ```

   Erwartete Ausgabe:

   ```
   {
       "InventoryRetrievalParameters": {
           "Format": "JSON"
       },
       "VaultARN": "*** vault arn ***",
       "Completed": false,
       "JobId": "*** jobid ***",
       "Action": "InventoryRetrieval",
       "CreationDate": "*** job creation date ***",
       "StatusCode": "InProgress"
   }
   ```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Die Job-ID läuft mindestens 24 Stunden nach Abschluss des Auftrags durch Amazon Glacier nicht ab. Wenn Sie entweder eine Benachrichtigungskonfiguration für den Tresor eingerichtet oder bei der Initiierung des Auftrags ein Amazon Simple Notification Service (Amazon SNS) -Thema angegeben haben, sendet Amazon Glacier nach Abschluss des Auftrags eine Nachricht an das Thema.

   Sie können die Benachrichtigungskonfiguration für bestimmte Ereignisse auf dem Tresor einrichten. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier. Amazon Glacier sendet jedes Mal, wenn die spezifischen Ereignisse eintreten, eine Nachricht an das angegebene SNS-Thema.

4. Nach Abschluss laden Sie die Abrufaufgabe mit dem `get-job-output`-Befehl in die Datei „`output.json`“ herunter.

   ```
   aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
    --job-id *** jobid *** output.json
   ```

   Dieser Befehl erzeugt eine Datei mit den folgenden Feldern.

   ```
   {
   "VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
   "InventoryDate":"*** job completion date ***",
   "ArchiveList":[
   {"ArchiveId":"*** archiveid ***",
   "ArchiveDescription":"*** archive description (if set) ***",
   "CreationDate":"*** archive creation date ***",
   "Size":"*** archive size (in bytes) ***",
   "SHA256TreeHash":"*** archive hash ***"
   }
   {"ArchiveId":
   ...
   ]}
   ```

Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier

Das Abrufen von Daten aus Amazon Glacier, z. B. eines Archivs aus einem Tresor oder einem Tresorbestand, erfolgt in zwei Schritten.

1. Initiieren Sie einen Abrufauftrag.
2. Sie können die Auftragsausgabe herunterladen, nachdem der Auftrag abgeschlossen wurde.

Sie können eine Benachrichtigungskonfiguration für einen Tresor einrichten, sodass nach Abschluss eines Auftrags eine Nachricht an ein Amazon Simple Notification Service (Amazon SNS)-Thema gesendet wird.

Themen

- Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier: Allgemeine Konzepte
- Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier mithilfe der AWS SDK für Java
- Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier mithilfe der AWS SDK für .NET
- Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier mithilfe der REST-API
- Konfiguration von Vault-Benachrichtigungen mithilfe der Amazon Glacier-Konsole
- Konfiguration von Tresor-Benachrichtigungen mit dem AWS Command Line Interface

Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier: Allgemeine Konzepte

Eine Amazon Glacier-Abruf-Job-Anfrage wird asynchron ausgeführt. Sie müssen warten, bis Amazon Glacier den Job abgeschlossen hat, bevor Sie seine Ausgabe abrufen können. Sie können Amazon Glacier regelmäßig abfragen, um den Auftragsstatus zu ermitteln, aber das ist kein optimaler Ansatz. Amazon Glacier unterstützt auch Benachrichtigungen. Wenn ein Auftrag abgeschlossen wurde, kann er eine Mitteilung an ein Amazon Simple Notification Service (Amazon SNS)-Thema senden. Dafür müssen Sie eine Benachrichtigungskonfiguration für den Tresor einrichten. In der Konfiguration identifizieren Sie ein oder mehrere Ereignisse und ein Amazon SNS SNS-Thema, an das Amazon Glacier eine Nachricht senden soll, wenn das Ereignis eintritt.

Amazon Glacier definiert Ereignisse, die sich speziell auf den Abschluss von Jobs (`ArchiveRetrievalCompleted`,`InventoryRetrievalCompleted`) beziehen und die Sie

zur Benachrichtigungskonfiguration des Tresors hinzufügen können. Wenn ein bestimmter Job abgeschlossen ist, veröffentlicht Amazon Glacier eine Benachrichtigung zum SNS-Thema.

Die Benachrichtigungskonfiguration ist ein JSON-Dokument, wie im folgenden Beispiel gezeigt.

```
{
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

Sie können für einen Tresor nur ein Amazon-SNS-Thema konfigurieren.

> **Note**
>
> Durch das Hinzufügen einer Benachrichtigungskonfiguration zu einem Tresor sendet Amazon Glacier jedes Mal eine Benachrichtigung, wenn das in der Benachrichtigungskonfiguration angegebene Ereignis eintritt. Optional können Sie auch in jeder Anforderung zur Auftragsinitiierung ein Amazon-SNS-Thema angeben. Wenn Sie sowohl die Benachrichtigungskonfiguration zum Tresor hinzufügen als auch ein Amazon SNS SNS-Thema in Ihrer Job-Initiation-Anfrage angeben, sendet Amazon Glacier beide Benachrichtigungen.

Die Auftragsabschlussnachricht, die Amazon Glacier sendet, enthält Informationen wie die Art des Auftrags (`InventoryRetrieval`,`ArchiveRetrieval`), den Auftragsabschlussstatus, den SNS-Themennamen, den Jobstatuscode und den Vault-ARN. Im Folgenden finden Sie ein Beispiel für eine Benachrichtigung, die Amazon Glacier nach Abschluss eines `InventoryRetrieval` Auftrags an ein SNS-Thema gesendet hat.

```
{
 "Action": "InventoryRetrieval",
 "ArchiveId": null,
 "ArchiveSizeInBytes": null,
 "Completed": true,
 "CompletionDate": "2012-06-12T22:20:40.790Z",
 "CreationDate": "2012-06-12T22:20:36.814Z",
 "InventorySizeInBytes":11693,
 "JobDescription": "my retrieval job",
 "JobId":"HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID",
```

```
  "SHA256TreeHash":null,
  "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
  "StatusCode":"Succeeded",
  "StatusMessage": "Succeeded",
  "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
}
```

Wenn das Feld `Completed` wahr ist, müssen Sie auch anhand des `StatusCode` überprüfen, ob der Job erfolgreich beendet wurde oder fehlgeschlagen ist.

> **Note**
>
> Das Amazon-SNS-Thema muss dem Tresor erlauben, eine Benachrichtigung zu veröffentlichen. Standardmäßig kann nur der Eigentümer des Amazon-SNS-Themas eine Nachricht im Thema veröffentlichen. Wenn das Amazon SNS SNS-Thema und der Tresor jedoch unterschiedlichen Eigentümern gehören AWS-Konten, müssen Sie das Amazon SNS SNS-Thema so konfigurieren, dass Veröffentlichungen aus dem Tresor akzeptiert werden. Sie können die Richtlinie für das Amazon-SNS-Thema in der Amazon-SNS-Konsole konfigurieren.

Weitere Informationen zu Amazon SNS finden Sie unter Erste Schritte mit Amazon SNS.

Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier mithilfe der AWS SDK für Java

Nachfolgend werden die Schritte zum Konfigurieren von Benachrichtigungen auf einem Tresor mithilfe der Low-Level-API für das AWS SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der sich der Tresor befindet. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Stellen Sie Informationen zur Benachrichtigungskonfiguration bereit, indem Sie eine Instance der `SetVaultNotificationsRequest`-Klasse erstellen.

   Sie müssen den Tresornamen, Informationen zur Benachrichtigungskonfiguration und die Konto-ID bereitstellen. Wenn Sie eine Benachrichtigungskonfiguration angeben, stellen Sie den Amazon-Ressourcennamen (ARN) eines Amazon-SNS-Themas und eines oder mehrere Ereignisse bereit,

bei denen Sie benachrichtigt werden möchten. Eine Liste der unterstützten Ereignisse finden Sie unter Set Vault Notification Configuration (PUT notification-configuration).

3. Führen Sie die `setVaultNotifications`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

Der folgende Java-Codeausschnitt veranschaulicht die vorherigen Schritte. Der Ausschnitt richtet eine Benachrichtigungskonfiguration auf einem Tresor ein. Die Konfiguration fordert Amazon Glacier (Amazon Glacier) auf, eine Benachrichtigung an das angegebene Amazon SNS SNS-Thema zu senden, wenn entweder das `ArchiveRetrievalCompleted` Ereignis oder das `InventoryRetrievalCompleted` Ereignis eintritt.

```
SetVaultNotificationsRequest request = new SetVaultNotificationsRequest()
        .withAccountId("-")
        .withVaultName("*** provide vault name ***")
        .withVaultNotificationConfig(
                new VaultNotificationConfig()
                .withSNSTopic("*** provide SNS topic ARN ***")
                .withEvents("ArchiveRetrievalCompleted", "InventoryRetrievalCompleted")
         );
client.setVaultNotifications(request);
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Tresor-Operationen.

Beispiel: Einstellung der Benachrichtigungskonfiguration in einem Tresor mithilfe von AWS SDK für Java

Das folgende Java-Codebeispiel richtet Benachrichtigungskonfiguration für einen Tresor ein, löscht die Konfiguration und stellt die Konfiguration wieder her. step-by-stepAnweisungen zur Ausführung des folgenden Beispiels finden Sie unterVerwenden von AWS SDK für Java mit Amazon Glacier.

Example

```
import java.io.IOException;
```

```
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.DeleteVaultNotificationsRequest;
import com.amazonaws.services.glacier.model.GetVaultNotificationsRequest;
import com.amazonaws.services.glacier.model.GetVaultNotificationsResult;
import com.amazonaws.services.glacier.model.SetVaultNotificationsRequest;
import com.amazonaws.services.glacier.model.VaultNotificationConfig;


public class AmazonGlacierVaultNotifications {

    public static AmazonGlacierClient client;
    public static String vaultName = "*** provide vault name ****";
    public static String snsTopicARN = "*** provide sns topic ARN ***";

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        try {

            System.out.println("Adding notification configuration to the vault.");
            setVaultNotifications();
            getVaultNotifications();
            deleteVaultNotifications();

        } catch (Exception e) {
            System.err.println("Vault operations failed." + e.getMessage());
        }
    }

    private static void setVaultNotifications() {
        VaultNotificationConfig config = new VaultNotificationConfig()
            .withSNSTopic(snsTopicARN)
            .withEvents("ArchiveRetrievalCompleted", "InventoryRetrievalCompleted");

        SetVaultNotificationsRequest request = new SetVaultNotificationsRequest()
                .withVaultName(vaultName)
                .withVaultNotificationConfig(config);

        client.setVaultNotifications(request);
```

```
        System.out.println("Notification configured for vault: " + vaultName);
    }

    private static void getVaultNotifications() {
        VaultNotificationConfig notificationConfig = null;
        GetVaultNotificationsRequest request = new GetVaultNotificationsRequest()
                .withVaultName(vaultName);
        GetVaultNotificationsResult result = client.getVaultNotifications(request);
        notificationConfig = result.getVaultNotificationConfig();

        System.out.println("Notifications configuration for vault: "
                + vaultName);
        System.out.println("Topic: " + notificationConfig.getSNSTopic());
        System.out.println("Events: " + notificationConfig.getEvents());
    }

    private static void deleteVaultNotifications() {
            DeleteVaultNotificationsRequest request = new
 DeleteVaultNotificationsRequest()
                .withVaultName(vaultName);
            client.deleteVaultNotifications(request);
            System.out.println("Notifications configuration deleted for vault: " +
 vaultName);
    }
}
```

Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier mithilfe der AWS SDK für .NET

Nachfolgend werden die Schritte zum Konfigurieren von Benachrichtigungen auf einem Tresor mithilfe der Low-Level-API für das AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der sich der Tresor befindet. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Stellen Sie Informationen zur Benachrichtigungskonfiguration bereit, indem Sie eine Instance der `SetVaultNotificationsRequest`-Klasse erstellen.

   Sie müssen den Tresornamen, Informationen zur Benachrichtigungskonfiguration und die Konto-ID bereitstellen. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den

Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für .NET mit Amazon Glacier.

Wenn Sie eine Benachrichtigungskonfiguration angeben, stellen Sie den Amazon-Ressourcennamen (ARN) eines Amazon-SNS-Themas und eines oder mehrere Ereignisse bereit, bei denen Sie benachrichtigt werden möchten. Eine Liste der unterstützten Ereignisse finden Sie unter Set Vault Notification Configuration (PUT notification-configuration).

3. Führen Sie die `SetVaultNotifications`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.
4. Nachdem Sie die Benachrichtigungskonfiguration für einen Tresor eingerichtet haben, können Sie die Konfigurationsinformationen abrufen, indem Sie die Methode `GetVaultNotifications` aufrufen und sie entfernen, indem Sie die vom Client bereitgestellte Methode `DeleteVaultNotifications` aufrufen.

Beispiel: Einstellung der Benachrichtigungskonfiguration in einem Tresor mithilfe von AWS SDK für .NET

Im folgenden C#-Codebeispiel werden die vorstehenden Schritte veranschaulicht. Im Beispiel wird eine Benachrichtigungskonfiguration für den Tresor „`examplevault`“ in der Region „USA West (Oregon)“ eingerichtet, die Konfiguration abgerufen und dann gelöscht. Die Konfiguration fordert Amazon Glacier (Amazon Glacier) auf, eine Benachrichtigung an das angegebene Amazon SNS SNS-Thema zu senden, wenn entweder das `ArchiveRetrievalCompleted` Ereignis oder das `InventoryRetrievalCompleted` Ereignis eintritt.

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Tresor-Operationen.

step-by-stepAnweisungen zur Ausführung des folgenden Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt aktualisieren und einen vorhandenen Tresornamen und ein Amazon-SNS-Thema angeben.

Example

```
using System;
using System.Collections.Generic;
```

```
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class VaultNotificationSetGetDelete
  {
    static string vaultName   = "examplevault";
    static string snsTopicARN = "*** Provide Amazon SNS topic ARN ***";

    static IAmazonGlacier client;

    public static void Main(string[] args)
    {
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Adding notification configuration to the vault.");
          SetVaultNotificationConfig();
          GetVaultNotificationConfig();
          Console.WriteLine("To delete vault notification configuration, press Enter");
          Console.ReadKey();
          DeleteVaultNotificationConfig();
        }
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static void SetVaultNotificationConfig()
    {

      SetVaultNotificationsRequest request = new SetVaultNotificationsRequest()
      {
        VaultName = vaultName,
        VaultNotificationConfig = new VaultNotificationConfig()
        {
          Events   = new List<string>() { "ArchiveRetrievalCompleted",
 "InventoryRetrievalCompleted" },
```

```
          SNSTopic = snsTopicARN
        }
      };
      SetVaultNotificationsResponse response = client.SetVaultNotifications(request);
    }

    static void GetVaultNotificationConfig()
    {
      GetVaultNotificationsRequest request = new GetVaultNotificationsRequest()
      {
        VaultName = vaultName,
        AccountId = "-"
      };
      GetVaultNotificationsResponse response = client.GetVaultNotifications(request);
      Console.WriteLine("SNS Topic ARN: {0}",
 response.VaultNotificationConfig.SNSTopic);
      foreach (string s in response.VaultNotificationConfig.Events)
        Console.WriteLine("Event : {0}", s);
    }

    static void DeleteVaultNotificationConfig()
    {
      DeleteVaultNotificationsRequest request = new DeleteVaultNotificationsRequest()
      {
        VaultName = vaultName
      };
      DeleteVaultNotificationsResponse response =
 client.DeleteVaultNotifications(request);
    }
  }
}
```

Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier mithilfe der REST-API

Informationen zum Konfigurieren von Tresorbenachrichtigungen mithilfe der REST-API finden Sie unter Set Vault Notification Configuration (PUT notification-configuration). Darüber hinaus können Sie auch Tresorbenachrichtigungen abrufen (Get Vault Notifications (GET notification-configuration)) und Tresorbenachrichtigungen löschen (Delete Vault Notifications (DELETE notification-configuration)).

Konfiguration von Vault-Benachrichtigungen mithilfe der Amazon Glacier-Konsole

In diesem Abschnitt wird beschrieben, wie Sie Tresorbenachrichtigungen mithilfe der Amazon Glacier-Konsole konfigurieren. Wenn Sie Benachrichtigungen konfigurieren, geben Sie Auftragsabschlussereignisse an, bei denen Benachrichtigungen an ein Amazon Simple Notification Service (Amazon SNS)-Thema gesendet werden. Neben der Benachrichtigungskonfiguration für den Tresor können Sie auch ein Thema angeben, in dem Benachrichtigungen veröffentlicht werden sollen, wenn Sie einen Auftrag initiieren. Wenn Ihr Tresor so konfiguriert ist, dass für ein bestimmtes Ereignis eine Benachrichtigung gesendet wird, und Sie in der Anforderung zur Auftragsinitiierung ebenfalls Benachrichtigungen konfigurieren, werden zwei Benachrichtigungen gesendet.

Konfigurieren einer Tresorbenachrichtigung

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie im linken Navigationsbereich Tresore aus.
3. Wählen Sie in der Liste Tresore einen Tresor aus.
4. Wählen Sie im Bereich Benachrichtigungen die Option Bearbeiten aus.
5. Wählen Sie auf der Seite Ereignis-Benachrichtigungen die Option Benachrichtigungen aktivieren aus.
6. Wählen Sie im Abschnitt Benachrichtigungen eine der folgenden Amazon Simple Notification Service (Amazon SNS)-Optionen aus und folgen Sie dann den entsprechenden Schritten:

Amazon-SNS-Optionen	Action
Erstellen eines neuen SNS-Themas	1. Wählen Sie Neues SNS-Thema erstellen aus. 2. Geben Sie unter Themenname den Namen eines neuen Themas ein. Der Name kann eine Länge von bis zu 256 Zeichen haben. Alphanumerische Zeichen, Bindestriche („-“) und Unterstriche („_“) sind erlaubt. Themennamen

Amazon-SNS-Optionen	Action
	müssen innerhalb des Kontos eindeutig sein und AWS-Region. 3. (Optional) Wenn Sie das Thema über SMS-Nachrichten abonnieren möchten, geben Sie einen Namen unter Anzeigename ein. Ein Anzeigename kann bis zu 100 Zeichen lang sein.

Amazon-SNS-Optionen	Action
Auswählen eines vorhandenen SNS-Themas	1. Wählen Sie Wählen Sie ein vorhandenes SNS-Thema aus. 2. Wählen Sie unter SNS-Thema angeben eine der folgenden Optionen aus: • Wählen Sie aus Ihren SNS-Themen Die Drop-down-Liste SNS-Thema wird angezeigt. Wählen Sie ein vorhandenes Thema aus der Drop-down-Liste aus. • SNS-Thema-ARN eingeben Das Textfeld Thema-ARN von Amazon SNS wird angezeigt. Geben Sie den Amazon-Ressourcennamen (ARN) für das SNS-Thema ein. Der ARN eines SNS-Themas hat das folgende Format: `arn:aws:sns:` *`region`*`:`*`account-id`*`:`*`topic-name`* Sie finden den ARN eines SNS-Themas in der Amazon-SNS-Konsole.

7. Wählen Sie unter Ereignisse eines oder beide Ereignisse aus, für die Sie Benachrichtigungen senden möchten:

   - Damit nur dann eine Benachrichtigung gesendet wird, wenn die Archivabrufaufträge abgeschlossen sind, wählen Sie Archivabrufauftrag abgeschlossen aus.
   - Wenn nur beim Abschluss von Tresorinventuraufträgen eine Benachrichtigung gesendet werden soll, wählen Sie Auftrag zum Abrufen des Tresorbestands abgeschlossen aus.

Konfiguration von Tresor-Benachrichtigungen mit dem AWS Command Line Interface

In diesem Abschnitt wird beschrieben, wie Sie Tresorbenachrichtigungen mithilfe der AWS Command Line Interface konfigurieren. Wenn Sie Benachrichtigungen konfigurieren, geben Sie Auftragsabschlussereignisse an, die Benachrichtigungen an ein Amazon Simple Notification Service (Amazon SNS)-Thema auslösen. Neben der Benachrichtigungskonfiguration für den Tresor können Sie auch ein Thema angeben, zu dem Benachrichtigungen veröffentlicht werden sollen, wenn Sie einen Auftrag initiieren. Wenn Ihr Tresor so konfiguriert ist, dass für ein bestimmtes Ereignis eine Benachrichtigung erfolgt und Sie in der Anfrage zur Auftragsinitiierung Benachrichtigungen angeben, werden zwei Benachrichtigungen gesendet.

Gehen Sie wie folgt vor, um die Tresorbenachrichtigung mit der AWS CLI zu konfigurieren.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Konfigurieren Sie Vault-Benachrichtigungen mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface

2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

```
aws glacier list-vaults --account-id 123456789012
```

- Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

```
aws configure list
```

Beispiel: Konfigurieren Sie Vault-Benachrichtigungen mit dem AWS CLI

1. Verwenden Sie den `set-vault-notifications`-Befehl, um Benachrichtigungen zu konfigurieren, die bei Eintritt bestimmter Ereignisse im Tresor gesendet werden. Standardmäßig erhalten Sie keine Benachrichtigungen.

```
aws glacier set-vault-notifications --vault-name examplevault --account-
id 111122223333 --vault-notification-config file://notificationconfig.json
```

2. Die Benachrichtigungskonfiguration ist ein JSON-Dokument, wie im folgenden Beispiel gezeigt.

```
{
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

   Weitere Informationen zur Verwendung von Amazon SNS SNS-Themen für Amazon Glacier finden Sie unter Konfiguration von Tresorbenachrichtigungen in Amazon Glacier: Allgemeine Konzepte

   Weitere Informationen zu Amazon SNS finden Sie unter Erste Schritte mit Amazon SNS.

Löschen eines Tresors in Amazon Glacier

Amazon Glacier (Amazon Glacier) löscht einen Tresor nur, wenn sich zum Zeitpunkt der letzten Inventarisierung keine Archive im Tresor befinden und seit der letzten Inventarisierung keine Schreibvorgänge in den Tresor stattgefunden haben. Informationen zum Löschen von Archiven finden Sie unter Löschen eines Archivs in Amazon Glacier. Informationen zum Herunterladen eines Tresorbestands finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

> **Note**
>
> Amazon Glacier bereitet regelmäßig eine Inventur für jeden Tresor vor, alle 24 Stunden. Da das Inventar möglicherweise nicht die neuesten Informationen enthält, stellt Amazon Glacier sicher, dass der Tresor tatsächlich leer ist, indem überprüft wird, ob seit der letzten Tresorinventur Schreibvorgänge stattgefunden haben.

> **Note**
>
> Informationen zum automatisierten Löschen von Tresorarchiven finden Sie unter Automatisiertes Löschen von Tresorarchiven in Amazon S3 Glacier.

Themen

- Löschen eines Tresors in Amazon Glacier mithilfe der AWS SDK für Java
- Löschen eines Tresors in Amazon Glacier mithilfe der AWS SDK für .NET
- Löschen eines Tresors in Amazon Glacier mithilfe der REST-API
- Löschen eines leeren Tresors mithilfe der Amazon Glacier-Konsole
- Löschen eines Tresors in Amazon Glacier mithilfe der AWS Command Line Interface

Löschen eines Tresors in Amazon Glacier mithilfe der AWS SDK für Java

Nachfolgend werden die Schritte zum Löschen eines Tresors mithilfe der Low-Level-API für AWS SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, aus der Sie einen Tresor löschen möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `DeleteVaultRequest`-Klasse.

   Sie müssen den Tresornamen und die Konto-ID angeben. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren

der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier.

3. Führen Sie die `deleteVault`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Amazon Glacier (Amazon Glacier) löscht den Tresor nur, wenn er leer ist. Weitere Informationen finden Sie unter Delete Vault (DELETE vault).

Der folgende Java-Codeausschnitt veranschaulicht die vorherigen Schritte.

```
try {
    DeleteVaultRequest request = new DeleteVaultRequest()
        .withVaultName("*** provide vault name ***");

    client.deleteVault(request);
    System.out.println("Deleted vault: " + vaultName);
} catch (Exception e) {
    System.err.println(e.getMessage());
}
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Delete Vault (DELETE vault).

Beispiel: Löschen eines Tresors mit dem AWS SDK für Java

Ein funktionierendes Codebeispiel finden Sie unter Beispiel: Erstellen eines Tresors mit dem AWS SDK für Java. Das Java-Codebeispiel zeigt grundlegende Tresoroperationen einschließlich dem Erstellen und Löschen von Tresoren.

Löschen eines Tresors in Amazon Glacier mithilfe der AWS SDK für .NET

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for .NET APIs bereitgestellt werden, bieten eine Methode zum Löschen eines Tresors.

Themen

- Löschen eines Tresors mithilfe der High-Level-API von AWS SDK für .NET
- Löschen eines Tresors mithilfe der Low-Level-API von AWS SDK für .NET

Löschen eines Tresors mithilfe der High-Level-API von AWS SDK für .NET

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `DeleteVault`-Methode, mit der Sie einen Tresor löschen können.

Beispiel: Löschen eines Tresors mithilfe der High-Level-API von AWS SDK für .NET

Ein funktionierendes Codebeispiel finden Sie unter Beispiel: Tresoroperationen unter Verwendung der High-Level-API von AWS SDK für .NET. Das C#-Codebeispiel zeigt grundlegende Tresoroperationen einschließlich dem Erstellen und Löschen von Tresoren.

Löschen eines Tresors mithilfe der Low-Level-API von AWS SDK für .NET

Im Folgenden werden die Schritte zum Löschen eines Tresors mithilfe des AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, aus der Sie einen Tresor löschen möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `DeleteVaultRequest`-Klasse.

   Sie müssen den Tresornamen und die Konto-ID angeben. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für .NET mit Amazon Glacier.
3. Führen Sie die `DeleteVault`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Amazon Glacier (Amazon Glacier) löscht den Tresor nur, wenn er leer ist. Weitere Informationen finden Sie unter Delete Vault (DELETE vault).

Der folgende C#-Codeausschnitt veranschaulicht die vorherigen Schritte. Das Snippet ruft Metadateninformationen eines Tresors ab, der in der Standardregion vorhanden ist. AWS

```
AmazonGlacier client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USEast1);

DeleteVaultRequest request = new DeleteVaultRequest()
{
  VaultName = "*** provide vault name ***"
};

DeleteVaultResponse response = client.DeleteVault(request);
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Delete Vault (DELETE vault).

Beispiel: Löschen eines Tresors mithilfe der Low-Level-API von AWS SDK für .NET

Ein funktionierendes Codebeispiel finden Sie unter Beispiel: Vault-Operationen unter Verwendung der Low-Level-API des AWS SDK für .NET. Das C#-Codebeispiel zeigt grundlegende Tresoroperationen einschließlich dem Erstellen und Löschen von Tresoren.

Löschen eines Tresors in Amazon Glacier mithilfe der REST-API

Informationen zum Löschen eines Tresors mithilfe der REST-API finden Sie unter Delete Vault (DELETE vault).

Löschen eines leeren Tresors mithilfe der Amazon Glacier-Konsole

> **Note**
>
> Vor dem Löschen eines Tresors müssen Sie alle vorhandenen Archive im Tresor löschen. Sie können dies tun, indem Sie Code schreiben, um eine Anfrage zum Löschen von Archiven zu stellen, indem Sie entweder die REST-API AWS SDK für Java, die AWS SDK für .NET, die oder verwenden, indem Sie AWS Command Line Interface (AWS CLI) verwenden. Informationen zum Löschen von Archiven finden Sie unter Schritt 5: Löschen eines Archivs aus einem Tresor in Amazon Glacier.

Wenn Ihr Tresor leer ist, können Sie ihn löschen, indem Sie die folgenden Schritte ausführen.

So löschen Sie einen leeren Tresor mithilfe der Amazon Glacier-Konsole

1. Melden Sie sich bei Amazon Glacier Console an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole.
2. Wählen Sie unter Region auswählen den Ort aus, AWS-Region an dem sich der Tresor befindet.
3. Wählen Sie im linken Navigationsbereich Tresore aus.
4. Wählen Sie in der Liste Tresore die Optionsschaltfläche neben dem Namen des Tresors aus, den Sie löschen möchten, und wählen Sie dann oben auf der Seite Löschen aus.
5. Bestätigen Sie im Dialogfeld Tresor löschen, dass Sie den Tresor löschen möchten, indem Sie Löschen auswählen.

   > **Important**
   >
   > Das Löschen eines Tresors kann nicht rückgängig gemacht werden.

6. Wenn Sie überprüfen möchten, ob Sie den Tresor gelöscht haben, öffnen Sie die Liste Tresore und geben den Namen des Tresors ein, den Sie gelöscht haben. Wenn der Tresor nicht gefunden wird, war der Löschvorgang erfolgreich.

Löschen eines Tresors in Amazon Glacier mithilfe der AWS Command Line Interface

Sie können leere und nicht leere Tresore in Amazon Glacier (Amazon Glacier) mithilfe von AWS Command Line Interface (AWS CLI) löschen.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Löschen eines leeren Tresors mit dem AWS CLI
- Beispiel: Löschen eines nicht leeren Tresors mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

Installation des AWS Command Line Interface

Konfiguration des AWS Command Line Interface

2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

     ```
     aws configure list
     ```

Beispiel: Löschen eines leeren Tresors mit dem AWS CLI

- Verwenden Sie den `delete-vault`-Befehl, um einen Tresor zu löschen, der keine Archive enthält.

  - ```
    aws glacier delete-vault --vault-name awsexamplevault --account-id 111122223333
    ```

Beispiel: Löschen eines nicht leeren Tresors mit dem AWS CLI

Amazon Glacier löscht einen Tresor nur, wenn sich zum Zeitpunkt der letzten Inventarisierung keine Archive im Tresor befinden und seit der letzten Inventarisierung keine Schreibvorgänge in den Tresor stattgefunden haben. Das Löschen eines nicht leeren Tresors erfolgt in drei Schritten: das Abrufen des Archivs IDs aus dem Inventarbericht eines Tresors, das Löschen der einzelnen Archive und das anschließende Löschen des Tresors.

1. Verwenden Sie den `initiate-job`-Befehl, um eine Lagerabrufaufgabe zu starten.
    ```

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters '{"Type": "inventory-retrieval"}'
```

Erwartete Ausgabe:

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. Verwenden Sie den `describe-job`-Befehl, um den Status der vorherigen Abrufaufgabe zu überprüfen.

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

Erwartete Ausgabe:

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie beim Initiieren des Jobs eine Benachrichtigungskonfiguration für den Tresor festgelegt oder ein Amazon Simple Notification Service (Amazon SNS) -Thema angegeben haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an das Thema.

Sie können Benachrichtigungskonfiguration für bestimmte Ereignisse auf dem Tresor einrichten. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier. Amazon Glacier sendet bei jedem Eintreten eines bestimmten Ereignisses eine Nachricht an das angegebene SNS-Thema.

4. Nach Abschluss laden Sie die Abrufaufgabe mit dem `get-job-output`-Befehl in die Datei „`output.json`“ herunter.

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output.json
```

Dieser Befehl erzeugt eine Datei mit den folgenden Feldern.

```
{
"VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
"InventoryDate":"*** job completion date ***",
"ArchiveList":[
{"ArchiveId":"*** archiveid ***",
"ArchiveDescription":*** archive description (if set) ***,
"CreationDate":"*** archive creation date ***",
"Size":"*** archive size (in bytes) ***",
"SHA256TreeHash":"*** archive hash ***"
}
{"ArchiveId":
...
]}
```

5. Verwenden Sie den `delete-archive`-Befehl, um jedes Archiv aus einem Tresor zu löschen, bis kein Archiv übrig bleibt.

```
aws glacier delete-archive --vault-name awsexamplevault --account-id 111122223333
 --archive-id "*** archiveid ***"
```

> **Note**
>
> Wenn Ihre Archiv-ID mit einem Bindestrich oder einem anderen Sonderzeichen beginnt, müssen Sie die Archiv-ID in Anführungszeichen setzen, um diesen Befehl auszuführen.

6. Verwenden Sie den `initiate-job`-Befehl, um eine neue Lagerabrufaufgabe zu starten.

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters '{"Type": "inventory-retrieval"}'
```

7. Nach Abschluss löschen Sie mit dem `delete-vault`-Befehl einen Tresor ohne Archive.

```
aws glacier delete-vault --vault-name awsexamplevault --account-id 111122223333
```

Kennzeichnen Ihrer Amazon Glacier-Tresore

Sie können Amazon Glacier-Tresors Ihre eigenen Metadaten in Form von Tags zuweisen. Ein Tag ist ein Schlüssel-Wert-Paar, das Sie für einen Tresor definieren. Grundlegende Informationen über das Markieren, einschließlich der Beschränkungen von Tags, finden Sie unter Taggen von Amazon Glacier-Ressourcen.

In den folgenden Themen wird beschrieben, wie Sie Tags zu Tresoren hinzufügen, auflisten und entfernen können.

Themen

- Kennzeichnen von Tresoren mithilfe der Amazon Glacier-Konsole
- Taggen von Tresoren mithilfe von AWS CLI
- Taggen von Tresoren mithilfe der Amazon Glacier-API
- Zugehörige Abschnitte

Kennzeichnen von Tresoren mithilfe der Amazon Glacier-Konsole

Sie können Tags mithilfe der Amazon Glacier-Konsole hinzufügen, auflisten und entfernen, wie in den folgenden Verfahren beschrieben.

So zeigen Sie die Tags für einen Tresor an

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie unter Region auswählen eine Region AWS-Region aus der Regionsauswahl aus.
3. Wählen Sie im linken Navigationsbereich Tresore aus.

4. Wählen Sie in der Liste Tresore einen Tresor aus.
5. Wählen Sie die Registerkarte Tresoreigenschaften aus. Scrollen Sie zum Abschnitt Tags, um die Tags anzuzeigen, die dem Tresor zugeordnet sind.

So fügen Sie einem Tresor ein Tag hinzu

Sie können einem Tresor bis zu 50 Tags zuordnen. Einem Tresor zugeordnete Tags müssen eindeutige Tag-Schlüssel aufweisen.

Weitere Informationen zu Tag-Einschränkungen finden Sie unter Tagging Amazon Glacier-Ressourcen.

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie unter Region auswählen eine Region AWS-Region aus der Regionsauswahl aus.
3. Wählen Sie im linken Navigationsbereich Tresore aus.
4. Wählen Sie in der Liste Tresore den Namen des Tresors aus, dem Tags hinzugefügt werden sollen.
5. Wählen Sie die Registerkarte Tresoreigenschaften aus.
6. Klicken Sie im Abschnitt Tags auf Add (Hinzufügen). Die Seite Add tags (Tags hinzufügen) wird angezeigt.
7. Geben Sie auf der Seite Tags hinzufügen den Tag-Schlüssel in das Feld Schlüssel. Optional können Sie einen Tag-Wert im Wert angeben.
8. Wählen Sie Änderungen speichern aus.

So bearbeiten Sie ein Tag

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie unter Region auswählen eine Region AWS-Region aus der Regionsauswahl aus.
3. Wählen Sie im linken Navigationsbereich Tresore aus.
4. Wählen Sie in der Liste Tresore einen Tresornamen aus.
5. Wählen Sie die Registerkarte Tresoreigenschaften aus und scrollen Sie dann nach unten zum Abschnitt Tags.

6. Aktivieren Sie unter Tags das Kontrollkästchen neben den Tags, die Sie ändern möchten, und wählen Sie dann Bearbeiten aus. Die Seite Tags bearbeiten wird angezeigt.
7. Aktualisieren Sie den Tag-Schlüssel im Feld Schlüssel und optional den Tag-Wert im Feld Wert.
8. Wählen Sie Änderungen speichern aus.

So entfernen Sie ein Tag aus einem Tresor

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie unter Region auswählen eine Region AWS-Region aus der Regionsauswahl aus.
3. Wählen Sie im linken Navigationsbereich Tresore aus.
4. Wählen Sie in der Liste Tresore den Namen des Tresors aus, dessen Tags Sie entfernen möchten.
5. Wählen Sie die Registerkarte Tresoreigenschaften aus. Scrollen Sie nach unten zum Abschnitt Tags.
6. Aktivieren Sie unter Tags das Kontrollkästchen neben den Tags, die Sie entfernen möchten, und wählen Sie dann Löschen aus.
7. Das Dialogfeld Tags löschen wird angezeigt. Zum Bestätigen, dass Sie die ausgewählten Tags löschen möchten, wählen Sie Löschen aus.

Taggen von Tresoren mithilfe von AWS CLI

Gehen Sie wie folgt vor, um Tags mithilfe von AWS Command Line Interface ()AWS CLI hinzuzufügen, aufzulisten oder zu entfernen.

Jedes Tag besteht aus einem Schlüssel und einem Wert. Jeder Tresor kann bis zu 50 Tags aufweisen.

1. Verwenden Sie den `add-tags-to-vault`-Befehl, um Tags zu einem Tresor hinzuzufügen.

```
aws glacier add-tags-to-vault --vault-name examplevault --account-id 111122223333
 --tags id=1234,date=2020
```

   Weitere Informationen zu diesem Tresorvorgang finden Sie unter Add Tags To Vault.

2. Verwenden Sie den `list-tags-for-vault`-Befehl, um alle Tags aufzulisten, die einem Tresor angefügt sind.

```
aws glacier list-tags-for-vault --vault-name examplevault --account-id 111122223333
```

   Weitere Informationen zu diesem Tresorvorgang finden Sie unter List Tags For Vault.

3. Wenn Sie einen oder mehrere Tags, die einem Tresor angefügt wurden, entfernen möchten, verwenden Sie den `remove-tags-from-vault`-Befehl.

```
aws glacier remove-tags-from-vault --vault-name examplevault --account-
id 111122223333 --tag-keys date
```

   Weitere Informationen zu diesem Tresorvorgang finden Sie unter Remove Tags From Vault.

Taggen von Tresoren mithilfe der Amazon Glacier-API

Mithilfe der Amazon Glacier-API können Sie Tags hinzufügen, auflisten und entfernen. Beispiele finden Sie in der folgenden Dokumentation:

Add Tags To Vault (POST tags add)

Fügt Tags für den angegebenen Tresor hinzu oder aktualisiert diese.

List Tags For Vault (GET tags)

Listet die Tags für den angegebenen Tresor auf.

Remove Tags From Vault (POST tags remove)

Entfernt Tags aus dem angegebenen Tresor.

Zugehörige Abschnitte

- Taggen von Amazon Glacier-Ressourcen

Amazon Glacier Tresorschloss

In den folgenden Themen wird beschrieben, wie Sie einen Tresor in Amazon Glacier sperren und wie Sie Vault Lock-Richtlinien verwenden.

Themen

- Übersicht über Tresorverriegelung
- Sperren eines Tresors mithilfe der Amazon Glacier-API
- Sperren eines Tresors mit dem AWS Command Line Interface
- Sperren eines Tresors mithilfe der Amazon Glacier-Konsole

Übersicht über Tresorverriegelung

Amazon Glacier Vault Lock unterstützt Sie bei der einfachen Implementierung und Durchsetzung von Compliance-Kontrollen für einzelne Amazon Glacier-Tresore mit einer Vault Lock-Richtlinie. Sie können in einer Tresorverriegelungs-Richtlinie beispielsweise eine WORM-Bedingung („write once, read many“) festlegen und die Richtlinie vor zukünftigen Änderungen schützen.

> **Important**
>
> Nachdem eine Tresorverriegelungs-Richtlinie gesperrt wurde, kann sie nicht mehr geändert oder gelöscht werden.

Amazon Glacier setzt die in der Vault Lock-Richtlinie festgelegten Kontrollen durch, um Ihre Compliance-Ziele zu erreichen. Mithilfe von Tresorverriegelungs-Richtlinien lässt sich z. B. die Datenaufbewahrung erzwingen. Sie können eine Vielzahl von Compliance-Kontrollen in einer Vault Lock-Richtlinie implementieren, indem Sie die Richtliniensprache AWS Identity and Access Management (IAM) verwenden. Weitere Informationen zu Tresorverriegelungs-Richtlinien finden Sie unter Tresorverriegelungs-Richtlinien.

Eine Tresorverriegelungs-Richtlinie unterscheidet sich von einer Tresorzugriffs-Richtlinie. Beide Richtlinien regeln Zugriffskontrollen zu Ihrem Tresor. Eine Tresorverriegelungs-Richtlinie kann jedoch gesperrt werden, um zukünftige Änderungen zu verhindern, und bietet damit eine konsequente Durchsetzung für Ihre Compliance-Kontrollen. Mit der Tresorverriegelungs-Richtlinie können Sie rechtliche und Compliance-Kontrollen bereitstellen, die in der Regel strenge Kontrollen des Datenzugriffs erfordern.

> **Important**
>
> Wir empfehlen Ihnen, zuerst einen Tresor zu erstellen, eine Tresorverriegelungs-Richtlinie einzurichten und dann Ihre Archive in den Tresor hochzuladen, damit die Richtlinie auf sie angewendet wird.

Im Gegensatz dazu implementieren Sie mithilfe von Tresorzugriffsrichtlinien Zugriffskontrollen, die nicht Compliance-abhängig sind, die vielmehr temporär sind und häufig geändert werden. Tresorverriegelungs- und Tresorzugriffsrichtlinien können zusammen eingesetzt werden. Beispielsweise können Sie zeitbasierte Datenaufbewahrungsregeln in der Tresorverriegelungs-Richtlinie implementieren (Löschen verweigern) und bestimmten Dritten oder Ihren Geschäftspartnern in der Tresorzugriffsrichtlinie Lesezugriff gewähren (Lesen erlauben).

Die Verriegelung eines Tresors erfolgt in zwei Schritten:

1. Initiieren Sie die Verriegelung, indem Sie Ihrem Tresor eine Tresorverriegelungs-Richtlinie anfügen, wodurch die Verriegelung in den Status „In Bearbeitung“ versetzt und eine Vault-Lock-ID zurückgegeben wird. Während sie im Status „In Bearbeitung“ ist, haben Sie 24 Stunden Zeit, Ihre Tresorverriegelungs-Richtlinie zu prüfen, bevor die Vault-Lock--ID abläuft. Um zu verhindern, dass Ihr Tresor den Status „In Bearbeitung“ verlässt, müssen Sie den Tresorverriegelungsvorgang innerhalb dieser 24 Stunden abschließen. Andernfalls wird Ihre Tresorverriegelungs-Richtlinie gelöscht.
2. Verwenden Sie die Vault-Lock-ID, um den Verriegelungsprozess abzuschließen. Wenn die Tresorverriegelungs-Richtlinie nicht wie erwartet funktioniert, können Sie den Verriegelungsvorgang abbrechen und noch einmal von Neuem beginnen. Informationen zur Verwendung der Amazon Glacier-API zum Sperren eines Tresors finden Sie unterSperren eines Tresors mithilfe der Amazon Glacier-API.

Sperren eines Tresors mithilfe der Amazon Glacier-API

Um Ihren Tresor mit der Amazon Glacier-API zu sperren, rufen Sie zunächst Initiate Vault Lock (POST lock-policy) mit einer Vault Lock-Richtlinie auf, die die Kontrollen festlegt, die Sie bereitstellen möchten. Der `Initiate Vault Lock`-Vorgang fügt die Richtlinie an den Tresor an, versetzt die Tresorverriegelung in den Status „In Bearbeitung“ und gibt eine eindeutige Vault-Lock-ID zurück. Nachdem die Tresorverriegelung in den Status „In Bearbeitung“ versetzt wurde, haben Sie

24 Stunden Zeit, um die Verriegelung abzuschließen, indem Sie Complete Vault Lock (POST lockId) mit der vom `Initiate Vault Lock`-Aufruf zurückgegebenen Vault-Lock-ID aufrufen.

> **Important**
>
> - Wir empfehlen Ihnen, zuerst einen Tresor zu erstellen, eine Tresorverriegelungs-Richtlinie einzurichten und dann Ihre Archive in den Tresor hochzuladen, damit die Richtlinie auf sie angewendet wird.
> - Nachdem die Tresorverriegelungs-Richtlinie gesperrt ist, kann sie nicht mehr geändert oder gelöscht werden.

Wenn Sie den Tresorverriegelungsprozess nicht innerhalb von 24 Stunden nach Eintritt in den Status „In Bearbeitung“ abschließen, verlässt der Tresor automatisch den Status „In Bearbeitung“ und die Tresorverriegelungs-Richtlinie wird entfernt. Sie können `Initiate Vault Lock` erneut aufrufen, um eine neue Tresorverriegelungs-Richtlinie zu installieren und in den Status „In Bearbeitung“ zu wechseln.

Der Status „In Bearbeitung“ bietet die Möglichkeit, Ihre Tresorverriegelungs-Richtlinie vor der Verriegelung zu testen. Ihre Tresorverriegelungs-Richtlinie erreicht während des Status „In Bearbeitung“ ihre volle Wirksamkeit, so als ob der Tresor verriegelt wäre, mit der Ausnahme, dass Sie die Richtlinie durch einen Aufruf von Abort Vault Lock (DELETE lock-policy) entfernen können. Sie können die Kombination `Abort Vault Lock`/`Initiate Vault Lock` so oft wie nötig wiederholen, um Ihre Richtlinie anzupassen und um die Änderungen an Ihrer Tresorverriegelungs-Richtlinie zu prüfen.

Nachdem Sie die Tresorverriegelungs-Richtlinie geprüft haben, können Sie Complete Vault Lock (POST lockId) mit der aktuellen Vault-Lock-ID aufrufen, um den Tresorverriegelungs-Prozess abzuschließen. Ihr Tresor wechselt in den verriegelten Status, in dem die Tresorverriegelungs-Richtlinie nicht mehr geändert werden und nicht mehr durch Aufruf von `Abort Vault Lock` entfernt werden kann.

Zugehörige Abschnitte

- Tresorverriegelungs-Richtlinien
- Abort Vault Lock (DELETE lock-policy)
- Complete Vault Lock (POST lockId)

- Get Vault Lock (GET lock-policy)
- Initiate Vault Lock (POST lock-policy)

Sperren eines Tresors mit dem AWS Command Line Interface

Sie können Ihren Tresor mit dem sperren AWS Command Line Interface. Dadurch wird eine Tresorverriegelungs-Richtlinie für den angegebenen Tresor installiert und die Vault-Lock-ID zurückgegeben. Sie müssen den Tresorverriegelungsvorgang innerhalb von 24 Stunden abschließen, andernfalls wird die Tresorverriegelungs-Richtlinie aus dem Tresor entfernt.

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface
2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

     ```
     aws configure list
     ```

1. Verwenden Sie den Befehl `initiate-vault-lock`, um eine Tresorverriegelungs-Richtlinie zu installieren und den Status der Tresorverriegelung auf `InProgress` festzulegen.

```
aws glacier initiate-vault-lock --vault-name examplevault --account-id 111122223333
 --policy file://lockconfig.json
```

2. Die Verriegelungskonfiguration ist ein JSON-Dokument, wie im folgenden Beispiel gezeigt. Bevor Sie diesen Befehl verwenden, ersetzen Sie das *VAULT_ARN* und *Principal* durch die entsprechenden Werte für Ihren Anwendungsfall.

   Um den ARN des Tresors zu finden, den Sie verriegeln möchten, können Sie den `list-vaults`-Befehl verwenden.

```
{"Policy":"{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Sid
\":\"Define-vault-lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":
\"arn:aws:iam::111122223333:root\"},\"Action\":\"glacier:DeleteArchive
\",\"Resource\":\"VAULT_ARN\",\"Condition\":{\"NumericLessThanEquals\":
{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}"}
```

3. Nachdem Sie die Tresorverriegelung initiiert haben, sollten Sie die zurückgegebene `lockId` sehen.

```
{
    "lockId": "LOCK_ID"
}
```

Zum Abschließen der Tresorverriegelung müssen Sie `complete-vault-lock` innerhalb von 24 Stunden ausführen, andernfalls wird die Tresorverriegelungs-Richtlinie aus dem Tresor entfernt.

```
aws glacier complete-vault-lock --vault-name examplevault --account-id 111122223333 --
lock-id LOCK_ID
```

Zugehörige Abschnitte

- initiate-vault-lock in der AWS CLI Befehlsreferenz
- list-vaults in der AWS CLI -Befehlsreferenz
- complete-vault-lock in der AWS CLI Befehlsreferenz
- Tresorverriegelungs-Richtlinien
- Abort Vault Lock (DELETE lock-policy)
- Complete Vault Lock (POST lockId)

- Get Vault Lock (GET lock-policy)
- Initiate Vault Lock (POST lock-policy)

Sperren eines Tresors mithilfe der Amazon Glacier-Konsole

Amazon Glacier Vault Lock unterstützt Sie bei der einfachen Implementierung und Durchsetzung von Compliance-Kontrollen für einzelne Amazon Glacier-Tresore mit einer Vault Lock-Richtlinie. Weitere Informationen zu Amazon Glacier Vault Lock finden Sie unter Amazon Glacier Access Control with Vault Lock-Richtlinien.

> **Important**
>
> - Wir empfehlen Ihnen, zuerst einen Tresor zu erstellen, eine Tresorverriegelungs-Richtlinie einzurichten und dann Ihre Archive in den Tresor hochzuladen, damit die Richtlinie auf sie angewendet wird.
> - Nachdem die Tresorverriegelungs-Richtlinie gesperrt ist, kann sie nicht mehr geändert oder gelöscht werden.

So initiieren Sie mithilfe der Amazon Glacier-Konsole eine Vault Lock-Richtlinie für Ihren Tresor

Initiieren Sie die Verriegelung, indem Sie Ihrem Tresor eine Tresorverriegelungs-Richtlinie anfügen, wodurch die Verriegelung in den Status „In Bearbeitung“ versetzt und eine Vault-Lock-ID zurückgegeben wird. Während sie im Status „In Bearbeitung“ ist, haben Sie 24 Stunden Zeit, Ihre Tresorverriegelungs-Richtlinie zu prüfen, bevor die Vault-Lock--ID abläuft.

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie unter Region auswählen eine Region AWS-Region aus der Regionsauswahl aus.
3. Wählen Sie im linken Navigationsbereich Tresore aus.
4. Wählen Sie auf der Seite Tresore die Option Tresor erstellen aus.
5. Erstellen Sie einen neuen Tresor.

> **Important**
>
> Wir empfehlen Ihnen, zuerst einen Tresor zu erstellen, eine Tresorverriegelungs-Richtlinie einzurichten und dann Ihre Archive in den Tresor hochzuladen, damit die Richtlinie auf sie angewendet wird.

6. Wählen Sie den neuen Tresor in der Liste Tresore aus.
7. Wählen Sie die Registerkarte Tresorrichtlinien aus.
8. Wählen Sie im Abschnitt Tresorverriegelungs-Richtlinie die Option Tresorverriegelungs-Richtlinie initiieren aus.
9. Geben Sie auf der Seite Tresorverriegelungs-Richtlinie initiieren im Standardtextfeld die in Ihrer Tresorverriegelungs-Richtlinie enthaltenen Kontrollen zur Aufbewahrung von Datensätzen im Textformat an.

> **Note**
>
> Sie können die Kontrollen zur Aufbewahrung von Datensätzen in einer Vault Lock-Richtlinie im Textformat angeben und die Vault Lock initiieren, indem Sie den `Initiate Vault Lock` API-Vorgang aufrufen oder die interaktive Benutzeroberfläche in der Amazon Glacier-Konsole verwenden. Informationen zur Formatierung Ihrer Vault Lock-Richtlinie finden Sie unter Beispiele für Amazon Glacier Vault Lock-Richtlinien.

10. Wählen Sie Änderungen speichern aus.
11. Kopieren Sie im Dialogfeld Vault-Lock-ID aufzeichnen Ihre Vault-Lock-ID und speichern Sie sie an einem sicheren Ort.

> **Important**
>
> Nachdem die Tresorverriegelungs-Richtlinie initiiert wurde, haben Sie 24 Stunden Zeit, um die Richtlinie zu prüfen und den Verriegelungsvorgang abzuschließen. Zum Abschließen des Verriegelungsvorgangs müssen Sie die Vault-Lock-ID angeben. Wenn diese nicht innerhalb von 24 Stunden bereitgestellt wird, läuft die Vault-Lock-ID ab und Ihre in Bearbeitung befindliche Richtlinie wird gelöscht.

12. Nachdem Sie Ihre Vault-Lock-ID an einem sicheren Ort gespeichert haben, wählen Sie Schließen aus.

13. Testen Sie Ihre Tresorverriegelungs-Richtlinie innerhalb der nächsten 24 Stunden. Wenn die Richtlinie wie vorgesehen funktioniert, wählen Sie Vault-Lock-Richtlinie vervollständigen aus.
14. Aktivieren Sie im Dialogfeld Vault Lock abschließen das Kontrollkästchen, um zu bestätigen, dass das Abschließen des Prozesses für die Tresorverriegelungs-Richtlinie nicht rückgängig gemacht werden kann.
15. Geben Sie Ihre angegebene Vault-Lock-ID in das Textfeld ein.
16. Wählen Sie Vault Lock abschließen aus.

Arbeiten mit Archiven in Amazon Glacier

Als Archiv wird jedes beliebige Objekt bezeichnet, wie z. B. ein Foto, Video oder Dokument, das Sie in einem Tresor speichern. Es ist eine Basisspeichereinheit in Amazon Glacier (Amazon Glacier). Jedes Archiv besitzt eine eindeutige ID und optional eine Beschreibung. Wenn Sie ein Archiv hochladen, gibt Amazon Glacier eine Antwort zurück, die eine Archiv-ID enthält. Diese Archiv-ID ist in der AWS Region, in der das Archiv gespeichert ist, eindeutig. Es folgt ein Beispiel für eine Archiv-ID.

```
TJgHcrOSfAkV6hdPqOATYfp_0ZaxL1pIBOc02iZ0gDPMr2ig-
nhwd_PafstsdIf6HSrjHnP-3p6LCJClYytFT_CBhT9CwNxbRaM5MetS3I-
GqwxI3Y8QtgbJbhEQPs0mJ3KExample
```

IDs Die Archive sind 138 Byte lang. Wenn Sie ein Archiv hochladen, können Sie optional eine Beschreibung angeben. Sie können ein Archiv mithilfe seiner ID, aber nicht mithilfe seiner Beschreibung abrufen.

> **Important**
>
> Amazon Glacier bietet eine Verwaltungskonsole. Sie können mit der Konsole Tresore erstellen und löschen. Für alle anderen Interaktionen mit Amazon Glacier müssen Sie jedoch die AWS Command Line Interface (CLI) verwenden oder Code schreiben. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über Amazon SDKs. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie unter AWS CLI Referenz für Amazon Glacier. Um das zu installieren AWS CLI, gehen Sie zu AWS Command Line Interface.

Themen

- Archivierungsvorgänge in Amazon Glacier
- Pflegen von clientseitigen Archivmetadaten
- Hochladen eines Archivs in Amazon Glacier
- Ein Archiv in Amazon Glacier herunterladen
- Löschen eines Archivs in Amazon Glacier

Archivierungsvorgänge in Amazon Glacier

Amazon Glacier unterstützt die folgenden grundlegenden Archivierungsvorgänge: Hochladen, Herunterladen und Löschen. Das Herunterladen eines Archivs ist ein asynchroner Vorgang.

Hochladen eines Archivs in Amazon Glacier

Sie können ein Archiv in einer einzigen Operation oder in Teilen hochladen. Der API-Aufruf, den Sie verwenden, um ein Archiv in Teilen hochzuladen, wird als „mehrteiliger Upload“ bezeichnet. Weitere Informationen finden Sie unter Hochladen eines Archivs in Amazon Glacier.

> **Important**
>
> Amazon Glacier bietet eine Verwaltungskonsole. Sie können mit der Konsole Tresore erstellen und löschen. Für alle anderen Interaktionen mit Amazon Glacier müssen Sie jedoch die AWS Command Line Interface (CLI) verwenden oder Code schreiben. Um beispielsweise Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über Amazon SDKs. Weitere Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie unter AWS CLI Referenz für Amazon Glacier. Um das zu installieren AWS CLI, gehen Sie zu AWS Command Line Interface.

Suchen einer Archiv-ID in Amazon Glacier

Sie können die Archiv-ID ermitteln, indem Sie den Tresorbestand für den Tresor, in dem sich Archiv befindet, herunterladen. Weitere Informationen zum Herunterladen des Tresorbestands finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

Ein Archiv in Amazon Glacier herunterladen

Das Herunterladen eines Archivs ist ein asynchroner Vorgang. Sie müssen zunächst einen Auftrag initiieren, um ein bestimmtes Archiv herunterzuladen. Nach Erhalt der Jobanfrage bereitet Amazon Glacier Ihr Archiv für den Download vor. Sie können Ihre Archivdaten herunterladen, nachdem der Auftrag abgeschlossen ist. Da der Job asynchron ist, können Sie Amazon Glacier bitten, nach Abschluss des Auftrags eine Benachrichtigung an ein Amazon Simple Notification Service (Amazon SNS) -Thema zu senden. Sie können ein SNS-Thema für jede einzelne Auftragsanfrage angeben

oder Ihren Tresor so konfigurieren, dass eine Benachrichtigung gesendet wird, wenn bestimmte Tresorereignisse eintreten. Weitere Informationen über das Herunterladen von Archiven finden Sie unter Ein Archiv in Amazon Glacier herunterladen.

Löschen eines Archivs in Amazon Glacier

Amazon Glacier bietet einen API-Aufruf, mit dem Sie jeweils ein Archiv löschen können. Weitere Informationen finden Sie unter Löschen eines Archivs in Amazon Glacier.

Aktualisieren eines Archivs in Amazon Glacier

Nachdem ein Archiv hochgeladen wurde, kann weder sein Inhalt noch seine Beschreibung aktualisiert werden. Sie können den Inhalt oder die Beschreibung des Archivs nur aktualisieren, indem Sie das Archiv löschen und ein anderes Archiv hochladen. Beachten Sie, dass Amazon Glacier Ihnen jedes Mal, wenn Sie ein Archiv hochladen, eine eindeutige Archiv-ID zurückgibt.

Pflegen von clientseitigen Archivmetadaten

Mit Ausnahme der optionalen Archivbeschreibung unterstützt Amazon Glacier keine zusätzlichen Metadaten für die Archive. Wenn Sie ein Archiv hochladen, weist Amazon Glacier eine ID zu, eine undurchsichtige Zeichenfolge, aus der Sie keine Bedeutung für das Archiv ableiten können. Es bietet sich darum an, clientseitig Metadaten für die Archive zu erstellen. Die Metadaten können den Archivnamen und weitere sinnvolle Informationen über das Archiv umfassen.

> **Note**
>
> Wenn Sie eine Kunde von Amazon Simple Storage Service (Amazon S3), dann wissen Sie, dass Sie einem Objekt, wenn Sie es in einen Bucket hochladen, einen Objektschlüssel wie z. B. `MyDocument.txt` oder `SomePhoto.jpg` zuweisen können. In Amazon Glacier können Sie den Archiven, die Sie hochladen, keinen Objektschlüssel zuweisen.

Wenn Sie clientseitige Archivmetadaten verwalten, beachten Sie, dass Amazon Glacier ein Tresorinventar verwaltet, das das Archiv IDs und alle Beschreibungen enthält, die Sie beim Archiv-Upload angegeben haben. Sie können den Tresorbestand gelegentlich herunterladen, um alle Informationen in Ihrer kundenseitigen Datenbank, die Sie für die Archivmetadaten pflegen, abzugleichen. Amazon Glacier führt jedoch ungefähr täglich ein Tresorinventar durch. Wenn Sie ein

Tresorinventar anfordern, gibt Amazon Glacier das letzte erstellte Inventar zurück, ein Point-in-Time-Snapshot.

Hochladen eines Archivs in Amazon Glacier

Amazon Glacier (Amazon Glacier) bietet eine Verwaltungskonsole, mit der Sie Tresore erstellen und löschen können. Sie können jedoch keine Archive mit der Management-Konsole auf Amazon Glacier hochladen. Um Daten wie Fotos, Videos und andere Dokumente hochzuladen, müssen Sie entweder den Code verwenden AWS CLI oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über Amazon SDKs.

Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie unter AWS CLI Referenz für Amazon Glacier. Um das zu installieren AWS CLI, gehen Sie zu AWS Command Line Interface. In den folgenden Themen zum Hochladen wird beschrieben, wie Archive mithilfe des Amazon SDK for Java, des Amazon SDK for .NET und der REST-API auf Amazon Glacier hochgeladen werden.

Themen

- Optionen für das Hochladen eines Archivs auf Amazon Glacier
- Hochladen eines Archivs in einer einzigen Operation
- Hochladen von großen Archiven in Teilen (Multipart Upload)

Optionen für das Hochladen eines Archivs auf Amazon Glacier

Abhängig von der Größe der Daten, die Sie hochladen, bietet Amazon Glacier die folgenden Optionen:

- Archive in einer einzelnen Operation hochladen: Sie können Archive mit einer Größe zwischen 1 Byte und 4 GB in einem Vorgang hochladen. Wir empfehlen Amazon Glacier-Kunden jedoch, den mehrteiligen Upload zu verwenden, um Archive mit mehr als 100 MB hochzuladen. Weitere Informationen finden Sie unter Hochladen eines Archivs in einer einzigen Operation.
- Archive in Teilen hochladen: Mit der API für den mehrteiligen Upload können Sie große Archive mit bis zu 40 000 GB (10 000 * 4 GB) hochladen.

  Die API-Funktion für den mehrteiligen Upload soll die Benutzererfahrung beim Hochladen größerer Archive verbessern. Sie können Archive in mehreren Teilen hochladen. Diese Teile können unabhängig voneinander in beliebiger Reihenfolge und parallel hochgeladen werden. Wenn der

Upload eines Teils fehlschlägt, müssen Sie nur diesen Teil und nicht das gesamte Archiv erneut hochladen. Sie können mehrteilige Uploads für Archive mit einer Größe von 1 Byte bis zu ca. 40.000 GB verwenden. Weitere Informationen finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

> **Important**
>
> Das Amazon Glacier-Tresor-Inventar wird nur einmal täglich aktualisiert. Wenn Sie ein Archiv hochladen, wird das dem Tresor neu hinzugefügte Archiv nicht sofort angezeigt (in der Konsole und in der heruntergeladenen Tresorbestandsliste), sondern erst nach der Aktualisierung des Tresorbestands.

Den AWS Snowball Edge Service nutzen

AWS Snowball Edge beschleunigt das Verschieben großer Datenmengen in und aus AWS Geräten von Amazon unter Umgehung des Internets. Weitere Informationen erhalten Sie auf der Detailseite von AWS Snowball Edge.

Um vorhandene Daten auf Amazon Glacier (Amazon Glacier) hochzuladen, könnten Sie erwägen, einen der AWS Snowball Edge Gerätetypen zum Importieren von Daten in Amazon S3 zu verwenden und sie dann zur Archivierung mithilfe von Lebenszyklusregeln in die Amazon Glacier-Speicherklasse zu verschieben. Wenn Sie Amazon S3-Objekte auf die Amazon Glacier-Speicherklasse umstellen, verwendet Amazon S3 intern Amazon Glacier für dauerhaften Speicher zu geringeren Kosten. Obwohl die Objekte in Amazon Glacier gespeichert sind, bleiben sie Amazon S3-Objekte, die Sie in Amazon S3 verwalten. Sie haben direkt über Amazon Glacier keinen Zugriff auf diese Objekte.

Weitere Informationen zur Amazon S3 S3-Lebenszykluskonfiguration und zur Übertragung von Objekten in die Amazon Glacier-Speicherklasse finden Sie unter Object Lifecycle Management and Transitioning Objects im Amazon Simple Storage Service-Benutzerhandbuch.

Hochladen eines Archivs in einer einzigen Operation

Wie in Hochladen eines Archivs in Amazon Glacier beschrieben, können Sie kleinere Archive in einer einzigen Operation hochladen. Wir empfehlen Kunden von Amazon Glacier (Amazon Glacier) jedoch, Multipart Upload zu verwenden, um Archive mit mehr als 100 MB hochzuladen.

Themen

- Hochladen eines Archivs in einem einzigen Vorgang mit dem AWS Command Line Interface
- Hochladen eines Archivs in einem einzigen Vorgang mit dem AWS SDK für Java
- Hochladen eines Archivs in einem einzigen Vorgang mit dem AWS SDK für .NET in Amazon Glacier
- Hochladen eines Archivs in einer einzigen Operation mit der REST-API

Hochladen eines Archivs in einem einzigen Vorgang mit dem AWS Command Line Interface

Sie können ein Archiv in Amazon Glacier (Amazon Glacier) hochladen, indem Sie AWS Command Line Interface (AWS CLI) verwenden.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Laden Sie ein Archiv hoch mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface

2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Eingabeaufforderung eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

- Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

```
aws configure list
```

Beispiel: Laden Sie ein Archiv hoch mit dem AWS CLI

Zum Hochladen eines Archivs müssen Sie einen Tresor erstellt haben. Weitere Informationen zum Erstellen von Tresoren finden Sie unter Einen Tresor in Amazon Glacier erstellen.

1. Verwenden Sie den `upload-archive`-Befehl, um einem vorhandenen Tresor ein Archiv hinzuzufügen. Ersetzen Sie im folgenden Beispiel `vault name` und `account ID`. Geben Sie für den `body`-Parameter einen Pfad zu der Datei an, die Sie hochladen möchten.

```
aws glacier upload-archive --vault-name awsexamplevault --account-id 123456789012
 --body archive.zip
```

2. Erwartete Ausgabe:

```
{
    "archiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
    "checksum": "969fb39823836d81f0cc028195fcdbcbbe76cdde932d4646fa7de5f21e18aa67",
    "location": "/123456789012/vaults/awsexamplevault/archives/
kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw"
}
```

   Wenn der Befehl fertig ist, gibt er die Archiv-ID, die Prüfsumme und den Speicherort in Amazon Glacier aus. Weitere Informationen zum Befehl „upload-archive“ finden Sie unter upload-archive in der AWS CLI -Befehlsreferenz.

Hochladen eines Archivs in einem einzigen Vorgang mit dem AWS SDK für Java

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for Java APIs bereitgestellt werden, bieten eine Methode zum Hochladen eines Archivs.

Themen

- Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java
- Hochladen eines Archivs in einem einzigen Vorgang mithilfe der Low-Level-API von AWS SDK für Java

Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `upload`-Methode, mit der Sie ein Archiv in einen Tresor hochladen können.

> **Note**
>
> Sie können die `upload`-Methode zum Hochladen von kleinen und großen Archiven verwenden. Die Größe des hochzuladenden Archivs entscheidet bei dieser Methode darüber, ob das Archiv in einem einzigen Vorgang hochgeladen wird oder ob die API für mehrteilige Uploads verwendet wird, um das Archiv in mehreren Teilen hochzuladen.

Beispiel: Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java

Mit dem folgenden Java-Beispielcode wird ein Archiv in einen Tresor (`examplevault`) in der Region „USA West (Oregon)“ (`us-west-2`) hochgeladen. Eine Liste der unterstützten AWS Regionen und Endpunkte finden Sie unter. Zugriff auf Amazon Glacier

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt aktualisieren und dabei den Namen des Tresors, in den Sie das Archiv hochladen möchten, und den Namen der hochzuladenden Datei angeben.

Example

```
import java.io.File;
import java.io.IOException;
import java.util.Date;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.transfer.ArchiveTransferManager;
import com.amazonaws.services.glacier.transfer.UploadResult;
```

```
public class ArchiveUploadHighLevel {
    public static String vaultName = "*** provide vault name ***";
    public static String archiveToUpload = "*** provide name of file to upload ***";

    public static AmazonGlacierClient client;

    public static void main(String[] args) throws IOException {


     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

        try {
            ArchiveTransferManager atm = new ArchiveTransferManager(client,
 credentials);

            UploadResult result = atm.upload(vaultName, "my archive " + (new Date()),
 new File(archiveToUpload));
            System.out.println("Archive ID: " + result.getArchiveId());

        } catch (Exception e)
        {
            System.err.println(e);
        }
    }
}
```

Hochladen eines Archivs in einem einzigen Vorgang mithilfe der Low-Level-API von AWS SDK für Java

Die Low-Level-API bietet Methoden für alle Archivvorgänge. Im Folgenden werden die Schritte zum Hochladen eines Archivs mit AWS SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in die Sie das Archiv hochladen möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `UploadArchiveRequest`-Klasse.

Zusätzlich zu den hochzuladenden Daten müssen Sie eine Prüfsumme (SHA-256-Struktur-Hash) für die Nutzlast, den Tresornamen, die Inhaltslänge der Daten und Ihre Konto-ID angeben.

Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier.

3. Führen Sie die `uploadArchive`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Als Antwort gibt Amazon Glacier (Amazon Glacier) eine Archiv-ID des neu hochgeladenen Archivs zurück.

Der folgende Java-Codeausschnitt veranschaulicht die vorherigen Schritte.

```
AmazonGlacierClient client;

UploadArchiveRequest request = new UploadArchiveRequest()
    .withVaultName("*** provide vault name ***")
    .withChecksum(checksum)
    .withBody(new ByteArrayInputStream(body))
    .withContentLength((long)body.length);

UploadArchiveResult uploadArchiveResult = client.uploadArchive(request);

System.out.println("Location (includes ArchiveID): " +
 uploadArchiveResult.getLocation());
```

Beispiel: Hochladen eines Archivs in einem einzigen Vorgang mithilfe der Low-Level-API von AWS SDK für Java

Das folgende Java-Codebeispiel verwendet den AWS SDK für Java , um ein Archiv in einen Tresor hochzuladen ()`examplevault`. step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt aktualisieren und dabei den Namen des Tresors, in den Sie das Archiv hochladen möchten, und den Namen der hochzuladenden Datei angeben.

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
```

```
import java.io.IOException;
import java.io.InputStream;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.TreeHashGenerator;
import com.amazonaws.services.glacier.model.UploadArchiveRequest;
import com.amazonaws.services.glacier.model.UploadArchiveResult;
public class ArchiveUploadLowLevel {

    public static String vaultName = "*** provide vault name ****";
    public static String archiveFilePath = "*** provide to file upload ****";
    public static AmazonGlacierClient client;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        try {
            // First open file and read.
            File file = new File(archiveFilePath);
            InputStream is = new FileInputStream(file);
            byte[] body = new byte[(int) file.length()];
            is.read(body);

            // Send request.
            UploadArchiveRequest request = new UploadArchiveRequest()
                .withVaultName(vaultName)
                .withChecksum(TreeHashGenerator.calculateTreeHash(new
 File(archiveFilePath)))
                .withBody(new ByteArrayInputStream(body))
                .withContentLength((long)body.length);

            UploadArchiveResult uploadArchiveResult = client.uploadArchive(request);

            System.out.println("ArchiveID: " + uploadArchiveResult.getArchiveId());

        } catch (Exception e)
        {
            System.err.println("Archive not uploaded.");
            System.err.println(e);
```

```
        }
    }
}
```

Hochladen eines Archivs in einem einzigen Vorgang mit dem AWS SDK für .NET in Amazon Glacier

Sowohl die vom Amazon SDK for .NET APIs bereitgestellte High-Level- als auch die Low-Level-Version bieten eine Methode, um ein Archiv in einem einzigen Vorgang hochzuladen.

Themen

- Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET
- Hochladen eines Archivs in einem einzigen Vorgang mithilfe der Low-Level-API von AWS SDK für .NET

Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `Upload`-Methode, mit der Sie ein Archiv in einen Tresor hochladen können.

> **Note**
>
> Sie können die `Upload`-Methode zum Hochladen von kleinen und großen Dateien verwenden. Die Größe der hochzuladenden Datei entscheidet bei dieser Methode darüber, ob die Datei in einer einziger Operationen hochgeladen wird oder ob die API für mehrteilige Uploads verwendet wird, um die Datei in mehreren Teilen hochzuladen.

Beispiel: Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

Mit dem folgenden C#-Beispielcode wird ein Archiv in einen Tresor (`examplevault`) in der Region „USA West (Oregon)“ hochgeladen.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit dem Namen der hochzuladenden Datei aktualisieren.

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveUploadHighLevel
  {
    static string vaultName = "examplevault";
    static string archiveToUpload = "*** Provide file name (with full path) to upload
 ***";

    public static void Main(string[] args)
    {
       try
      {
         var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
         // Upload an archive.
         string archiveId = manager.Upload(vaultName, "upload archive test",
 archiveToUpload).ArchiveId;
         Console.WriteLine("Archive ID: (Copy and save this ID for use in other
 examples.) : {0}", archiveId);
         Console.WriteLine("To continue, press Enter");
         Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

Hochladen eines Archivs in einem einzigen Vorgang mithilfe der Low-Level-API von AWS SDK für .NET

Die Low-Level-API bietet Methoden für alle Archivvorgänge. Im Folgenden werden die Schritte zum Hochladen eines Archivs mit AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in die Sie das Archiv hochladen möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `UploadArchiveRequest`-Klasse.

   Zusätzlich zu den hochzuladenden Daten müssen Sie eine Prüfsumme (SHA-256-Struktur-Hash) für die Nutzlast, den Tresornamen und Ihre Konto-ID angeben.

   Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für .NET mit Amazon Glacier.
3. Führen Sie die `UploadArchive`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

   Als Antwort gibt Amazon Glacier eine Archiv-ID des neu hochgeladenen Archivs zurück.

Beispiel: Hochladen eines Archivs in einem einzigen Vorgang mithilfe der Low-Level-API von AWS SDK für .NET

Im folgenden C#-Codebeispiel werden die vorstehenden Schritte veranschaulicht. Das Beispiel verwendet den AWS SDK für .NET , um ein Archiv in einen Tresor hochzuladen ()`examplevault`.

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API für das Hochladen eines Archivs in einer einzigen Anforderung finden Sie unter Upload Archive (POST archive).

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit dem Namen der hochzuladenden Datei aktualisieren.

Example

```
using System;
using System.IO;
using Amazon.Glacier;
```

```
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveUploadSingleOpLowLevel
  {
    static string vaultName       = "examplevault";
    static string archiveToUpload = "*** Provide file name (with full path) to upload
 ***";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
         using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Uploading an archive.");
          string archiveId = UploadAnArchive(client);
          Console.WriteLine("Archive ID: {0}", archiveId);
        }
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static string UploadAnArchive(AmazonGlacierClient client)
    {
      using (FileStream fileStream = new FileStream(archiveToUpload, FileMode.Open,
 FileAccess.Read))
      {
        string treeHash = TreeHashGenerator.CalculateTreeHash(fileStream);
        UploadArchiveRequest request = new UploadArchiveRequest()
        {
          VaultName = vaultName,
          Body = fileStream,
          Checksum = treeHash
        };
        UploadArchiveResponse response = client.UploadArchive(request);
        string archiveID = response.ArchiveId;
```

```
        return archiveID;
      }
    }
  }
}
```

Hochladen eines Archivs in einer einzigen Operation mit der REST-API

Sie können mit dem API-Aufruf "Upload Archive (Archiv hochladen)" ein Archiv in einer einzigen Operation hochladen. Weitere Informationen finden Sie unter Upload Archive (POST archive).

Hochladen von großen Archiven in Teilen (Multipart Upload)

Themen

- Mehrteiliger Upload
- Kurzinfo
- Hochladen großer Archive mit dem AWS CLI
- Hochladen von großen Archiven in Teilen mit dem Amazon-SDK für Java
- Hochladen großer Archive mit dem AWS SDK für .NET
- Hochladen von großen Archiven in Teilen mit der REST-API

Mehrteiliger Upload

Wie unter beschriebenHochladen eines Archivs in Amazon Glacier, empfehlen wir Kunden von Amazon Glacier (Amazon Glacier), Multipart Upload zu verwenden, um Archive mit mehr als 100 Mebibyte (MiB) hochzuladen.

1. Initiieren eines mehrteiligen Uploads

   Wenn Sie eine Anfrage zur Initiierung eines mehrteiligen Uploads senden, gibt Amazon Glacier eine mehrteilige Upload-ID zurück, die eine eindeutige Kennung für Ihren mehrteiligen Upload ist. Diese ID ist bei allen darauf folgenden mehrteiligen Upload-Vorgängen erforderlich. Diese ID läuft mindestens 24 Stunden nach Abschluss des Auftrags durch Amazon Glacier nicht ab.

   Geben Sie in der Anforderung, mit der Sie den mehrteiligen Upload starten, die Größe der einzelnen Teile in Bytes an. Mit Ausnahme des letzten Teils muss jeder der hochzuladenden Teile diese Größe aufweisen.

> **Note**
>
> Es ist nicht notwendig, dass Ihnen die Gesamtgröße des Archivs bekannt ist, wenn Sie mehrteilige Uploads verwenden. Das bedeutet, dass Sie mehrteilige Uploads verwenden können, wenn Sie die Archivgröße beim Hochladen des Archivs nicht kennen. Sie müssen lediglich beim Start des mehrteiligen Uploads die Größe der Teile festlegen.

Bei der Anforderung zur Initiierung des mehrteiligen Uploads können Sie optional auch eine Beschreibung des Archivs angeben.

2. Teile hochladen

   Bei jeder Anforderung zum Hochladen eines Teils müssen Sie die ID für den mehrteiligen Upload angeben, die Sie in Schritt 1 erhalten haben. Die Anforderung muss außerdem Informationen zum Inhaltsbereich (in Bytes) enthalten, durch die die Position des Teils im fertigen Archiv identifiziert wird. Amazon Glacier verwendet später die Inhaltsbereichsinformationen, um das Archiv in der richtigen Reihenfolge zusammenzustellen. Da Sie den Inhaltsbereich für jedes hochzuladende Teil angeben, ist dessen Position im abschließend zusammengesetzten Archiv genau gekennzeichnet. Darum können Sie die Teile in beliebiger Reihenfolge hochladen. Sie können Teile auch parallel hochladen. Wenn Sie einen neuen Teil mit denselben Informationen zum Inhaltsbereich hochladen wie für einen bereits hochgeladenen Teil, wird der zuvor hochgeladene Teil überschrieben.

3. Abschließen (oder Beenden) eines mehrteiligen Uploads

   Nachdem Sie alle Archivteile hochgeladen haben, schließen Sie den Vorgang ab. Auch hier müssen Sie die Upload-ID in Ihrer Anforderung angeben. Amazon Glacier erstellt ein Archiv, indem Teile auf der Grundlage des von Ihnen angegebenen Inhaltsbereichs in aufsteigender Reihenfolge verkettet werden. Die Antwort von Amazon Glacier auf eine Complete Multipart Upload-Anfrage enthält eine Archiv-ID für das neu erstellte Archiv. Wenn Sie in der Anfrage zum Initiate Multipart Upload eine optionale Archivbeschreibung angegeben haben, ordnet Amazon Glacier diese dem zusammengestellten Archiv zu. Wenn Sie einen mehrteiligen Upload erfolgreich abgeschlossen haben, können Sie nicht mehr auf die ID für den mehrteiligen Upload verweisen. Sie können also nicht auf Teile zugreifen, die mit der ID für den mehrteiligen Upload assoziiert sind.

Wenn Sie einen mehrteiligen Upload beenden, können Sie keine weiteren Teile mit der ID für diesen mehrteiligen Upload hochladen. Der Speicherplatz, der von den Teilen beansprucht wurde, die mit dem beendeten mehrteiligen Upload zusammenhängen, wird daraufhin freigegeben. Wenn der mehrteilige Upload beendet wird, während Teile hochgeladen werden, können diese Uploads auch nach dem Beenden erfolgreich abgeschlossen werden oder fehlschlagen.

Zusätzliche Operationen für den mehrteiligen Upload

Amazon Glacier (Amazon Glacier) bietet die folgenden zusätzlichen mehrteiligen Upload-API-Aufrufe.

- List Parts: Mit diesem Vorgang können Sie alle Teile eines bestimmten mehrteiligen Uploads auflisten. Dabei werden Informationen zu den Teilen zurückgegeben, die Sie bei einem mehrteiligen Upload hochgeladen haben. Für jede Anfrage zur Ersatzteilliste sendet Amazon Glacier Informationen für bis zu 1.000 Teile zurück. Wenn eine größere Anzahl von Teilen für den mehrteiligen Upload aufzuführen ist, werden die Ergebnisse in Seiten unterteilt und die Antwort enthält eine Markierung an der Stelle, an der die Liste fortgesetzt werden muss. Sie müssen weitere Anforderungen senden, um die nachfolgenden Teile abzurufen. Beachten Sie, dass die zurückgegebene Teileauflistung keine Teile enthält, die noch nicht vollständig hochgeladen wurden.
- List Multipart Uploads: Mit dieser Operation können Sie eine Liste aller mehrteiligen Uploads in Bearbeitung erhalten. Ein mehrteiliger Upload in Verarbeitung ist ein Upload, den Sie gestartet haben, der aber noch nicht abgeschlossen ist oder abgebrochen wurde. Auf jede Anforderung zum Auflisten der mehrteiligen Uploads gibt Amazon Glacier bis zu 1.000 mehrteilige Uploads zurück. Wenn eine größere Anzahl von mehrteiligen Uploads aufzuführen ist, werden die Ergebnisse in Seiten unterteilt und die Antwort enthält eine Markierung an der Stelle, an der die Liste fortgesetzt werden muss. Sie müssen weitere Anforderungen senden, um die verbleibenden mehrteiligen Uploads abzurufen.

Kurzinfo

Die folgende Tabelle enthält die Core-Spezifikationen für den mehrteiligen Upload.

Item	Spezifikation
Maximale Archivgröße	10 000 x 4 GiB (Gibibytes)
Maximale Anzahl von Teilen pro Upload	10.000
Teilegröße	1 MiB bis 4 GiB, der letzte Teil kann < 1 MiB sein. Sie geben die Größe in Bytes an. Die Größe der Teile muss ein Mebibyte (1 024 Kibibyte [KiB]) multipliziert mit einer Potenz von 2 sein. Zum Beispiel `1048576` (1 MiB), `2097152` (2 MiB), `4194304` (4 MiB), `8388608` (8 MiB).
Maximale Anzahl der zurückgegebenen Teile bei einer Anforderung zum Auflisten der Teile	1.000
Maximale Anzahl der zurückgegebenen mehrteiligen Uploads bei einer Anforderung zum Auflisten mehrteiliger Uploads	1.000

Hochladen großer Archive mit dem AWS CLI

Sie können ein Archiv in Amazon Glacier (Amazon Glacier) hochladen, indem Sie AWS Command Line Interface (AWS CLI) verwenden. Um das Upload-Erlebnis für größere Archive zu verbessern, bietet Amazon Glacier mehrere API-Operationen zur Unterstützung mehrteiliger Uploads. Mithilfe dieser API-Operationen können Sie Archive in Teilen hochladen. Diese Teile können unabhängig voneinander in beliebiger Reihenfolge und parallel hochgeladen werden. Wenn der Upload eines Teils fehlschlägt, müssen Sie nur diesen Teil und nicht das gesamte Archiv erneut hochladen. Sie können mehrteilige Uploads für Archive mit einer Größe von 1 Byte bis zu ca. 40 000 Gibibytes (GiB) verwenden.

Weitere Informationen zu mehrteiligen Amazon Glacier-Uploads finden Sie unter. Hochladen von großen Archiven in Teilen (Multipart Upload)

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- (Voraussetzung) Installieren von Python
- (Voraussetzung) Erstellen Sie einen Amazon Glacier Vault
- Beispiel: Teilweises Hochladen großer Archive mithilfe von AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface

2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

     ```
     aws configure list
     ```

(Voraussetzung) Installieren von Python

Um einen mehrteiligen Upload abzuschließen, müssen Sie den SHA256 Tree-Hash des Archivs berechnen, das Sie hochladen. Dies unterscheidet sich von der Berechnung des SHA256 Baumhashs der Datei, die Sie hochladen möchten. Um den SHA256 Baumhash des Archivs zu

berechnen, das Sie hochladen, können Sie C# (mit.NET) oder verwendenJava. Python In diesem Beispiel verwenden Sie Python. Anweisungen zur Verwendung von Java oder C# finden Sie unter Berechnen von Prüfsummen.

Informationen über das Installieren von Python finden Sie unter Installieren von Python im Boto3-Benutzerhandbuch.

(Voraussetzung) Erstellen Sie einen Amazon Glacier Vault

Um das folgende Beispiel verwenden zu können, muss mindestens ein Amazon Glacier-Tresor erstellt worden sein. Weitere Informationen zum Erstellen von Tresoren finden Sie unter Einen Tresor in Amazon Glacier erstellen.

Beispiel: Teilweises Hochladen großer Archive mithilfe von AWS CLI

In diesem Beispiel erstellen Sie eine Datei und verwenden API-Operationen für mehrteilige Uploads, um diese Datei in mehreren Teilen in Amazon Glacier hochzuladen.

> **Important**
>
> Bevor Sie mit diesem Verfahren beginnen, stellen Sie sicher, dass alle Voraussetzungen erfüllt sind. Um ein Archiv hochzuladen, müssen Sie einen Tresor erstellt und AWS CLI konfiguriert haben und bereit seinJava, C# zu verwenden oder einen Python SHA256 Tree-Hash zu berechnen.

Das folgende Verfahren verwendet die `complete-multipart-upload` AWS CLI Befehle `initiate-multipart-uploadupload-multipart-part`, und.

Ausführliche Informationen zu diesen Befehlen finden Sie unter initiate-multipart-upload, upload-multipart-part und complete-multipart-upload in der AWS CLI -Befehlsreferenz.

1. Verwenden Sie den initiate-multipart-upload-Befehl, um eine Ressource für den mehrteiligen Upload zu erstellen. Geben Sie in der Anforderung die Größe der einzelnen Teile in Bytes an. Mit Ausnahme des letzten Teils muss jeder der hochzuladenden Teile diese Größe aufweisen. Sie müssen beim Initiieren eines mehrteiligen Uploads die Gesamtgröße des Archivs nicht kennen. Sie benötigen jedoch die Gesamtgröße jedes Teils in Byte, wenn Sie den Upload im letzten Schritt abschließen.

Ersetzen Sie im folgenden Befehl die Werte für die Parameter `--vault-name` und `--account-ID` durch Ihre eigenen Informationen. Dieser Befehl gibt an, dass Sie ein Archiv mit einer Teilegröße von 1 Mebibyte (MiB) (1 024 x 1 024 Byte) pro Datei hochladen. Ersetzen Sie diesen `--part-size`-Parameterwert bei Bedarf.

```
aws glacier initiate-multipart-upload --vault-name awsexamplevault --part-
size 1048576 --account-id 123456789012
```

Erwartete Ausgabe:

```
{
"location": "/123456789012/vaults/awsexamplevault/multipart-uploads/uploadId",
"uploadId": "uploadId"
}
```

Wenn der Befehl fertig ist, gibt er die Upload-ID und den Speicherort der mehrteiligen Upload-Ressource in Amazon Glacier aus. Sie verwenden diese Upload-ID in den nachfolgenden Schritten.

2. In diesem Beispiel können Sie die folgenden Befehle verwenden, um eine Datei mit 4,4 MiB zu erstellen, sie in Blöcke von 1 MiB aufzuteilen und die einzelnen Blöcke hochzuladen. Um Ihre eigenen Dateien hochzuladen, können Sie die Daten mit diesem Verfahren in Blöcke aufteilen und die einzelnen Teile hochladen.

   Linux oder macOS

   Der folgende Befehl erstellt eine 4,4 MiB große Datei mit dem Namen `file_to_upload` unter Linux oder macOS.

   ```
   mkfile -n 9000b file_to_upload
   ```

   Windows

   Der folgende Befehl erstellt eine 4,4 MiB große Datei mit dem Namen `file_to_upload` unter Windows.

   ```
   fsutil file createnew file_to_upload 4608000
   ```

3. Als Nächstes teilen Sie diese Datei in Blöcke von 1 MiB auf.

```
split -b 1048576 file_to_upload chunk
```

Sie haben nun die folgenden fünf Blöcke. Die ersten vier sind 1 MiB groß, der letzte ist ungefähr 400 Kibibyte (KiB) groß.

```
chunkaa
chunkab
chunkac
chunkad
chunkae
```

4. Verwenden Sie den upload-multipart-part-Befehl, um einen Teil eines Archivs hochzuladen. Sie können die Archivteile in beliebiger Reihenfolge hochladen. Sie können die Teile auch parallel hochladen. Sie können bis zu 10 000 Teile für einen mehrteiligen Upload hochladen.

Ersetzen Sie im folgenden Befehl die Werte für die Parameter `--vault-name`, `--account-ID` und `--upload-id`. Die Upload-ID muss mit der ID übereinstimmen, die in der Ausgabe des `initiate-multipart-upload`-Befehls angegeben wurde. Der `--range`-Parameter gibt an, dass Sie einen Teil mit einer Größe von 1 MiB (1 024 x 1 024 Byte) hochladen. Diese Größe muss mit der Größe übereinstimmen, die Sie im `initiate-multipart-upload`-Befehl angegeben haben. Passen Sie diesen Größenwert bei Bedarf an. Der `--body`-Parameter gibt den Namen des Teils an, den Sie hochladen.

```
aws glacier upload-multipart-part --body chunkaa --range='bytes 0-1048575/*' --
vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

Bei erfolgreicher Ausführung erzeugt der Befehl eine Ausgabe, die die Prüfsumme für den hochgeladenen Teil enthält.

5. Führen Sie den `upload-multipart-part`-Befehl erneut aus, um die verbleibenden Teile Ihres mehrteiligen Uploads hochzuladen. Aktualisieren Sie die Parameterwerte für `--range` und `--body` in jeden Befehl so, dass sie mit dem Teil übereinstimmen, den Sie hochladen.

```
aws glacier upload-multipart-part --body chunkab --range='bytes 1048576-2097151/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

```
aws glacier upload-multipart-part --body chunkac --range='bytes 2097152-3145727/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

```
aws glacier upload-multipart-part --body chunkad --range='bytes 3145728-4194303/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

```
aws glacier upload-multipart-part --body chunkae --range='bytes 4194304-4607999/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

> **Note**
>
> Der `--range`-Parameterwert des letzten Befehls ist kleiner, da der letzte Teil des Uploads kleiner als 1 MiB ist. Bei erfolgreicher Ausführung erzeugt jeder Befehl eine Ausgabe, die die Prüfsumme für den hochgeladenen Teil enthält.

6. Als Nächstes stellen Sie das Archiv zusammen und schließen den Upload ab. Sie müssen die Gesamtgröße und den SHA256 Baumhash des Archivs angeben.

   Um den SHA256 Baumhash des Archivs zu berechnen, können Sie C# oder Python verwendenJava. In diesem Beispiel verwenden Sie Python. Anweisungen zur Verwendung von Java oder C# finden Sie unter Berechnen von Prüfsummen.

   Erstellen Sie die Python-Datei `checksum.py` und fügen Sie den folgenden Code ein. Falls erforderlich, ersetzen Sie den Namen der Originaldatei.

   ```
   from botocore.utils import calculate_tree_hash

   checksum = calculate_tree_hash(open('file_to_upload', 'rb'))
   print(checksum)
   ```

7. Führen Sie `checksum.py` den Befehl aus, um den SHA256 Baumhash zu berechnen. Der folgende Hash stimmt möglicherweise nicht mit Ihrer Ausgabe überein.

   ```
   $ python3 checksum.py
   $ 3d760edb291bfc9d90d35809243de092aea4c47b308290ad12d084f69988ae0c
   ```

8. Verwenden Sie den complete-multipart-upload-Befehl, um den Upload des Archivs abzuschließen. Ersetzen Sie die Werte für die Parameter `--vault-name`, `--account-ID`,

`--upload-ID` und `--checksum`. Der `--archive`-Parameterwert gibt die Gesamtgröße des Archivs in Byte an. Dieser Wert muss die Summe aller Größen der einzelnen Teile sein, die Sie hochgeladen haben. Ersetzen Sie diesen Wert bei Bedarf.

```
aws glacier complete-multipart-upload --archive-size 4608000 --vault-
name awsexamplevault --account-id 123456789012 --upload-id upload_ID --
checksum checksum
```

Wenn der Befehl fertig ist, gibt er die ID, die Prüfsumme und den Speicherort des Archivs in Amazon Glacier aus.

Hochladen von großen Archiven in Teilen mit dem Amazon-SDK für Java

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for Java APIs bereitgestellt werden, bieten eine Methode zum Hochladen eines großen Archivs (sieheHochladen eines Archivs in Amazon Glacier).

- Die High-Level-API bietet eine Methode, mit der Sie ein Archiv jeder Größe hochladen können. Abhängig von der Datei, die Sie hochladen, lädt die Methode entweder ein Archiv in einem einzigen Vorgang hoch oder verwendet die Unterstützung für mehrteiliges Hochladen in Amazon Glacier (Amazon Glacier), um das Archiv in Teilen hochzuladen.
- Die Low-Level-API wird eng auf die zugrundeliegende REST-Implementierung abgebildet. Dem entsprechend wird eine Methode zum Hochladen kleiner Archive in einer Operation bereitgestellt und eine Gruppen von Methoden, die den mehrteiligen Upload größerer Archive unterstützen. In diesem Abschnitt wird das Hochladen großer Archive in Teilen mithilfe der Low-Level-API beschrieben.

Weitere Informationen zur High-Level- und APIs Low-Level-Version finden Sie unter. Verwenden von AWS SDK für Java mit Amazon Glacier

Themen

- Teilweises Hochladen großer Archive mithilfe der High-Level-API von AWS SDK für Java
- Laden Sie große Archive in Teilen hoch, indem Sie die Low-Level-API von AWS SDK für Java

Teilweises Hochladen großer Archive mithilfe der High-Level-API von AWS SDK für Java

Sie verwenden die gleichen Methoden der High-Level-API für den Upload kleiner und großer Archive. Basierend auf der Archivgröße entscheiden die High-Level-API-Methoden, ob das Archiv in einem einzigen Vorgang hochgeladen oder die von Amazon Glacier bereitgestellte mehrteilige Upload-API verwendet wird. Weitere Informationen finden Sie unter Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java.

Laden Sie große Archive in Teilen hoch, indem Sie die Low-Level-API von AWS SDK für Java

Zur präzisen Steuerung des Uploads können Sie das Low-Level-API verwenden, über das Sie die Anfrage konfigurieren und die Antwort bearbeiten können. Im Folgenden werden die Schritte zum Hochladen großer Archive in Teilen mit dem AWS SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der Sie das Archiv speichern möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Initiieren eines mehrteiligen Uploads durch Aufrufen der Methode `initiateMultipartUpload`.

   Sie müssen den Namen des Tresors, in den Sie das Archiv hochladen möchten, die Größe der Teile, die Sie verwenden möchten, um Archivteile hochzuladen, und optional eine Beschreibung angeben. Diese Informationen stellen Sie bereit, indem Sie eine Instance der `InitiateMultipartUploadRequest`-Klasse erstellen. Als Antwort gibt Amazon Glacier eine Upload-ID zurück.
3. Laden Sie Teile durch den Aufruf der `uploadMultipartPart`-Methode hoch.

   Für jeden hochgeladenen Teil müssen folgende Angaben gemacht werden: der Tresorname und Byte-Bereich im endgültig zusammengestellten Archiv, das in diesem Teil hochgeladen wird, sowie die Prüfsumme der Teildaten und die Upload-ID.
4. Führen Sie den mehrteiligen Upload aus, indem Sie die Methode `completeMultipartUpload` aufrufen.

   Sie müssen die Upload-ID, die Prüfsumme des gesamten Archivs, die Archivgröße (kombinierte Größe aller hochgeladenen Teile) und den Tresornamen bereitstellen. Amazon Glacier erstellt das Archiv aus den hochgeladenen Teilen und gibt eine Archiv-ID zurück.

Beispiel: Hochladen eines großen Archivs in ein Bauteil mithilfe der AWS SDK für Java

Das folgende Java-Codebeispiel verwendet den AWS SDK für Java , um ein Archiv in einen Tresor hochzuladen (`examplevault`). step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt mit dem Namen der hochzuladenden Datei aktualisieren.

> **Note**
>
> Dieses Beispiel gilt für Teilgrößen von 1 MB bis 1 GB. Allerdings unterstützt Amazon Glacier Teilgrößen bis zu 4 GB.

Example

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Date;
import java.util.LinkedList;
import java.util.List;

import com.amazonaws.AmazonClientException;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.TreeHashGenerator;
import com.amazonaws.services.glacier.model.CompleteMultipartUploadRequest;
import com.amazonaws.services.glacier.model.CompleteMultipartUploadResult;
import com.amazonaws.services.glacier.model.InitiateMultipartUploadRequest;
import com.amazonaws.services.glacier.model.InitiateMultipartUploadResult;
import com.amazonaws.services.glacier.model.UploadMultipartPartRequest;
import com.amazonaws.services.glacier.model.UploadMultipartPartResult;
import com.amazonaws.util.BinaryUtils;

public class ArchiveMPU {

    public static String vaultName = "examplevault";
    // This example works for part sizes up to 1 GB.
```

```
    public static String partSize = "1048576"; // 1 MB.
    public static String archiveFilePath = "*** provide archive file path ***";
    public static AmazonGlacierClient client;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

        try {
            System.out.println("Uploading an archive.");
            String uploadId = initiateMultipartUpload();
            String checksum = uploadParts(uploadId);
            String archiveId = CompleteMultiPartUpload(uploadId, checksum);
            System.out.println("Completed an archive. ArchiveId: " + archiveId);

        } catch (Exception e) {
            System.err.println(e);
        }

    }

    private static String initiateMultipartUpload() {
        // Initiate
        InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest()
            .withVaultName(vaultName)
            .withArchiveDescription("my archive " + (new Date()))
            .withPartSize(partSize);

        InitiateMultipartUploadResult result = client.initiateMultipartUpload(request);

        System.out.println("ArchiveID: " + result.getUploadId());
        return result.getUploadId();
    }

    private static String uploadParts(String uploadId) throws AmazonServiceException,
NoSuchAlgorithmException, AmazonClientException, IOException {

        int filePosition = 0;
        long currentPosition = 0;
        byte[] buffer = new byte[Integer.valueOf(partSize)];
        List<byte[]> binaryChecksums = new LinkedList<byte[]>();
```

```
        File file = new File(archiveFilePath);
        FileInputStream fileToUpload = new FileInputStream(file);
        String contentRange;
        int read = 0;
        while (currentPosition < file.length())
        {
            read = fileToUpload.read(buffer, filePosition, buffer.length);
            if (read == -1) { break; }
            byte[] bytesRead = Arrays.copyOf(buffer, read);

            contentRange = String.format("bytes %s-%s/*", currentPosition,
currentPosition + read - 1);
            String checksum = TreeHashGenerator.calculateTreeHash(new
ByteArrayInputStream(bytesRead));
            byte[] binaryChecksum = BinaryUtils.fromHex(checksum);
            binaryChecksums.add(binaryChecksum);
            System.out.println(contentRange);

            //Upload part.
            UploadMultipartPartRequest partRequest = new UploadMultipartPartRequest()
            .withVaultName(vaultName)
            .withBody(new ByteArrayInputStream(bytesRead))
            .withChecksum(checksum)
            .withRange(contentRange)
            .withUploadId(uploadId);

            UploadMultipartPartResult partResult =
client.uploadMultipartPart(partRequest);
            System.out.println("Part uploaded, checksum: " + partResult.getChecksum());

            currentPosition = currentPosition + read;
        }
        fileToUpload.close();
        String checksum = TreeHashGenerator.calculateTreeHash(binaryChecksums);
        return checksum;
    }

    private static String CompleteMultiPartUpload(String uploadId, String checksum)
throws NoSuchAlgorithmException, IOException {

        File file = new File(archiveFilePath);
```

```
        CompleteMultipartUploadRequest compRequest = new
 CompleteMultipartUploadRequest()
            .withVaultName(vaultName)
            .withUploadId(uploadId)
            .withChecksum(checksum)
            .withArchiveSize(String.valueOf(file.length()));

        CompleteMultipartUploadResult compResult =
 client.completeMultipartUpload(compRequest);
        return compResult.getLocation();
    }
}
```

Hochladen großer Archive mit dem AWS SDK für .NET

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for .NET APIs bereitgestellt werden, bieten eine Methode, um große Archive in Teilen hochzuladen (siehe Hochladen eines Archivs in Amazon Glacier).

- Die High-Level-API bietet eine Methode, mit der Sie ein Archiv jeder Größe hochladen können. Abhängig von der Datei, die Sie hochladen, lädt die Methode entweder das Archiv in einem einzigen Vorgang hoch oder verwendet die Unterstützung für mehrteiliges Hochladen in Amazon Glacier (Amazon Glacier), um das Archiv in Teilen hochzuladen.
- Die Low-Level-API wird eng auf die zugrundeliegende REST-Implementierung abgebildet. Dem entsprechend wird eine Methode zum Hochladen kleiner Archive in einer Operation bereitgestellt und eine Gruppen von Methoden, die den mehrteiligen Upload größerer Archive unterstützen. In diesem Abschnitt wird das Hochladen großer Archive in Teilen mithilfe der Low-Level-API beschrieben.

Weitere Informationen zur High-Level- und APIs Low-Level-Version finden Sie unter. Verwenden von AWS SDK für .NET mit Amazon Glacier

Themen

- Teilweises Hochladen großer Archive mithilfe der High-Level-API von AWS SDK für .NET
- Teilweises Hochladen großer Archive mithilfe der Low-Level-API von AWS SDK für .NET

Teilweises Hochladen großer Archive mithilfe der High-Level-API von AWS SDK für .NET

Sie verwenden die gleichen Methoden der High-Level-API für den Upload kleiner und großer Archive. Basierend auf der Archivgröße entscheiden die High-Level-API-Methoden, ob das Archiv in einem einzigen Vorgang hochgeladen oder die von Amazon Glacier bereitgestellte mehrteilige Upload-API verwendet wird. Weitere Informationen finden Sie unter Hochladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET.

Teilweises Hochladen großer Archive mithilfe der Low-Level-API von AWS SDK für .NET

Zur präzisen Steuerung des Uploads können Sie das Low-Level-API verwenden, über das Sie die Anfrage konfigurieren und die Antwort bearbeiten können. Im Folgenden werden die Schritte zum Hochladen großer Archive in Teilen mit dem AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der Sie das Archiv speichern möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Initiieren eines mehrteiligen Uploads durch Aufrufen der Methode `InitiateMultipartUpload`.

   Sie müssen den Namen des Tresors, in den Sie das Archiv hochladen möchten, die Größe der Teile, die Sie verwenden möchten, um Archivteile hochzuladen, und optional eine Beschreibung angeben. Diese Informationen stellen Sie bereit, indem Sie eine Instance der `InitiateMultipartUploadRequest`-Klasse erstellen. Als Antwort gibt Amazon Glacier eine Upload-ID zurück.
3. Laden Sie Teile durch den Aufruf der `UploadMultipartPart`-Methode hoch.

   Für jeden hochgeladenen Teil müssen folgende Angaben gemacht werden: der Tresorname und Byte-Bereich im endgültig zusammengestellten Archiv, das in diesem Teil hochgeladen wird, sowie die Prüfsumme der Teildaten und die Upload-ID.
4. Führen Sie den mehrteiligen Upload aus, indem Sie die Methode `CompleteMultipartUpload` aufrufen.

   Sie müssen die Upload-ID, die Prüfsumme des gesamten Archivs, die Archivgröße (kombinierte Größe aller hochgeladenen Teile) und den Tresornamen bereitstellen. Amazon Glacier erstellt das Archiv aus den hochgeladenen Teilen und gibt eine Archiv-ID zurück.

Beispiel: Hochladen eines großen Archivs in Teilen mithilfe des Amazon-SDK für .NET

Das folgende C#-Codebeispiel verwendet den AWS SDK für .NET , um ein Archiv in einen Tresor hochzuladen ()`examplevault`. step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit dem Namen einer hochzuladenden Datei aktualisieren.

Example

```
using System;
using System.Collections.Generic;
using System.IO;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveUploadMPU
  {
    static string vaultName       = "examplevault";
    static string archiveToUpload = "*** Provide file name (with full path) to upload
 ***";
    static long partSize          = 4194304; // 4 MB.

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      List<string> partChecksumList = new List<string>();
      try
      {
         using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Uploading an archive.");
          string uploadId  = InitiateMultipartUpload(client);
          partChecksumList = UploadParts(uploadId, client);
          string archiveId = CompleteMPU(uploadId, client, partChecksumList);
          Console.WriteLine("Archive ID: {0}", archiveId);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
```

```
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static string InitiateMultipartUpload(AmazonGlacierClient client)
    {
      InitiateMultipartUploadRequest initiateMPUrequest = new
InitiateMultipartUploadRequest()
      {

        VaultName = vaultName,
        PartSize = partSize,
        ArchiveDescription = "Test doc uploaded using MPU."
      };

      InitiateMultipartUploadResponse initiateMPUresponse =
client.InitiateMultipartUpload(initiateMPUrequest);

      return initiateMPUresponse.UploadId;
    }

    static List<string> UploadParts(string uploadID, AmazonGlacierClient client)
    {
      List<string> partChecksumList = new List<string>();
      long currentPosition = 0;
      var buffer = new byte[Convert.ToInt32(partSize)];

      long fileLength = new FileInfo(archiveToUpload).Length;
      using (FileStream fileToUpload = new FileStream(archiveToUpload, FileMode.Open,
FileAccess.Read))
      {
        while (fileToUpload.Position < fileLength)
        {
          Stream uploadPartStream = GlacierUtils.CreatePartStream(fileToUpload,
partSize);
          string checksum = TreeHashGenerator.CalculateTreeHash(uploadPartStream);
          partChecksumList.Add(checksum);
          // Upload part.
          UploadMultipartPartRequest uploadMPUrequest = new
UploadMultipartPartRequest()
          {
```

```
            VaultName = vaultName,
            Body = uploadPartStream,
            Checksum = checksum,
            UploadId = uploadID
          };
          uploadMPUrequest.SetRange(currentPosition, currentPosition +
 uploadPartStream.Length - 1);
          client.UploadMultipartPart(uploadMPUrequest);

          currentPosition = currentPosition + uploadPartStream.Length;
        }
      }
      return partChecksumList;
    }

    static string CompleteMPU(string uploadID, AmazonGlacierClient client, List<string>
 partChecksumList)
    {
      long fileLength = new FileInfo(archiveToUpload).Length;
      CompleteMultipartUploadRequest completeMPUrequest = new
 CompleteMultipartUploadRequest()
      {
        UploadId = uploadID,
        ArchiveSize = fileLength.ToString(),
        Checksum = TreeHashGenerator.CalculateTreeHash(partChecksumList),
        VaultName = vaultName
      };

      CompleteMultipartUploadResponse completeMPUresponse =
 client.CompleteMultipartUpload(completeMPUrequest);
      return completeMPUresponse.ArchiveId;
    }
  }
}
```

Hochladen von großen Archiven in Teilen mit der REST-API

Wie unter Hochladen von großen Archiven in Teilen (Multipart Upload) beschrieben, bezieht sich der Begriff "mehrteiliger Upload" auf einen Satz von -Operationen, mit denen Sie ein Archiv in Teilen hochladen und damit zusammenhängende Operationen ausführen können. Weitere Informationen zu diesen Operationen finden Sie unter folgenden API-Referenzthemen:

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Part (PUT uploadID)
- Complete Multipart Upload (POST uploadID)
- Abort Multipart Upload (DELETE uploadID)
- List Parts (GET uploadID)
- List Multipart Uploads (GET multipart-uploads)

Ein Archiv in Amazon Glacier herunterladen

Amazon Glacier bietet eine Verwaltungskonsole, mit der Sie Tresore erstellen und löschen können. Sie können jedoch keine Archive von Amazon Glacier mithilfe der Managementkonsole herunterladen. Um Daten wie Fotos, Videos und andere Dokumente herunterzuladen, müssen Sie entweder die AWS Command Line Interface (AWS CLI) verwenden oder Code schreiben, um Anfragen zu stellen, entweder direkt über die REST-API oder über die AWS SDKs.

Informationen zur Verwendung von Amazon Glacier mit dem AWS CLI finden Sie in der AWS CLI Referenz für Amazon Glacier. Informationen zur Installation von AWS CLI finden Sie unter AWS Command Line Interface. In den folgenden Themen wird beschrieben, wie Archive mithilfe der AWS SDK für Java Amazon Glacier REST-API auf Amazon Glacier heruntergeladen werden. AWS SDK für .NET

Themen

- Amazon Glacier-Archive abrufen
- Herunterladen eines Archivs in Amazon Glacier mit dem AWS SDK für Java
- Herunterladen eines Archivs in Amazon Glacier mit dem AWS SDK für .NET
- Herunterladen eines großen Archivs mithilfe der Parallelverarbeitung mit Python
- Herunterladen eines Archivs mit der REST-API
- Herunterladen eines Archivs in Amazon Glacier mit dem AWS CLI

Amazon Glacier-Archive abrufen

Das Abrufen eines Archivs von Amazon Glacier ist ein asynchroner Vorgang, bei dem Sie zuerst einen Job initiieren und dann die Ausgabe herunterladen, nachdem der Job abgeschlossen ist. Um

einen Job zum Abrufen von Archiven zu initiieren, verwenden Sie den Initiate Job (POST jobs) REST-API-Vorgang oder einen gleichwertigen Vorgang in der AWS CLI, oder der. AWS SDKs

Themen

- Archiv-Abrufoptionen
- Abrufen von Archivbereichen

Das Abrufen eines Archivs von Amazon Glacier erfolgt in zwei Schritten. Im Folgenden finden Sie eine Übersicht über diesen Prozess.

So rufen Sie ein Archiv ab

1. Initiieren eines Auftrags zum Abrufen eines Archivs

   a. Ermitteln Sie die ID des Archivs, das Sie abrufen möchten. Sie finden die Archiv-ID in einer Inventarliste des Tresors. Sie können die Archiv-ID mit der REST-API, AWS CLI, oder abrufen AWS SDKs. Weitere Informationen finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

   b. Initiieren Sie mithilfe des Initiate Job (POST jobs) Vorgangs einen Job, der Amazon Glacier auffordert, ein ganzes Archiv oder einen Teil des Archivs für den späteren Download vorzubereiten.

   Wenn Sie einen Job initiieren, gibt Amazon Glacier in der Antwort eine Job-ID zurück und führt den Job asynchron aus. (Sie können die Auftragsausgabe erst herunterladen, nachdem der Auftrag wie in Schritt 2 beschrieben abgeschlossen wurde.)

   > **Important**
   >
   > Bei Standardabrufen kann eine Richtlinie für den Datenabruf dazu führen, dass die `Initiate Job`-Anforderung zu einer `PolicyEnforcedException`-Ausnahme führt und fehlschlägt. Weitere Informationen zu Datenabrufrichtlinien finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf. Weitere Informationen zur `PolicyEnforcedException`-Ausnahme finden Sie unter Fehlermeldungen.

   Bei Bedarf können Sie große Segmente der in Amazon Glacier gespeicherten Daten wiederherstellen. Weitere Informationen zum Wiederherstellen von Daten aus den Amazon

Glacier-Speicherklassen finden Sie unter Speicherklassen für die Archivierung von Objekten im Amazon Simple Storage Service-Benutzerhandbuch.

2. Laden Sie nach Abschluss des Auftrags die Bytes mit der Operation Get Job Output (GET output) herunter.

   Sie können alle Bytes herunterladen oder einen Bytebereich festlegen, sodass nur ein Teil der Auftragsausgabe heruntergeladen wird. Bei einer größeren Ausgabemenge bietet es sich an, die Ausgabe in Datenblöcken herunterzuladen, um Fehlern beim Herunterladen, wie z. B. durch einen Ausfall des Netzwerks, vorzubeugen. Wenn Sie eine einzige Anforderung zum Abrufen der gesamten Auftragsausgabe übermitteln, sind Sie bei Netzwerkausfällen gezwungen, den Vorgang zum Herunterladen der Ausgabe von Anfang an neu zu starten. Wenn Sie die Ausgabe jedoch in Datenblöcken herunterladen, müssen Sie bei einem Fehler nur den betroffenen Teil der Ausgabe und nicht die gesamten Ausgabedaten erneut herunterladen.

Amazon Glacier muss einen Auftrag abschließen, bevor Sie seine Ausgabe abrufen können. Nachdem ein Auftrag abgeschlossen wurde, bleibt er 24 Stunden lang verfügbar. Sie haben also 24 Stunden Zeit, um nach Beendigung des Auftrags die Ausgabe herunterzuladen. Die Wiederherstellung kann jederzeit innerhalb von 24 Stunden nach Abschluss des Auftrags ablaufen. Verwenden Sie eine der folgenden Optionen, um den Status des Auftrags zu überprüfen und festzustellen, ob der Auftrag abgeschlossen ist:

- Warten Sie auf eine Benachrichtigung über den Abschluss des Auftrags — Sie können ein Amazon Simple Notification Service (Amazon SNS) -Thema angeben, zu dem Amazon Glacier nach Abschluss des Auftrags eine Benachrichtigung senden kann. Amazon Glacier sendet erst dann eine Benachrichtigung, wenn der Auftrag abgeschlossen ist.

  Beim Initiieren eines Auftrags können Sie ein Amazon-SNS-Thema für den Auftrag angeben. Zusätzlich zur Angabe eines Amazon SNS SNS-Themas in Ihrer Jobanfrage veröffentlicht Amazon Glacier auch eine Benachrichtigung zu diesem SNS-Thema, wenn Ihr Tresor Benachrichtigungen für Archivabrufereignisse eingerichtet hat. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier.
- Jobinformationen explizit anfordern — Sie können auch die Amazon `Describe Job` Glacier-API-Operation (Describe Job (GET JobID)) verwenden, um regelmäßig Jobinformationen abzufragen. Wir empfehlen jedoch die Verwendung von Amazon-SNS-Benachrichtigungen.

> **Note**
>
> Die Informationen, die Sie in der Amazon-SNS-Benachrichtigung erhalten, sind identisch mit den Informationen, die Sie durch Aufrufen des `Describe Job`-API-Vorgangs erhalten.

Archiv-Abrufoptionen

Entsprechend den Anforderungen im Hinblick auf Zugriffszeit und Kosten können Sie beim Initiieren eines Auftrags zum Abrufen eines Archivs die folgenden Einstellungen festlegen. Informationen zu den Abrufpreisen finden Sie unter Amazon Glacier-Preise.

- Expedited – Hiermit können Sie schnell auf Ihre Daten zugreifen, die in der Speicherklasse „S3 Glacier Flexible Retrieval“ oder der S3-Intelligent-Tiering-Archivzugriffsebene gespeichert sind, wenn gelegentliche dringende Anforderungen für die Wiederherstellung von Archiven erforderlich sind. Daten, die unter Verwendung von „Expedited“ abgerufen werden, stehen normalerweise innerhalb von 1 bis 5 Minuten zur Verfügung, außer wenn es sich um sehr große Archive (über 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.
- Standard – Standardabrufe ermöglichen Ihnen den Zugriff auf alle Ihre Archive innerhalb einiger Stunden. Standardabrufe werden in der Regel innerhalb von 3 bis 5 Stunden ausgeführt. Standard ist die Standardoption für Abrufanforderungen, in denen keine Abrufoption angegeben ist.
- Massenabrufe — Massenabrufe sind die kostengünstigste Amazon Glacier-Abruf-Option, mit der Sie große Datenmengen, sogar Petabyte, kostengünstig an einem Tag abrufen können. Bulk-Abrufe werden in der Regel innerhalb von 5 bis 12 Stunden ausgeführt.

Die folgende Tabelle fasst die Archivabrufoptionen zusammen. Informationen zu Preisen finden Sie unter Amazon Glacier -Preise.

Um einen`Expedited`, oder `Bulk` -Abruf `Standard` durchzuführen, setzen Sie das `Tier` Anforderungselement in der RestoreObjectREST-API-Operationsanforderung auf die gewünschte Option oder die entsprechende Option in () oder. AWS Command Line Interface AWS CLI AWS SDKs Wenn Sie bereitgestellte Durchsatzkapazität gekauft haben, werden alle Expedited-Abrufe automatisch über Ihre bereitgestellte Kapazität erledigt.

Bereitgestellte Kapazität

Die bereitgestellte Kapazität hilft sicherzustellen, dass für Expedited-Abrufe Abrufkapazität verfügbar ist, wenn Sie sie benötigen. Jede Kapazitätseinheit sorgt dafür, dass mindestens drei beschleunigte Abrufe alle 5 Minuten ausgeführt werden können und bietet einen Abrufdurchsatz von bis zu 150 Megabyte pro Sekunde (). MBps

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 normalerweise 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.

Kaufen von bereitgestellter Durchsatzkapazität

Sie können bereitgestellte Kapazitätseinheiten erwerben, indem Sie die Amazon Glacier-Konsole, den Purchase Provisioned Capacity (POST provisioned-capacity) REST-API-Vorgang AWS SDKs, den oder den AWS CLI verwenden. Preisinformationen für bereitgestellte Kapazität finden Sie unter Amazon Glacier Pricing.

Eine bereitgestellte Kapazitätseinheit ist ab dem Datum und der Uhrzeit des Kaufs einen Monat lang gültig.

Wenn das Startdatum der 31. Tag eines Monats ist, dann ist der letzte Tag des nächsten Monats das Ablaufdatum. Beispiel: Ist das Startdatum der 31. August, dann ist der 30. September das Ablaufdatum. Ist das Startdatum der 31. Januar, dann ist der 28. Februar das Ablaufdatum.

So erwerben Sie bereitgestellte Kapazität mithilfe der Amazon Glacier-Konsole

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie im Navigationsbereich auf der linken Seite Datenabruf-Einstellungen aus.
3. Wählen Sie unter Bereitgestellte Kapazitätseinheiten (PCUs) die Option PCU kaufen aus. Das Dialogfeld PCU kaufen wird angezeigt.
4. Wenn Sie bereitgestellte Kapazität erwerben möchten, geben Sie **`confirm`** im Feld Kauf bestätigen ein.
5. Wählen Sie PCU kaufen aus.

Abrufen von Archivbereichen

Beim Abrufen eines Archivs von Amazon Glacier haben Sie die Möglichkeit, nur einen bestimmten Bereich bzw. einen Teil des Archivs abzurufen. In der Standardeinstellung wird das vollständige Archiv abgerufen. In den folgenden Fällen ist es von Vorteil einen bestimmten Bereich von Bytes anzugeben:

- Verwalten Sie Ihre Datendownloads — Amazon Glacier ermöglicht das Herunterladen von abgerufenen Daten für 24 Stunden, nachdem die Abrufanforderung abgeschlossen ist. Sie können also einzelne Bereiche des Archivs separat herunterladen, um den gesamten Download-Vorgang über den verfügbaren Zeitraum zu verteilen.
- Abrufen eines bestimmten Teils eines großen Archivs: Es kann beispielsweise vorkommen, dass Sie in der Vergangenheit viele Dateien aggregiert und zusammen in ein einziges Archiv hochgeladen haben, jetzt aber nur eine kleine Auswahl der Dateien abrufen möchten. In diesem Fall können Sie einen Bereich des Archivs angeben, der die gewünschten Dateien enthält, und eine Abrufanforderung übermitteln. Sie können allerdings auch mehrere Abrufanforderungen initiieren, die jeweils für einen Bereich mit einer oder mehreren Dateien gelten.

Beim Initiieren eines Abrufauftrags für bestimmte Bereiche müssen Sie sich bei der Angabe des Bereichs auf Megabytes beziehen. Das heißt, dass der Bytebereich bei null (am Anfang des Archivs) oder bei jedem nachfolgenden vollen Megabytewert beginnen kann (1 MB, 2 MB, 3 MB usw.).

Das Ende des Bereichs kann entweder das Ende Ihres Archivs oder ein beliebiges 1-MB-Intervall sein, das größer als der Anfang Ihres Bereichs ist. Wenn Sie beim Herunterladen der Daten Prüfsummenwerte erhalten möchten (nach Beendigung des Abrufauftrags), muss der Bereich, den Sie bei der Auftragsinitiierung anfordern, außerdem mit dem Struktur-Hash abgeglichen werden. Mithilfe von Prüfsummen können Sie sicherstellen, dass die Daten bei der Übertragung nicht beschädigt wurden. Weitere Informationen zum Megabyte- und Struktur-Hash-Abgleich finden Sie unter Erhalten von Prüfsummen bei Daten-Download.

Herunterladen eines Archivs in Amazon Glacier mit dem AWS SDK für Java

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for Java APIs bereitgestellt werden, bieten eine Methode zum Herunterladen eines Archivs.

Themen

- Herunterladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java
- Herunterladen eines Archivs mithilfe der Low-Level-API von AWS SDK für Java

Herunterladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `download`-Methode, mit der Sie ein Archiv herunterladen können.

> **Important**
>
> Die `ArchiveTransferManager`-Klasse erstellt ein Amazon Simple Notification Service (Amazon SNS)-Thema und eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange, die dieses Thema abonniert hat. Die Klasse initiiert daraufhin den Abrufauftrag für das Archiv und fragt die Warteschlange für das Archiv nach der Verfügbarkeit ab. Wenn das Archiv verfügbar ist, beginnt der Download. Weitere Informationen zu Abrufzeiten finden Sie unter Archiv-Abrufoptionen.

Beispiel: Herunterladen eines Archivs mithilfe der High-Level-API von AWS SDK für Java

Mit dem folgenden Java-Beispielcode wird ein Archiv aus einem Tresor („`examplevault`“) in der Region „USA West (Oregon)“ („`us-west-2`“) heruntergeladen.

step-by-stepAnweisungen zum Ausführen dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt mit einer vorhandenen Archiv-ID und dem lokalen Dateipfad zum Speicherort des heruntergeladenen Archivs aktualisieren.

Example

```
import java.io.File;
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.transfer.ArchiveTransferManager;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sqs.AmazonSQSClient;


public class ArchiveDownloadHighLevel {
    public static String vaultName = "examplevault";
    public static String archiveId = "*** provide archive ID ***";
    public static String downloadFilePath  = "*** provide location to download archive
 ***";
```

```
    public static AmazonGlacierClient glacierClient;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        glacierClient = new AmazonGlacierClient(credentials);

        sqsClient = new AmazonSQSClient(credentials);
        snsClient = new AmazonSNSClient(credentials);
        glacierClient.setEndpoint("glacier.us-west-2.amazonaws.com");
        sqsClient.setEndpoint("sqs.us-west-2.amazonaws.com");
        snsClient.setEndpoint("sns.us-west-2.amazonaws.com");

        try {
            ArchiveTransferManager atm = new ArchiveTransferManager(glacierClient,
 sqsClient, snsClient);

            atm.download(vaultName, archiveId, new File(downloadFilePath));
            System.out.println("Downloaded file to " + downloadFilePath);

        } catch (Exception e)
        {
            System.err.println(e);
        }
    }
}
```

Herunterladen eines Archivs mithilfe der Low-Level-API von AWS SDK für Java

Nachfolgend werden die Schritte zum Abrufen eines Tresorbestands mithilfe der Low-Level-API für AWS SDK für Java beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, aus der Sie das Archiv herunterladen möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Initiieren eines mehrteiligen `archive-retrieval`-Auftrags durch Ausführen der `initiateJob`-Methode.

Sie geben Jobinformationen an, z. B. die Archiv-ID des Archivs, das Sie herunterladen möchten, und das optionale Amazon SNS SNS-Thema, zu dem Amazon Glacier (Amazon Glacier) eine Nachricht zum Abschluss des Auftrags veröffentlichen soll, indem Sie eine Instanz der `InitiateJobRequest` Klasse erstellen. Amazon Glacier gibt als Antwort eine Job-ID zurück. Die Antwort ist in einer Instance der `InitiateJobResult`-Klasse verfügbar.

```
JobParameters jobParameters = new JobParameters()
    .withArchiveId("*** provide an archive id ***")
    .withDescription("archive retrieval")
    .withRetrievalByteRange("*** provide a retrieval range***") // optional
    .withType("archive-retrieval");

InitiateJobResult initiateJobResult = client.initiateJob(new InitiateJobRequest()
    .withJobParameters(jobParameters)
    .withVaultName(vaultName));

String jobId = initiateJobResult.getJobId();
```

Sie können optional einen Bytebereich angeben, um Amazon Glacier aufzufordern, nur einen Teil des Archivs vorzubereiten. Sie können beispielsweise die vorherige Anforderung aktualisieren, indem Sie die folgende Anweisung hinzufügen und von Amazon Glacier nur die Vorbereitung des 1 bis 2 MB großen Archivteils anfordern.

```
int ONE_MEG = 1048576;
String retrievalByteRange = String.format("%s-%s", ONE_MEG, 2*ONE_MEG -1);

JobParameters jobParameters = new JobParameters()
    .withType("archive-retrieval")
    .withArchiveId(archiveId)
    .withRetrievalByteRange(retrievalByteRange)
    .withSNSTopic(snsTopicARN);

InitiateJobResult initiateJobResult = client.initiateJob(new InitiateJobRequest()
    .withJobParameters(jobParameters)
    .withVaultName(vaultName));

String jobId = initiateJobResult.getJobId();
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie entweder im Tresor eine Benachrichtigungskonfiguration eingerichtet haben, die ein Amazon Simple Notification Service (Amazon SNS) -Thema identifiziert, oder ein Amazon SNS-Thema angegeben haben, als Sie einen Job initiiert haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an dieses Thema.

   Sie können Amazon Glacier auch abfragen, indem Sie die `describeJob` Methode aufrufen, um den Status der Auftragsabwicklung zu ermitteln. Allerdings wird die Verwendung eines Amazon-SNS-Themas für Benachrichtigungen empfohlen.

4. Laden Sie die Auftragsausgabe (Archivdaten) herunter, indem Sie die `getJobOutput`-Methode ausführen.

   Sie erstellen eine Instance der `GetJobOutputRequest`-Klasse, um die Informationen in der Anforderung, wie beispielsweise die Auftrags-ID und den Tresornamen, bereitzustellen. Die Ausgabe, die Amazon Glacier zurückgibt, ist im `GetJobOutputResult` Objekt verfügbar.

```
GetJobOutputRequest jobOutputRequest = new GetJobOutputRequest()
        .withJobId("*** provide a job ID ***")
        .withVaultName("*** provide a vault name ****");
GetJobOutputResult jobOutputResult = client.getJobOutput(jobOutputRequest);

// jobOutputResult.getBody() // Provides the input stream.
```

   Mit dem vorherigen Codeabschnitt wird die gesamte Auftragsausgabe heruntergeladen. Sie können optional nur einen Teil der Ausgabe abrufen oder die gesamte Ausgabe in kleineren Datenblöcken herunterladen, indem Sie den Bytebereich für `GetJobOutputRequest` angeben.

```
GetJobOutputRequest jobOutputRequest = new GetJobOutputRequest()
        .withJobId("*** provide a job ID ***")
        .withRange("bytes=0-1048575")   // Download only the first 1 MB of the
 output.
        .withVaultName("*** provide a vault name ****");
```

Als Antwort auf Ihren `GetJobOutput` Anruf gibt Amazon Glacier die Prüfsumme des Teils der Daten zurück, den Sie heruntergeladen haben, sofern bestimmte Bedingungen erfüllt sind. Weitere Informationen finden Sie unter Erhalten von Prüfsummen bei Daten-Download.

Um sicherzustellen, dass der Download fehlerfrei ist, können Sie dann die Prüfsumme auf der Clientseite berechnen und mit der Prüfsumme vergleichen, die Amazon Glacier in der Antwort gesendet hat.

Bei einem Job zum Abrufen von Archiven mit dem angegebenen optionalen Bereich enthält die Jobbeschreibung die Prüfsumme des Bereichs, den Sie abrufen (). SHA256 TreeHash Mit diesem Wert können Sie die Genauigkeit des gesamten später herunterzuladenden Byte-Bereichs weiter überprüfen. Sie können beispielsweise einen Auftrag initiieren, durch den zuerst ein mit dem Struktur-Hash abgeglichener Bereich abgerufen und dann die Ausgabe in Datenblöcken so heruntergeladen wird, dass für jede der `GetJobOutput`-Anforderungen eine Prüfsumme zurückgegeben wird. Anschließend können Sie die Prüfsumme für jeden auf der Clientseite heruntergeladenen Teil und schließlich das Struktur-Hash berechnen. Sie können sie mit der Prüfsumme vergleichen, die Amazon Glacier als Antwort auf Ihre Anfrage zur Beschreibung des Jobs zurückgibt, um sicherzustellen, dass der gesamte Bytebereich, den Sie heruntergeladen haben, mit dem Bytebereich übereinstimmt, der in Amazon Glacier gespeichert ist.

Ein funktionierendes Beispiel finden Sie unter Beispiel 2: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für Java—Output in Chunks herunterladen.

Beispiel 1: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für Java

Mit dem folgenden Java-Beispielcode wird ein Archiv aus dem angegebenen Tresor heruntergeladen. Nachdem der Auftrag abgeschlossen ist, lädt der Beispielcode die gesamte Ausgabe in einem einzigen `getJobOutput`-Aufruf herunter. Ein Beispiel für das Herunterladen der Ausgabe in Datenblöcken finden Sie unter Beispiel 2: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für Java—Output in Chunks herunterladen.

Das Beispiel führt die folgenden Aufgaben durch:

- Erstellt ein Amazon Simple Notification Service (Amazon SNS)-Thema.

  Amazon Glacier sendet nach Abschluss des Auftrags eine Benachrichtigung zu diesem Thema.
- Erstellt eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange.

Im Beispiel wird der Warteschlange eine Richtlinie angefügt, damit das Amazon-SNS-Thema Nachrichten in der Warteschlange veröffentlichen kann.

- Initiiert einen Auftrag zum Herunterladen des angegebenen Archivs.

  In der Jobanfrage ist das Amazon SNS SNS-Thema angegeben, das erstellt wurde, sodass Amazon Glacier nach Abschluss des Jobs eine Benachrichtigung zu dem Thema veröffentlichen kann.

- Prüft die Amazon-SQS-Warteschlange regelmäßig auf Nachrichten, die die Auftrags-ID enthalten.

  Ist eine Nachricht vorhanden, parsen Sie das JSON und prüfen Sie, ob der Auftrag erfolgreich abgeschlossen wurde. Wenn eine Nachricht enthalten ist, laden Sie das Archiv herunter.

- Bereinigt das System durch Löschen des Amazon-SNS-Themas und der erstellten Amazon-SQS-Warteschlange.

```
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.codehaus.jackson.JsonFactory;
import org.codehaus.jackson.JsonNode;
import org.codehaus.jackson.JsonParseException;
import org.codehaus.jackson.JsonParser;
import org.codehaus.jackson.map.ObjectMapper;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
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.SQSActions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.GetJobOutputRequest;
import com.amazonaws.services.glacier.model.GetJobOutputResult;
import com.amazonaws.services.glacier.model.InitiateJobRequest;
import com.amazonaws.services.glacier.model.InitiateJobResult;
import com.amazonaws.services.glacier.model.JobParameters;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sns.model.DeleteTopicRequest;
import com.amazonaws.services.sns.model.SubscribeRequest;
import com.amazonaws.services.sns.model.SubscribeResult;
import com.amazonaws.services.sns.model.UnsubscribeRequest;
import com.amazonaws.services.sqs.AmazonSQSClient;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.CreateQueueResult;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesResult;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;


public class AmazonGlacierDownloadArchiveWithSQSPolling {

    public static String archiveId = "*** provide archive ID ****";
    public static String vaultName = "*** provide vault name ***";
    public static String snsTopicName = "*** provide topic name ***";
    public static String sqsQueueName = "*** provide queue name ***";
    public static String sqsQueueARN;
    public static String sqsQueueURL;
    public static String snsTopicARN;
    public static String snsSubscriptionARN;
    public static String fileName = "*** provide file name ***";
    public static String region = "*** region ***";
    public static long sleepTime = 600;
    public static AmazonGlacierClient client;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;
```

```
    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier." + region + ".amazonaws.com");
        sqsClient = new AmazonSQSClient(credentials);
        sqsClient.setEndpoint("https://sqs." + region + ".amazonaws.com");
        snsClient = new AmazonSNSClient(credentials);
        snsClient.setEndpoint("https://sns." + region + ".amazonaws.com");

        try {
            setupSQS();

            setupSNS();

            String jobId = initiateJobRequest();
            System.out.println("Jobid = " + jobId);

            Boolean success = waitForJobToComplete(jobId, sqsQueueURL);
            if (!success) { throw new Exception("Job did not complete
successfully."); }

            downloadJobOutput(jobId);

            cleanUp();

        } catch (Exception e) {
            System.err.println("Archive retrieval failed.");
            System.err.println(e);
        }
    }

    private static void setupSQS() {
        CreateQueueRequest request = new CreateQueueRequest()
            .withQueueName(sqsQueueName);
        CreateQueueResult result = sqsClient.createQueue(request);
        sqsQueueURL = result.getQueueUrl();

        GetQueueAttributesRequest qRequest = new GetQueueAttributesRequest()
            .withQueueUrl(sqsQueueURL)
            .withAttributeNames("QueueArn");

        GetQueueAttributesResult qResult = sqsClient.getQueueAttributes(qRequest);
```

```
        sqsQueueARN = qResult.getAttributes().get("QueueArn");

        Policy sqsPolicy =
            new Policy().withStatements(
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueARN)));
        Map<String, String> queueAttributes = new HashMap<String, String>();
        queueAttributes.put("Policy", sqsPolicy.toJson());
        sqsClient.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueURL,
queueAttributes));

    }
    private static void setupSNS() {
        CreateTopicRequest request = new CreateTopicRequest()
            .withName(snsTopicName);
        CreateTopicResult result = snsClient.createTopic(request);
        snsTopicARN = result.getTopicArn();

        SubscribeRequest request2 = new SubscribeRequest()
            .withTopicArn(snsTopicARN)
            .withEndpoint(sqsQueueARN)
            .withProtocol("sqs");
        SubscribeResult result2 = snsClient.subscribe(request2);

        snsSubscriptionARN = result2.getSubscriptionArn();
    }
    private static String initiateJobRequest() {

        JobParameters jobParameters = new JobParameters()
            .withType("archive-retrieval")
            .withArchiveId(archiveId)
            .withSNSTopic(snsTopicARN);

        InitiateJobRequest request = new InitiateJobRequest()
            .withVaultName(vaultName)
            .withJobParameters(jobParameters);

        InitiateJobResult response = client.initiateJob(request);

        return response.getJobId();
    }
```

```
    private static Boolean waitForJobToComplete(String jobId, String sqsQueueUrl)
throws InterruptedException, JsonParseException, IOException {

        Boolean messageFound = false;
        Boolean jobSuccessful = false;
        ObjectMapper mapper = new ObjectMapper();
        JsonFactory factory = mapper.getJsonFactory();

        while (!messageFound) {
            List<Message> msgs = sqsClient.receiveMessage(
               new
ReceiveMessageRequest(sqsQueueUrl).withMaxNumberOfMessages(10)).getMessages();

            if (msgs.size() > 0) {
                for (Message m : msgs) {
                    JsonParser jpMessage = factory.createJsonParser(m.getBody());
                    JsonNode jobMessageNode = mapper.readTree(jpMessage);
                    String jobMessage = jobMessageNode.get("Message").getTextValue();

                    JsonParser jpDesc = factory.createJsonParser(jobMessage);
                    JsonNode jobDescNode = mapper.readTree(jpDesc);
                    String retrievedJobId = jobDescNode.get("JobId").getTextValue();
                    String statusCode = jobDescNode.get("StatusCode").getTextValue();
                    if (retrievedJobId.equals(jobId)) {
                        messageFound = true;
                        if (statusCode.equals("Succeeded")) {
                            jobSuccessful = true;
                        }
                    }
                }

            } else {
              Thread.sleep(sleepTime * 1000);
            }
          }
        return (messageFound && jobSuccessful);
    }

    private static void downloadJobOutput(String jobId) throws IOException {

        GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
            .withVaultName(vaultName)
            .withJobId(jobId);
```

```
        GetJobOutputResult getJobOutputResult =
 client.getJobOutput(getJobOutputRequest);

        InputStream input = new BufferedInputStream(getJobOutputResult.getBody());
        OutputStream output = null;
        try {
            output = new BufferedOutputStream(new FileOutputStream(fileName));

            byte[] buffer = new byte[1024 * 1024];

            int bytesRead = 0;
            do {
                bytesRead = input.read(buffer);
                if (bytesRead <= 0) break;
                output.write(buffer, 0, bytesRead);
            } while (bytesRead > 0);
        } catch (IOException e) {
            throw new AmazonClientException("Unable to save archive", e);
        } finally {
            try {input.close();}  catch (Exception e) {}
            try {output.close();} catch (Exception e) {}
        }
        System.out.println("Retrieved archive to " + fileName);
    }

    private static void cleanUp() {
        snsClient.unsubscribe(new UnsubscribeRequest(snsSubscriptionARN));
        snsClient.deleteTopic(new DeleteTopicRequest(snsTopicARN));
        sqsClient.deleteQueue(new DeleteQueueRequest(sqsQueueURL));
    }
}
```

Beispiel 2: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für Java—Output in Chunks herunterladen

Das folgende Java-Codebeispiel ruft ein Archiv von Amazon Glacier ab. Der Beispielcode gibt den abzurufenden Bytebereich in einem `GetJobOutputRequest`-Objekt an und lädt die Auftragsausgabe in Datenblöcken herunter.

```
import java.io.BufferedInputStream;
import java.io.ByteArrayInputStream;
import java.io.FileOutputStream;
```

```
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import com.fasterxml.jackson.core.JsonFactory;
import com.fasterxml.jackson.core.JsonParseException;
import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;

import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
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.SQSActions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.TreeHashGenerator;
import com.amazonaws.services.glacier.model.GetJobOutputRequest;
import com.amazonaws.services.glacier.model.GetJobOutputResult;
import com.amazonaws.services.glacier.model.InitiateJobRequest;
import com.amazonaws.services.glacier.model.InitiateJobResult;
import com.amazonaws.services.glacier.model.JobParameters;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sns.model.DeleteTopicRequest;
import com.amazonaws.services.sns.model.SubscribeRequest;
import com.amazonaws.services.sns.model.SubscribeResult;
import com.amazonaws.services.sns.model.UnsubscribeRequest;
import com.amazonaws.services.sqs.AmazonSQSClient;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.CreateQueueResult;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesResult;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;


public class ArchiveDownloadLowLevelWithRange {
```

```
    public static String vaultName = "*** provide vault name ***";
    public static String archiveId = "*** provide archive id ***";
    public static String snsTopicName = "glacier-temp-sns-topic";
    public static String sqsQueueName = "glacier-temp-sqs-queue";
    public static long downloadChunkSize = 4194304; // 4 MB
    public static String sqsQueueARN;
    public static String sqsQueueURL;
    public static String snsTopicARN;
    public static String snsSubscriptionARN;
    public static String fileName = "*** provide file name to save archive to ***";
    public static String region   = "*** region ***";
    public static long sleepTime  = 600;

    public static AmazonGlacierClient client;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier." + region + ".amazonaws.com");
        sqsClient = new AmazonSQSClient(credentials);
        sqsClient.setEndpoint("https://sqs." + region + ".amazonaws.com");
        snsClient = new AmazonSNSClient(credentials);
        snsClient.setEndpoint("https://sns." + region + ".amazonaws.com");

        try {
            setupSQS();

            setupSNS();

            String jobId = initiateJobRequest();
            System.out.println("Jobid = " + jobId);

            long archiveSizeInBytes = waitForJobToComplete(jobId, sqsQueueURL);
            if (archiveSizeInBytes==-1) { throw new Exception("Job did not complete
successfully."); }

            downloadJobOutput(jobId, archiveSizeInBytes);

            cleanUp();
```

```
        } catch (Exception e) {
            System.err.println("Archive retrieval failed.");
            System.err.println(e);
        }
    }

    private static void setupSQS() {
        CreateQueueRequest request = new CreateQueueRequest()
            .withQueueName(sqsQueueName);
        CreateQueueResult result = sqsClient.createQueue(request);
        sqsQueueURL = result.getQueueUrl();

        GetQueueAttributesRequest qRequest = new GetQueueAttributesRequest()
            .withQueueUrl(sqsQueueURL)
            .withAttributeNames("QueueArn");

        GetQueueAttributesResult qResult = sqsClient.getQueueAttributes(qRequest);
        sqsQueueARN = qResult.getAttributes().get("QueueArn");

        Policy sqsPolicy =
            new Policy().withStatements(
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueARN)));
        Map<String, String> queueAttributes = new HashMap<String, String>();
        queueAttributes.put("Policy", sqsPolicy.toJson());
        sqsClient.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueURL,
queueAttributes));

    }
    private static void setupSNS() {
        CreateTopicRequest request = new CreateTopicRequest()
            .withName(snsTopicName);
        CreateTopicResult result = snsClient.createTopic(request);
        snsTopicARN = result.getTopicArn();

        SubscribeRequest request2 = new SubscribeRequest()
            .withTopicArn(snsTopicARN)
            .withEndpoint(sqsQueueARN)
            .withProtocol("sqs");
        SubscribeResult result2 = snsClient.subscribe(request2);
```

```
        snsSubscriptionARN = result2.getSubscriptionArn();
    }
    private static String initiateJobRequest() {

        JobParameters jobParameters = new JobParameters()
            .withType("archive-retrieval")
            .withArchiveId(archiveId)
            .withSNSTopic(snsTopicARN);

        InitiateJobRequest request = new InitiateJobRequest()
            .withVaultName(vaultName)
            .withJobParameters(jobParameters);

        InitiateJobResult response = client.initiateJob(request);

        return response.getJobId();
    }

    private static long waitForJobToComplete(String jobId, String sqsQueueUrl) throws
InterruptedException, JsonParseException, IOException {

        Boolean messageFound = false;
        Boolean jobSuccessful = false;
        long archiveSizeInBytes = -1;
        ObjectMapper mapper = new ObjectMapper();
        JsonFactory factory = mapper.getFactory();

        while (!messageFound) {
            List<Message> msgs = sqsClient.receiveMessage(
               new
ReceiveMessageRequest(sqsQueueUrl).withMaxNumberOfMessages(10)).getMessages();

            if (msgs.size() > 0) {
                for (Message m : msgs) {
                    JsonParser jpMessage = factory.createJsonParser(m.getBody());
                    JsonNode jobMessageNode = mapper.readTree(jpMessage);
                    String jobMessage = jobMessageNode.get("Message").textValue();

                    JsonParser jpDesc = factory.createJsonParser(jobMessage);
                    JsonNode jobDescNode = mapper.readTree(jpDesc);
                    String retrievedJobId = jobDescNode.get("JobId").textValue();
                    String statusCode = jobDescNode.get("StatusCode").textValue();
                    archiveSizeInBytes =
jobDescNode.get("ArchiveSizeInBytes").longValue();
```

```
                    if (retrievedJobId.equals(jobId)) {
                        messageFound = true;
                        if (statusCode.equals("Succeeded")) {
                            jobSuccessful = true;
                        }
                    }
                }

            } else {
              Thread.sleep(sleepTime * 1000);
            }
          }
        return (messageFound && jobSuccessful) ? archiveSizeInBytes : -1;
    }

    private static void downloadJobOutput(String jobId, long archiveSizeInBytes) throws
IOException {

        if (archiveSizeInBytes < 0) {
            System.err.println("Nothing to download.");
            return;
        }

        System.out.println("archiveSizeInBytes: " + archiveSizeInBytes);
        FileOutputStream fstream = new FileOutputStream(fileName);
        long startRange = 0;
        long endRange = (downloadChunkSize > archiveSizeInBytes) ? archiveSizeInBytes
-1 : downloadChunkSize - 1;

        do {

            GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
                .withVaultName(vaultName)
                .withRange("bytes=" + startRange + "-" + endRange)
                .withJobId(jobId);
            GetJobOutputResult getJobOutputResult =
client.getJobOutput(getJobOutputRequest);

            BufferedInputStream is = new
BufferedInputStream(getJobOutputResult.getBody());
            byte[] buffer = new byte[(int)(endRange - startRange + 1)];

            System.out.println("Checksum received: " +
getJobOutputResult.getChecksum());
```

```
            System.out.println("Content range " +
 getJobOutputResult.getContentRange());


            int totalRead = 0;
            while (totalRead < buffer.length) {
                int bytesRemaining = buffer.length - totalRead;
                int read = is.read(buffer, totalRead, bytesRemaining);
                if (read > 0) {
                    totalRead = totalRead + read;
                } else {
                    break;
                }

            }
            System.out.println("Calculated checksum: " +
 TreeHashGenerator.calculateTreeHash(new ByteArrayInputStream(buffer)));
            System.out.println("read = " + totalRead);
            fstream.write(buffer);

            startRange = startRange + (long)totalRead;
            endRange = ((endRange + downloadChunkSize) >  archiveSizeInBytes) ?
 archiveSizeInBytes : (endRange + downloadChunkSize);
            is.close();
        } while (endRange <= archiveSizeInBytes  && startRange < archiveSizeInBytes);

        fstream.close();
        System.out.println("Retrieved file to " + fileName);

    }

    private static void cleanUp() {
        snsClient.unsubscribe(new UnsubscribeRequest(snsSubscriptionARN));
        snsClient.deleteTopic(new DeleteTopicRequest(snsTopicARN));
        sqsClient.deleteQueue(new DeleteQueueRequest(sqsQueueURL));
    }
}
```

Herunterladen eines Archivs in Amazon Glacier mit dem AWS SDK für .NET

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for .NET APIs bereitgestellt werden, bieten eine Methode zum Herunterladen eines Archivs.

Themen

- Herunterladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET
- Herunterladen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET

Herunterladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `Download`-Methode, mit der Sie ein Archiv herunterladen können.

> **Important**
>
> Die `ArchiveTransferManager`-Klasse erstellt ein Amazon Simple Notification Service (Amazon SNS)-Thema und eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange, die dieses Thema abonniert hat. Die Klasse initiiert daraufhin den Abrufauftrag für das Archiv und fragt die Warteschlange für das Archiv nach der Verfügbarkeit ab. Wenn das Archiv verfügbar ist, beginnt der Download. Weitere Informationen zu Abrufzeiten finden Sie unter Archiv-Abrufoptionen.

Beispiel: Herunterladen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

Mit dem folgenden C#-Beispielcode wird ein Archiv aus einem Tresor („`examplevault`“) in der Region „USA West (Oregon)“ heruntergeladen.

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit einer vorhandenen Archiv-ID und dem lokalen Dateipfad zum Speicherort des heruntergeladenen Archivs aktualisieren.

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDownloadHighLevel
  {
    static string vaultName        = "examplevault";
    static string archiveId        = "*** Provide archive ID ***";
```

```
    static string downloadFilePath = "*** Provide the file name and path to where to
 store the download ***";

    public static void Main(string[] args)
    {
      try
      {
        var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);

        var options = new DownloadOptions();
        options.StreamTransferProgress += ArchiveDownloadHighLevel.progress;
        // Download an archive.
        Console.WriteLine("Intiating the archive retrieval job and then polling SQS
 queue for the archive to be available.");
        Console.WriteLine("Once the archive is available, downloading will begin.");
        manager.Download(vaultName, archiveId, downloadFilePath, options);
        Console.WriteLine("To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static int currentPercentage = -1;
    static void progress(object sender, StreamTransferProgressArgs args)
    {
      if (args.PercentDone != currentPercentage)
      {
        currentPercentage = args.PercentDone;
        Console.WriteLine("Downloaded {0}%", args.PercentDone);
      }
    }
  }
}
```

Herunterladen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET

Im Folgenden finden Sie die Schritte zum Herunterladen eines Amazon Glacier-Archivs (Amazon Glacier) mithilfe der Low-Level-API von. AWS SDK für .NET

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, aus der Sie das Archiv herunterladen möchten. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.

2. Initiieren eines mehrteiligen `archive-retrieval`-Auftrags durch Ausführen der `InitiateJob`-Methode.

   Sie geben Jobinformationen an, z. B. die Archiv-ID des Archivs, das Sie herunterladen möchten, und das optionale Amazon SNS SNS-Thema, zu dem Amazon Glacier eine Nachricht über den Abschluss des Auftrags senden soll, indem Sie eine Instanz der `InitiateJobRequest` Klasse erstellen. Amazon Glacier gibt als Antwort eine Job-ID zurück. Die Antwort ist in einer Instance der `InitiateJobResponse`-Klasse verfügbar.

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);

InitiateJobRequest initJobRequest = new InitiateJobRequest()
{
  VaultName = vaultName,
  JobParameters = new JobParameters()
  {
    Type      = "archive-retrieval",
    ArchiveId = "*** Provide archive id ***",
    SNSTopic  = "*** Provide Amazon SNS topic ARN ***",
  }
};

InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
string jobId = initJobResponse.JobId;
```

   Sie können optional einen Bytebereich angeben, um Amazon Glacier aufzufordern, nur einen Teil des Archivs vorzubereiten, wie in der folgenden Anfrage gezeigt. In der Anfrage wird angegeben, dass Amazon Glacier nur den Teil des Archivs mit einer Größe von 1 MB bis 2 MB vorbereitet.

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);


InitiateJobRequest initJobRequest = new InitiateJobRequest()
{
  VaultName = vaultName,
```

```
  JobParameters = new JobParameters()
  {
    Type      = "archive-retrieval",
    ArchiveId = "*** Provide archive id ***",
    SNSTopic  = "*** Provide Amazon SNS topic ARN ***",
  }
};
// Specify byte range.
int ONE_MEG = 1048576;
initJobRequest.JobParameters.RetrievalByteRange = string.Format("{0}-{1}", ONE_MEG, 2
 * ONE_MEG -1);

InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
string jobId = initJobResponse.JobId;
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie entweder im Tresor eine Benachrichtigungskonfiguration eingerichtet haben, die ein Amazon Simple Notification Service (Amazon SNS) -Thema identifiziert, oder ein Amazon SNS-Thema angegeben haben, als Sie einen Job initiiert haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an dieses Thema. Das Codebeispiel im folgenden Abschnitt verwendet Amazon SNS for Amazon Glacier, um eine Nachricht zu veröffentlichen.

   Sie können Amazon Glacier auch abfragen, indem Sie die `DescribeJob` Methode aufrufen, um den Status der Auftragsabwicklung zu ermitteln. Allerdings wird die Verwendung eines Amazon-SNS-Themas für Benachrichtigungen empfohlen.

4. Laden Sie die Auftragsausgabe (Archivdaten) herunter, indem Sie die `GetJobOutput`-Methode ausführen.

   Sie erstellen eine Instance der `GetJobOutputRequest`-Klasse, um die Informationen in der Anforderung, wie beispielsweise die Auftrags-ID und den Tresornamen, bereitzustellen. Die Ausgabe, die Amazon Glacier zurückgibt, ist im `GetJobOutputResponse` Objekt verfügbar.

```
GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
{
  JobId = jobId,
  VaultName = vaultName
};

GetJobOutputResponse getJobOutputResponse = client.GetJobOutput(getJobOutputRequest);
```

```
using (Stream webStream = getJobOutputResponse.Body)
{
  using (Stream fileToSave = File.OpenWrite(fileName))
  {
     CopyStream(webStream, fileToSave);
  }
}
```

Mit dem vorherigen Codeabschnitt wird die gesamte Auftragsausgabe heruntergeladen. Sie können optional nur einen Teil der Ausgabe abrufen oder die gesamte Ausgabe in kleineren Datenblöcken herunterladen, indem Sie den Bytebereich für `GetJobOutputRequest` angeben.

```
GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
{
  JobId = jobId,
  VaultName = vaultName
};
getJobOutputRequest.SetRange(0, 1048575); // Download only the first 1 MB chunk of
 the output.
```

Als Antwort auf Ihren `GetJobOutput` Anruf gibt Amazon Glacier die Prüfsumme des Teils der Daten zurück, den Sie heruntergeladen haben, sofern bestimmte Bedingungen erfüllt sind. Weitere Informationen finden Sie unter Erhalten von Prüfsummen bei Daten-Download.

Um sicherzustellen, dass der Download fehlerfrei ist, können Sie dann die Prüfsumme auf der Clientseite berechnen und sie mit der Prüfsumme vergleichen, die Amazon Glacier in der Antwort gesendet hat.

Bei einem Job zum Abrufen von Archiven mit dem angegebenen optionalen Bereich enthält die Jobbeschreibung beim Abrufen der Jobbeschreibung die Prüfsumme des Bereichs, den Sie abrufen (SHA256TreeHash). Sie können diesen Wert verwenden, um die Richtigkeit des gesamten Bytebereichs, den Sie später herunterladen, weiter zu überprüfen. Sie können beispielsweise einen Auftrag initiieren, durch den zuerst ein mit dem Struktur-Hash abgeglichener Bereich abgerufen und dann die Ausgabe in Datenblöcken so heruntergeladen wird, dass für jede der `GetJobOutput`-Anforderungen eine Prüfsumme zurückgegeben wird. Anschließend können Sie die Prüfsumme für jeden auf der Clientseite heruntergeladenen Teil und schließlich das Struktur-Hash berechnen. Sie können sie mit der Prüfsumme vergleichen, die Amazon Glacier als Antwort auf Ihre Anfrage zur Beschreibung des Jobs zurückgibt, um sicherzustellen, dass der gesamte

Bytebereich, den Sie heruntergeladen haben, mit dem Bytebereich übereinstimmt, der in Amazon Glacier gespeichert ist.

Ein funktionierendes Beispiel finden Sie unter Beispiel 2: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET—Download Output in Chunks.

Beispiel 1: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET

Mit dem folgenden C#-Beispielcode wird ein Archiv aus dem angegebenen Tresor heruntergeladen. Nachdem der Auftrag abgeschlossen ist, lädt der Beispielcode die gesamte Ausgabe in einem einzigen `GetJobOutput`-Aufruf herunter. Ein Beispiel für das Herunterladen der Ausgabe in Datenblöcken finden Sie unter Beispiel 2: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET—Download Output in Chunks.

Das Beispiel führt die folgenden Aufgaben durch:

- Richtet ein Amazon Simple Notification Service (Amazon SNS)-Thema ein.

  Amazon Glacier sendet nach Abschluss des Auftrags eine Benachrichtigung zu diesem Thema.
- Richtet eine Amazon Simple Queue Service (Amazon SQS)-Warteschlange ein.

  Im Beispiel wird der Warteschlange eine Richtlinie angefügt, damit das Amazon-SNS-Thema Nachrichten veröffentlichen kann.
- Initiiert einen Auftrag zum Herunterladen des angegebenen Archivs.

  In der Jobanfrage gibt das Beispiel das Amazon SNS SNS-Thema an, sodass Amazon Glacier nach Abschluss des Auftrags eine Nachricht senden kann.
- Überprüft regelmäßig die Amazon-SQS-Warteschlange auf Nachrichten.

  Ist eine Nachricht vorhanden, parsen Sie das JSON und prüfen Sie, ob der Auftrag erfolgreich abgeschlossen wurde. Wenn eine Nachricht enthalten ist, laden Sie das Archiv herunter. Im Codebeispiel wird die JSON.NET-Bibliothek (siehe JSON.NET) zum Analysieren von JSON verwendet.
- Bereinigt das System durch Löschen des Amazon-SNS-Themas und der erstellten Amazon-SQS-Warteschlange.

```
using System;
```

```
using System.Collections.Generic;
using System.IO;
using System.Threading;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
using Amazon.SQS;
using Amazon.SQS.Model;
using Newtonsoft.Json;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDownloadLowLevelUsingSNSSQS
  {
    static string topicArn;
    static string queueUrl;
    static string queueArn;
    static string vaultName = "*** Provide vault name ***";
    static string archiveID = "*** Provide archive ID ***";
    static string fileName  = "*** Provide the file name and path to where to store
 downloaded archive ***";
    static AmazonSimpleNotificationServiceClient snsClient;
    static AmazonSQSClient sqsClient;
    const string SQS_POLICY =
        "{" +
        "    \"Version\" : \"2012-10-17\",&TCX5-2025-waiver;" +
        "    \"Statement\" : [" +
        "        {" +
        "            \"Sid\" : \"sns-rule\"," +
        "            \"Effect\" : \"Allow\"," +
        "            \"Principal\" : {\"Service\" : \"sns.amazonaws.com\" }," +
        "            \"Action\"    : \"sqs:SendMessage\"," +
        "            \"Resource\"  : \"{QueueArn}\"," +
        "            \"Condition\" : {" +
        "                \"ArnLike\" : {" +
        "                    \"aws:SourceArn\" : \"{TopicArn}\"" +
        "                }" +
        "            }" +
        "        }" +
        "    ]" +
        "}";
```

```
    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Setup SNS topic and SQS queue.");
          SetupTopicAndQueue();
          Console.WriteLine("To continue, press Enter"); Console.ReadKey();
          Console.WriteLine("Retrieving...");
          RetrieveArchive(client);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      finally
      {
        // Delete SNS topic and SQS queue.
        snsClient.DeleteTopic(new DeleteTopicRequest() { TopicArn = topicArn });
        sqsClient.DeleteQueue(new DeleteQueueRequest() { QueueUrl = queueUrl });
      }
    }

    static void SetupTopicAndQueue()
    {
      snsClient = new
AmazonSimpleNotificationServiceClient(Amazon.RegionEndpoint.USWest2);
      sqsClient = new AmazonSQSClient(Amazon.RegionEndpoint.USWest2);

      long ticks = DateTime.Now.Ticks;
      topicArn = snsClient.CreateTopic(new CreateTopicRequest { Name =
"GlacierDownload-" + ticks }).TopicArn;
      Console.Write("topicArn: "); Console.WriteLine(topicArn);

      CreateQueueRequest createQueueRequest = new CreateQueueRequest();
      createQueueRequest.QueueName = "GlacierDownload-" + ticks;
      CreateQueueResponse createQueueResponse =
sqsClient.CreateQueue(createQueueRequest);
      queueUrl = createQueueResponse.QueueUrl;
      Console.Write("QueueURL: "); Console.WriteLine(queueUrl);
```

```
     GetQueueAttributesRequest getQueueAttributesRequest = new
GetQueueAttributesRequest();
     getQueueAttributesRequest.AttributeNames = new List<string> { "QueueArn" };
     getQueueAttributesRequest.QueueUrl = queueUrl;
     GetQueueAttributesResponse response =
sqsClient.GetQueueAttributes(getQueueAttributesRequest);
     queueArn = response.QueueARN;
     Console.Write("QueueArn: "); Console.WriteLine(queueArn);

     // Setup the Amazon SNS topic to publish to the SQS queue.
     snsClient.Subscribe(new SubscribeRequest()
     {
       Protocol = "sqs",
       Endpoint = queueArn,
       TopicArn = topicArn
     });

     // Add policy to the queue so SNS can send messages to the queue.
     var policy = SQS_POLICY.Replace("{TopicArn}", topicArn).Replace("{QueueArn}",
queueArn);

     sqsClient.SetQueueAttributes(new SetQueueAttributesRequest()
     {
         QueueUrl = queueUrl,
         Attributes = new Dictionary<string, string>
         {
             { QueueAttributeName.Policy, policy }
         }
     });
   }

   static void RetrieveArchive(AmazonGlacierClient client)
   {
     // Initiate job.
     InitiateJobRequest initJobRequest = new InitiateJobRequest()
     {
       VaultName = vaultName,
       JobParameters = new JobParameters()
       {
         Type = "archive-retrieval",
         ArchiveId = archiveID,
         Description = "This job is to download archive.",
         SNSTopic = topicArn,
```

```
        }
      };
      InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
      string jobId = initJobResponse.JobId;

      // Check queue for a message and if job completed successfully, download archive.
      ProcessQueue(jobId, client);
    }

    private static void ProcessQueue(string jobId, AmazonGlacierClient client)
    {
      ReceiveMessageRequest receiveMessageRequest = new ReceiveMessageRequest()
{ QueueUrl = queueUrl, MaxNumberOfMessages = 1 };
      bool jobDone = false;
      while (!jobDone)
      {
        Console.WriteLine("Poll SQS queue");
        ReceiveMessageResponse receiveMessageResponse =
sqsClient.ReceiveMessage(receiveMessageRequest);
        if (receiveMessageResponse.Messages.Count == 0)
        {
          Thread.Sleep(10000 * 60);
          continue;
        }
        Console.WriteLine("Got message");
        Message message = receiveMessageResponse.Messages[0];
        Dictionary<string, string> outerLayer =
JsonConvert.DeserializeObject<Dictionary<string, string>>(message.Body);
        Dictionary<string, object> fields =
JsonConvert.DeserializeObject<Dictionary<string, object>>(outerLayer["Message"]);
        string statusCode = fields["StatusCode"] as string;

        if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_SUCCEEDED,
StringComparison.InvariantCultureIgnoreCase))
        {
          Console.WriteLine("Downloading job output");
          DownloadOutput(jobId, client); // Save job output to the specified file
location.
        }
        else if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_FAILED,
StringComparison.InvariantCultureIgnoreCase))
          Console.WriteLine("Job failed... cannot download the archive.");

        jobDone = true;
```

```
        sqsClient.DeleteMessage(new DeleteMessageRequest() { QueueUrl = queueUrl,
 ReceiptHandle = message.ReceiptHandle });
      }
    }

    private static void DownloadOutput(string jobId, AmazonGlacierClient client)
    {
      GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
      {
        JobId = jobId,
        VaultName = vaultName
      };

      GetJobOutputResponse getJobOutputResponse =
 client.GetJobOutput(getJobOutputRequest);
      using (Stream webStream = getJobOutputResponse.Body)
      {
          using (Stream fileToSave = File.OpenWrite(fileName))
          {
              CopyStream(webStream, fileToSave);
          }
      }
    }

    public static void CopyStream(Stream input, Stream output)
    {
      byte[] buffer = new byte[65536];
      int length;
      while ((length = input.Read(buffer, 0, buffer.Length)) > 0)
      {
        output.Write(buffer, 0, length);
      }
    }
  }
}
```

Beispiel 2: Abrufen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET—Download Output in Chunks

Das folgende C#-Codebeispiel ruft ein Archiv von Amazon Glacier ab. Der Beispielcode gibt den abzurufenden Bytebereich in einem `GetJobOutputRequest`-Objekt an und lädt die Auftragsausgabe in Datenblöcken herunter.

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
using Amazon.SQS;
using Amazon.SQS.Model;
using Newtonsoft.Json;
using System.Collections.Specialized;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDownloadLowLevelUsingSQLSNSOutputUsingRange
  {
    static string topicArn;
    static string queueUrl;
    static string queueArn;
    static string vaultName = "*** Provide vault name ***";
    static string archiveId = "*** Provide archive ID ***";
    static string fileName  = "*** Provide the file name and path to where to store
 downloaded archive ***";
    static AmazonSimpleNotificationServiceClient snsClient;
    static AmazonSQSClient sqsClient;
    const string SQS_POLICY =
        "{" +
        "    \"Version\" : \"2012-10-17\",&TCX5-2025-waiver;" +
        "    \"Statement\" : [" +
        "        {" +
        "            \"Sid\" : \"sns-rule\"," +
        "            \"Effect\" : \"Allow\"," +
        "            \"Principal\" : {\"AWS\" : \"arn:aws:iam::123456789012:root\" },"
 +
        "            \"Action\"  : \"sqs:SendMessage\"," +
        "            \"Resource\"  : \"{QuernArn}\"," +
        "            \"Condition\" : {" +
        "                \"ArnLike\" : {" +
        "                    \"aws:SourceArn\" : \"{TopicArn}\"" +
        "                }" +
```

```
        "            }" +
        "        }" +
        "    ]" +
        "}";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;

      try
      {
          using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
          {
              Console.WriteLine("Setup SNS topic and SQS queue.");
              SetupTopicAndQueue();
              Console.WriteLine("To continue, press Enter"); Console.ReadKey();

              Console.WriteLine("Download archive");
              DownloadAnArchive(archiveId, client);
      }
      Console.WriteLine("Operations successful. To continue, press Enter");
      Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      finally
      {
        // Delete SNS topic and SQS queue.
        snsClient.DeleteTopic(new DeleteTopicRequest() { TopicArn = topicArn });
        sqsClient.DeleteQueue(new DeleteQueueRequest() { QueueUrl = queueUrl });
      }
    }

       static void SetupTopicAndQueue()
    {
      long ticks = DateTime.Now.Ticks;

      // Setup SNS topic.
      snsClient = new
AmazonSimpleNotificationServiceClient(Amazon.RegionEndpoint.USWest2);
      sqsClient = new AmazonSQSClient(Amazon.RegionEndpoint.USWest2);
```

```
      topicArn = snsClient.CreateTopic(new CreateTopicRequest { Name =
"GlacierDownload-" + ticks }).TopicArn;
      Console.Write("topicArn: "); Console.WriteLine(topicArn);

      CreateQueueRequest createQueueRequest = new CreateQueueRequest();
      createQueueRequest.QueueName = "GlacierDownload-" + ticks;
      CreateQueueResponse createQueueResponse =
sqsClient.CreateQueue(createQueueRequest);
      queueUrl = createQueueResponse.QueueUrl;
      Console.Write("QueueURL: "); Console.WriteLine(queueUrl);

      GetQueueAttributesRequest getQueueAttributesRequest = new
GetQueueAttributesRequest();
      getQueueAttributesRequest.AttributeNames = new List<string> { "QueueArn" };
      getQueueAttributesRequest.QueueUrl = queueUrl;
      GetQueueAttributesResponse response =
sqsClient.GetQueueAttributes(getQueueAttributesRequest);
      queueArn = response.QueueARN;
      Console.Write("QueueArn: "); Console.WriteLine(queueArn);

      // Setup the Amazon SNS topic to publish to the SQS queue.
      snsClient.Subscribe(new SubscribeRequest()
      {
        Protocol = "sqs",
        Endpoint = queueArn,
        TopicArn = topicArn
      });

      // Add the policy to the queue so SNS can send messages to the queue.
      var policy = SQS_POLICY.Replace("{TopicArn}", topicArn).Replace("{QuernArn}",
queueArn);

      sqsClient.SetQueueAttributes(new SetQueueAttributesRequest()
      {
          QueueUrl = queueUrl,
          Attributes = new Dictionary<string, string>
          {
              { QueueAttributeName.Policy, policy }
          }
      });
    }

    static void DownloadAnArchive(string archiveId, AmazonGlacierClient client)
    {
```

```
      // Initiate job.
      InitiateJobRequest initJobRequest = new InitiateJobRequest()
      {

        VaultName = vaultName,
        JobParameters = new JobParameters()
        {
          Type = "archive-retrieval",
          ArchiveId = archiveId,
          Description = "This job is to download the archive.",
          SNSTopic = topicArn,
        }
      };
      InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
      string jobId = initJobResponse.JobId;

      // Check queue for a message and if job completed successfully, download archive.
      ProcessQueue(jobId, client);
    }

    private static void ProcessQueue(string jobId, AmazonGlacierClient client)
    {
        var receiveMessageRequest = new ReceiveMessageRequest() { QueueUrl = queueUrl,
MaxNumberOfMessages = 1 };
        bool jobDone = false;
        while (!jobDone)
        {
            Console.WriteLine("Poll SQS queue");
            ReceiveMessageResponse receiveMessageResponse =
sqsClient.ReceiveMessage(receiveMessageRequest);
            if (receiveMessageResponse.Messages.Count == 0)
            {
                Thread.Sleep(10000 * 60);
                continue;
            }
            Console.WriteLine("Got message");
            Message message = receiveMessageResponse.Messages[0];
            Dictionary<string, string> outerLayer =
JsonConvert.DeserializeObject<Dictionary<string, string>>(message.Body);
            Dictionary<string, object> fields =
JsonConvert.DeserializeObject<Dictionary<string, object>>(outerLayer["Message"]);
            string statusCode = fields["StatusCode"] as string;
            if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_SUCCEEDED,
StringComparison.InvariantCultureIgnoreCase))
```

```
            {
                long archiveSize = Convert.ToInt64(fields["ArchiveSizeInBytes"]);
                Console.WriteLine("Downloading job output");
                DownloadOutput(jobId, archiveSize, client); // This where we save job
output to the specified file location.
            }
            else if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_FAILED,
StringComparison.InvariantCultureIgnoreCase))
                Console.WriteLine("Job failed... cannot download the archive.");
            jobDone = true;
            sqsClient.DeleteMessage(new DeleteMessageRequest() { QueueUrl = queueUrl,
ReceiptHandle = message.ReceiptHandle });
        }
    }

    private static void DownloadOutput(string jobId, long archiveSize,
AmazonGlacierClient client)
    {
      long partSize = 4 * (long)Math.Pow(2, 20);  // 4 MB.
      using (Stream fileToSave = new FileStream(fileName, FileMode.Create,
FileAccess.Write))
      {

        long currentPosition = 0;
        do
        {
          GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
          {
            JobId = jobId,
            VaultName = vaultName
          };

          long endPosition = currentPosition + partSize - 1;
          if (endPosition > archiveSize)
            endPosition = archiveSize;

          getJobOutputRequest.SetRange(currentPosition, endPosition);
          GetJobOutputResponse getJobOutputResponse =
client.GetJobOutput(getJobOutputRequest);

          using (Stream webStream = getJobOutputResponse.Body)
          {
            CopyStream(webStream, fileToSave);
          }
```

```
          currentPosition += partSize;
        } while (currentPosition < archiveSize);
      }
    }

    public static void CopyStream(Stream input, Stream output)
    {
      byte[] buffer = new byte[65536];
      int length;
      while ((length = input.Read(buffer, 0, buffer.Length)) > 0)
      {
        output.Write(buffer, 0, length);
      }
    }
  }
}
```

Herunterladen eines großen Archivs mithilfe der Parallelverarbeitung mit Python

In diesem Thema wird beschrieben, wie Sie mithilfe der Parallelverarbeitung mit Python ein großes Archiv von Amazon S3 Glacier (S3 Glacier) herunterladen. Mit diesem Ansatz können Sie Archive jeder Größe zuverlässig herunterladen, indem Sie sie in kleinere Teile aufteilen, die unabhängig voneinander verarbeitet werden können.

-Übersicht

Das in diesem Beispiel bereitgestellte Python-Skript führt die folgenden Aufgaben aus:

1. Richtet die erforderlichen AWS Ressourcen (Amazon SNS SNS-Thema und Amazon SQS SQS-Warteschlangen) für Benachrichtigungen ein
2. Initiiert einen Job zum Abrufen von Archiven mit Amazon Glacier
3. Überwacht eine Amazon SQS SQS-Warteschlange für Benachrichtigungen über den Abschluss von Jobs
4. Teilt das große Archiv in verwaltbare Teile auf
5. Lädt Chunks parallel unter Verwendung mehrerer Worker-Threads herunter
6. Speichert jeden Chunk auf der Festplatte, damit er später wieder zusammengebaut werden kann

Voraussetzungen

Bevor Sie beginnen, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Python 3.6 oder höher installiert
- AWS SDK for Python (Boto3) installiert
- AWS Anmeldeinformationen, die mit den entsprechenden Berechtigungen für Amazon Glacier, Amazon SNS und Amazon SQS konfiguriert sind
- Ausreichend Festplattenspeicher zum Speichern der heruntergeladenen Archivblöcke

Beispiel: Herunterladen eines Archivs mithilfe von Parallelverarbeitung mit Python

Das folgende Python-Skript veranschaulicht, wie ein großes Archiv mithilfe von Parallelverarbeitung von Amazon Glacier heruntergeladen wird:

```
import boto3
import time
import json
import jmespath
import re
import concurrent.futures
import os

output_file_path = "output_directory_path"
vault_name = "vault_name"

chunk_size = 1000000000 #1gb - size of chunks for parallel download.
notify_queue_name = 'GlacierJobCompleteNotifyQueue' # SQS queue for Glacier recall
 notification
chunk_download_queue_name='GlacierChunkReadyNotifyQueue' # SQS queue for chunks
sns_topic_name = 'GlacierRecallJobCompleted' # the SNS topic to be notified when
 Glacier archive is restored.
chunk_queue_visibility_timeout = 7200 # 2 hours - this may need to be adjusted.
region = 'us-east-1'
archive_id = "archive_id_to_restore"
retrieve_archive = True # set to false if you do not want to restore from Glacier -
 useful for restarting or parallel processing of the chunk queue.
workers = 12 # the number of parallel worker threads for downloading chunks.

def setup_queues_and_topic():
```

```
    sqs = boto3.client('sqs')
    sns = boto3.client('sns')

    # Create the SNS topic
    topic_response = sns.create_topic(
        Name=sns_topic_name
    )
    topic_arn = topic_response['TopicArn']
    print("Creating the SNS topic " + topic_arn)

    # Create the notification queue
    notify_queue_response = sqs.create_queue(
        QueueName=notify_queue_name,
        Attributes={
            'VisibilityTimeout': '300',  # 5 minutes
            'ReceiveMessageWaitTimeSeconds': '20'  # Enable long polling
        }
    )
    notify_queue_url = notify_queue_response['QueueUrl']
    print("Creating the archive-retrieval notification queue " + notify_queue_url)

    # Create the chunk download queue
    chunk_queue_response = sqs.create_queue(
        QueueName=chunk_download_queue_name,
        Attributes={
            'VisibilityTimeout': str(chunk_queue_visibility_timeout),  # 5 minutes
            'ReceiveMessageWaitTimeSeconds': '0'
        }
    )
    chunk_queue_url = chunk_queue_response['QueueUrl']

    print("Creating the chunk ready notification queue " + chunk_queue_url)


   # Get the ARN for the notification queue
    notify_queue_attributes = sqs.get_queue_attributes(
        QueueUrl=notify_queue_url,
        AttributeNames=['QueueArn']
    )
    notify_queue_arn = notify_queue_attributes['Attributes']['QueueArn']

    # Set up the SNS topic policy on the notification queue
    queue_policy = {
        "Version": "2012-10-17",
```

```
        "Statement": [{
            "Sid": "allow-sns-messages",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": "SQS:SendMessage",
            "Resource": notify_queue_arn,
            "Condition": {
                "ArnEquals": {
                    "aws:SourceArn": topic_arn
                }
            }
        }]
    }

    # Set the queue policy
    sqs.set_queue_attributes(
        QueueUrl=notify_queue_url,
        Attributes={
            'Policy': json.dumps(queue_policy)
        }
    )

    # Subscribe the notification queue to the SNS topic
    sns.subscribe(
        TopicArn=topic_arn,
        Protocol='sqs',
        Endpoint=notify_queue_arn
    )

    return {
        'topic_arn': topic_arn,
        'notify_queue_url': notify_queue_url,
        'chunk_queue_url': chunk_queue_url
    }


def split_and_send_chunks(archive_size, job_id,chunk_queue_url):
    ranges = []
    current = 0
    chunk_number = 0

    while current < archive_size:
        chunk_number += 1
        next_range = min(current + chunk_size - 1, archive_size - 1)
```

```
        ranges.append((current, next_range, chunk_number))
        current = next_range + 1

    # Send messages to SQS queue
    for start, end, chunk_number in ranges:
        body = {"start": start, "end": end, "job_id": job_id, "chunk_number":
 chunk_number}
        body = json.dumps(body)
        print("Sending SQS message for range:" + str(body))
        response = sqs.send_message(
            QueueUrl=chunk_queue_url,
            MessageBody=str(body)
        )

def GetJobOutputChunks(job_id, byterange, chunk_number):
    glacier = boto3.client('glacier')
    response = glacier.get_job_output(
        vaultName=vault_name,
        jobId=job_id,
        range=byterange,

    )

    with open(os.path.join(output_file_path,str(chunk_number)+".chunk"), 'wb') as
 output_file:
        output_file.write(response['body'].read())

    return response

def ReceiveArchiveReadyMessages(notify_queue_url,chunk_queue_url):

    response = sqs.receive_message(
        QueueUrl=notify_queue_url,
        AttributeNames=['All'],
        MaxNumberOfMessages=1,
        WaitTimeSeconds=20,
        MessageAttributeNames=['Message']
    )
    print("Polling archive retrieval job ready queue...")
    # Checking that there is a Messages key before proceeding. No 'Messages' key likely
 means the queue is empty

    if 'Messages' in response:
        print("Received a message from the archive retrieval job queue")
```

```
        jsonresponse = response
        # Loading the string into JSON and checking that ArchiveSizeInBytes key is
 present before continuing.
        jsonresponse=json.loads(jsonresponse['Messages'][0]['Body'])
        jsonresponse=json.loads(jsonresponse['Message'])
        if 'ArchiveSizeInBytes' in jsonresponse:
            receipt_handle = response['Messages'][0]['ReceiptHandle']
            if jsonresponse['ArchiveSizeInBytes']:
                archive_size = jsonresponse['ArchiveSizeInBytes']

                print(f'Received message: {response}')
                if archive_size > chunk_size:
                    split_and_send_chunks(archive_size,
 jsonresponse['JobId'],chunk_queue_url)

                    sqs.delete_message(
                    QueueUrl=notify_queue_url,
                    ReceiptHandle=receipt_handle)

            else:
                print("No ArchiveSizeInBytes value found in message")
                print(response)

    else:
        print('No messages available in the queue at this time.')

    time.sleep(1)

def ReceiveArchiveChunkMessages(chunk_queue_url):
    response = sqs.receive_message(
        QueueUrl=chunk_queue_url,
        AttributeNames=['All'],
        MaxNumberOfMessages=1,
        WaitTimeSeconds=0,
        MessageAttributeNames=['Message']
    )
    print("Polling archive chunk queue...")
    print(response)
    # Checking that there is a Messages key before proceeding. No 'Messages' key likely
 means the queue is empty
    if 'Messages' in response:
        jsonresponse = response
        # Loading the string into JSON and checking that ArchiveSizeInBytes key is
 present before continuing.
```

```
        jsonresponse=json.loads(jsonresponse['Messages'][0]['Body'])
        if 'job_id' in jsonresponse: #checking that there is a job id before continuing
            job_id = jsonresponse['job_id']
            byterange = "bytes="+str(jsonresponse['start']) + '-' +
 str(jsonresponse['end'])
            chunk_number = jsonresponse['chunk_number']
            receipt_handle = response['Messages'][0]['ReceiptHandle']
            if jsonresponse['job_id']:
                print(f'Received message: {response}')
                GetJobOutputChunks(job_id,byterange,chunk_number)
                sqs.delete_message(
                QueueUrl=chunk_queue_url,
                ReceiptHandle=receipt_handle)
    else:
        print('No messages available in the chunk queue at this time.')

def initiate_archive_retrieval(archive_id, topic_arn):
    glacier = boto3.client('glacier')

    job_parameters = {
        "Type": "archive-retrieval",
        "ArchiveId": archive_id,
        "Description": "Archive retrieval job",
        "SNSTopic": topic_arn,
        "Tier": "Bulk"  # You can change this to "Standard" or "Expedited" based on
 your needs
    }

    try:
        response = glacier.initiate_job(
            vaultName=vault_name,
            jobParameters=job_parameters
        )

        print("Archive retrieval job initiated:")
        print(f"Job ID: {response['jobId']}")
        print(f"Job parameters: {job_parameters}")
        print(f"Complete response: {json.dumps(response, indent=2)}")

        return response['jobId']

    except Exception as e:
        print(f"Error initiating archive retrieval job: {str(e)}")
        raise
```

```
def run_async_tasks(chunk_queue_url, workers):
    max_workers = workers  # Set the desired maximum number of concurrent tasks
    with concurrent.futures.ThreadPoolExecutor(max_workers=max_workers) as executor:
        for _ in range(max_workers):
            executor.submit(ReceiveArchiveChunkMessages, chunk_queue_url)

# One time setup of the necessary queues and topics.
queue_and_topic_atts = setup_queues_and_topic()

topic_arn = queue_and_topic_atts['topic_arn']
notify_queue_url = queue_and_topic_atts['notify_queue_url']
chunk_queue_url = queue_and_topic_atts['chunk_queue_url']

if retrieve_archive:
    print("Retrieving the defined archive... The topic arn we will notify when
 recalling the archive is: "+topic_arn)
    job_id = initiate_archive_retrieval(archive_id, topic_arn)
else:
    print("Retrieve archive is false, polling queues and downloading only.")

while True:
   ReceiveArchiveReadyMessages(notify_queue_url,chunk_queue_url)
   run_async_tasks(chunk_queue_url,workers)
```

Verwenden des Skripts

Gehen Sie wie folgt vor, um dieses Skript zu verwenden:

1. Ersetzen Sie die Platzhalterwerte im Skript durch Ihre spezifischen Informationen:
   - *output_file_path*: Verzeichnis, in dem Chunk-Dateien gespeichert werden
   - *vault_name*: Name Ihres S3 Glacier-Tresors
   - *notify_queue_name*: Name für die Warteschlange für Jobbenachrichtigungen
   - *chunk_download_queue_name*: Name für die Chunk-Download-Warteschlange
   - *sns_topic_name*: Name für das SNS-Thema
   - *region*: AWS Region, in der sich Ihr Tresor befindet
   - *archive_id*: ID des abzurufenden Archivs
2. Führen Sie das Skript aus:

```
python download_large_archive.py
```

3. Nachdem alle Chunks heruntergeladen wurden, können Sie sie mit einem Befehl wie dem folgenden zu einer einzigen Datei kombinieren:

```
cat /path/to/chunks/*.chunk > complete_archive.file
```

Wichtige Überlegungen

Beachten Sie bei der Verwendung dieses Skripts Folgendes:

- Das Abrufen von Archiven aus S3 Glacier kann je nach ausgewählter Abrufstufe mehrere Stunden dauern.
- Das Skript wird auf unbestimmte Zeit ausgeführt und fragt kontinuierlich die Warteschlangen ab. Möglicherweise möchten Sie eine Kündigungsbedingung hinzufügen, die Ihren spezifischen Anforderungen entspricht.
- Stellen Sie sicher, dass Sie über ausreichend Festplattenspeicher verfügen, um alle Teile Ihres Archivs zu speichern.
- Wenn das Skript unterbrochen wird, können Sie es mit neu starten, `retrieve_archive=False` um mit dem Herunterladen von Chunks fortzufahren, ohne einen neuen Abrufjob zu starten.
- Passen Sie die *`workers`* Parameter *`chunk_size`* und an Ihre Netzwerkbandbreite und Systemressourcen an.
- Für Amazon S3-Abrufe, Amazon SNS- und Amazon SQS SQS-Nutzung AWS fallen Standardgebühren an.

Herunterladen eines Archivs mit der REST-API

So laden Sie ein Archiv mit der REST-API herunter

Das Herunterladen eines Archivs ist ein zweistufiger Prozess.

1. Initiieren Sie einen Auftrag des Typs `archive-retrieval`. Weitere Informationen finden Sie unter Initiate Job (POST jobs).
2. Laden Sie die Archivdaten herunter, wenn der Auftrag abgeschlossen wurde. Weitere Informationen finden Sie unter Get Job Output (GET output).

Herunterladen eines Archivs in Amazon Glacier mit dem AWS CLI

Sie können Archive in Amazon Glacier (Amazon Glacier) mit AWS Command Line Interface (AWS CLI) herunterladen.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Laden Sie ein Archiv herunter mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface

2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

     ```
     aws help
     ```

   - Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

     ```
     aws configure list
     ```

Beispiel: Laden Sie ein Archiv herunter mit dem AWS CLI

> **Note**
>
> Um Ihre Archive herunterzuladen, müssen Sie die Archiv-IDs kennen. Mit den Schritten 1–4 werden die Archiv-IDs abgerufen. Wenn Sie IDs der Archive, die Sie herunterladen möchten, bereits kennen, fahren Sie mit Schritt 5 fort.

1. Verwenden Sie den `initiate-job`-Befehl, um einen Inventarabrufauftrag zu starten. Im Inventarbericht werden die Archiv-IDs aufgeführt.

   ```
   aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
   job-parameters="{\"Type\":\"inventory-retrieval\"}"
   ```

   Erwartete Ausgabe:

   ```
   {
       "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
       "jobId": "*** jobid ***"
   }
   ```

2. Verwenden Sie den `describe-job`-Befehl, um den Status des vorherigen -Auftragsbefehls zu überprüfen.

   ```
   aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
   job-id *** jobid ***
   ```

   Erwartete Ausgabe:

   ```
   {
       "InventoryRetrievalParameters": {
           "Format": "JSON"
       },
       "VaultARN": "*** vault arn ***",
       "Completed": false,
       "JobId": "*** jobid ***",
       "Action": "InventoryRetrieval",
   ```

```
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie beim Initiieren des Jobs eine Benachrichtigungskonfiguration für den Tresor festgelegt oder ein Amazon Simple Notification Service (Amazon SNS) -Thema angegeben haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an das Thema.

   Sie können Benachrichtigungskonfiguration für bestimmte Ereignisse auf dem Tresor einrichten. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier. Amazon Glacier sendet bei jedem Eintreten eines bestimmten Ereignisses eine Nachricht an das angegebene SNS-Thema.

4. Nach Abschluss laden Sie die Abrufaufgabe mit dem `get-job-output`-Befehl in die Datei „`output.json`“ herunter. Diese Datei enthält Ihre Archiv-IDs.

   ```
   aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
    --job-id *** jobid *** output.json
   ```

   Dieser Befehl erzeugt eine Datei mit den folgenden Feldern.

   ```
   {
   "VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
   "InventoryDate":"*** job completion date ***",
   "ArchiveList":[
   {"ArchiveId":"*** archiveid ***",
   "ArchiveDescription":*** archive description (if set) ***,
   "CreationDate":"*** archive creation date ***",
   "Size":"*** archive size (in bytes) ***",
   "SHA256TreeHash":"*** archive hash ***"
   }
   {"ArchiveId":
   ...
   ]}
   ```

5. Verwenden Sie den `initiate-job`-Befehl, um den Abrufvorgang für jedes Archiv aus einem Tresor zu starten. Sie müssen als Auftragsparameter `archive-retrieval` wie unten dargestellt angeben.

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333
 --job-parameters="{\"Type\":\"archive-retrieval\",\"ArchiveId\":\"*** archiveId
 ***\"}"
```

6. Warten Sie, bis der `archive-retrieval`-Auftrag abgeschlossen wurde. Verwenden Sie den `describe-job`-Befehl, um den Status des vorherigen Befehls zu überprüfen.

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

7. Wenn der oben beschriebene Auftrag abgeschlossen ist, verwenden Sie den `get-job-output`-Befehl, um Ihr Archiv herunterzuladen.

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output_file_name
```

Löschen eines Archivs in Amazon Glacier

Sie können ein Archiv nicht mit der Amazon Glacier (Amazon Glacier) -Managementkonsole löschen. Um ein Archiv zu löschen, müssen Sie die AWS Command Line Interface (CLI) verwenden oder Code schreiben, um eine Löschanforderung entweder direkt über die REST-API oder über die und.NET-Wrapper-Bibliotheken zu stellen. AWS SDK für Java In den folgenden Themen wird erklärt, wie Sie die AWS SDK für Java und .NET-Wrapper-Bibliotheken, die REST-API und die verwenden. AWS CLI

Themen

- Löschen eines Archivs in Amazon Glacier mithilfe der AWS SDK für Java
- Löschen eines Archivs in Amazon Glacier mithilfe der AWS SDK für .NET
- Löschen eines Amazon Glacier-Archivs mithilfe der REST-API
- Löschen eines Archivs in Amazon Glacier mithilfe der AWS Command Line Interface

Sie können zu einem gegebenen Zeitpunkt jeweils nur ein Archiv aus einem Tresor löschen. Zum Löschen des Archivs müssen Sie dessen Archiv-ID in Ihrer Löschanfrage angeben. Sie können

die Archiv-ID ermitteln, indem Sie den Tresorbestand für den Tresor, in dem sich Archiv befindet, herunterladen. Weitere Informationen zum Herunterladen des Tresorbestands finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

Nach der Löschung eines Archivs kann noch eine erfolgreiche Anforderung zur Initiierung eines Auftrags zum Abrufen des gelöschten Archivs möglich sein, der betreffende Auftrag schlägt jedoch fehl.

Wenn Sie ein Archiv löschen und währenddessen noch ein Abruf des Archivs über die Archiv-ID bearbeitet wird, kann der Abruf den folgenden Szenarien entsprechend fehlschlagen oder gelingen:

- Wenn der Archivabruf-Job die Daten aktiv für den Download vorbereitet, wenn Amazon Glacier die Anfrage zum Löschen von Archiven empfängt, schlägt der Archivierungsabruf möglicherweise fehl.
- Wenn der Archivabruf-Job das Archiv erfolgreich für den Download vorbereitet hat, als Amazon Glacier die Anfrage zum Löschen des Archivs empfängt, können Sie die Ausgabe herunterladen.

Weitere Informationen zum Archivabruf finden Sie unter Ein Archiv in Amazon Glacier herunterladen.

Dieser Vorgang ist idempotent. Das Löschen eines bereits gelöschten Archivs erzeugt keinen Fehler.

Wenn Sie nach dem Löschen eines Archivs das Tresorinventar sofort herunterladen, wird das gelöschte Archiv möglicherweise in die Liste aufgenommen, da Amazon Glacier das Tresorinventar nur etwa einmal täglich erstellt.

> **Note**
>
> Informationen zum automatisierten Löschen von Tresorarchiven finden Sie unter Automatisiertes Löschen von Tresorarchiven in Amazon S3 Glacier.

Löschen eines Archivs in Amazon Glacier mithilfe der AWS SDK für Java

Im Folgenden werden die Schritte zum Löschen eines Archivs mithilfe der AWS SDK für Java Low-Level-API beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

Sie müssen eine AWS Region angeben, in der das Archiv, das Sie löschen möchten, gespeichert ist. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.

2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `DeleteArchiveRequest`-Klasse.

   Sie müssen eine Archiv-ID, einen Tresornamen und Ihre Konto-ID angeben. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier.
3. Führen Sie die `deleteArchive`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

Der folgende Java-Codeausschnitt veranschaulicht die vorherigen Schritte.

```
AmazonGlacierClient client;

DeleteArchiveRequest request = new DeleteArchiveRequest()
    .withVaultName("*** provide a vault name ***")
    .withArchiveId("*** provide an archive ID ***");

client.deleteArchive(request);
```

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Delete Archive (DELETE archive).

Beispiel: Löschen eines Archivs mit dem AWS SDK für Java

Das folgende Java-Codebeispiel verwendet den AWS SDK für Java , um ein Archiv zu löschen. step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Java-Beispielen für Amazon Glacier mit Eclipse. Sie müssen den Code wie gezeigt mit einem Tresornamen und der Archiv-ID des zu löschenden Archivs aktualisieren.

Example

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.DeleteArchiveRequest;

public class ArchiveDelete {

    public static String vaultName = "*** provide vault name ****";
    public static String archiveId = "*** provide archive ID***";
    public static AmazonGlacierClient client;

    public static void main(String[] args) throws IOException {

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        try {

            // Delete the archive.
            client.deleteArchive(new DeleteArchiveRequest()
                .withVaultName(vaultName)
                .withArchiveId(archiveId));

            System.out.println("Deleted archive successfully.");

        } catch (Exception e) {
            System.err.println("Archive not deleted.");
            System.err.println(e);
        }
    }
}
```

Löschen eines Archivs in Amazon Glacier mithilfe der AWS SDK für .NET

Sowohl die High-Level- als auch die Low-Level-Version, die vom Amazon SDK for .NET APIs bereitgestellt werden, bieten eine Methode zum Löschen eines Archivs.

Themen

- Löschen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET
- Löschen eines Archivs mithilfe der Low-Level-API AWS SDK für .NET

Löschen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

Die `ArchiveTransferManager`-Klasse der High-Level-API bietet die `DeleteArchive`-Methode, mit der Sie ein Archiv löschen können.

Beispiel: Löschen eines Archivs mithilfe der High-Level-API von AWS SDK für .NET

Das folgende C#-Codebeispiel verwendet die High-Level-API von AWS SDK für .NET , um ein Archiv zu löschen. step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit der Archiv-ID des zu löschenden Archivs aktualisieren.

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDeleteHighLevel
  {
    static string vaultName = "examplevault";
    static string archiveId = "*** Provide archive ID ***";

    public static void Main(string[] args)
    {
      try
      {
        var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
        manager.DeleteArchive(vaultName, archiveId);
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
```

```
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

Löschen eines Archivs mithilfe der Low-Level-API AWS SDK für .NET

Im Folgenden werden die Schritte zum Löschen eines Archivs mit AWS SDK für .NET beschrieben.

1. Erstellen einer Instance der `AmazonGlacierClient`-Klasse (Client).

   Sie müssen eine AWS Region angeben, in der das Archiv, das Sie löschen möchten, gespeichert ist. Alle Operationen, die Sie mit diesem Client ausführen, gelten für diese AWS Region.
2. Um die Anforderungsinformationen bereitzustellen, erstellen Sie eine Instance der `DeleteArchiveRequest`-Klasse.

   Sie müssen eine Archiv-ID, einen Tresornamen und Ihre Konto-ID angeben. Wenn Sie keine Konto-ID angeben, wird die Konto-ID übernommen, die den Anmeldeinformationen zugeordnet ist, die Sie zum Signieren der Anforderung verwenden. Weitere Informationen finden Sie unter Verwenden von AWS SDKs mit Amazon Glacier.
3. Führen Sie die `DeleteArchive`-Methode aus, indem das Anforderungsobjekt als Parameter festgelegt wird.

Beispiel: Löschen eines Archivs mithilfe der Low-Level-API von AWS SDK für .NET

Im folgenden C#-Codebeispiel werden die vorstehenden Schritte veranschaulicht. Das Beispiel verwendet die Low-Level-API von, um ein AWS SDK für .NET Archiv zu löschen.

> **Note**
>
> Weitere Informationen zur zugrunde liegenden REST-API finden Sie unter Delete Archive (DELETE archive).

step-by-stepAnweisungen zur Ausführung dieses Beispiels finden Sie unterAusführen von Codebeispielen. Sie müssen den Code wie gezeigt mit der Archiv-ID des zu löschenden Archivs aktualisieren.

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDeleteLowLevel
  {
    static string vaultName = "examplevault";
    static string archiveId = "*** Provide archive ID ***";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Deleting the archive");
          DeleteAnArchive(client);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static void DeleteAnArchive(AmazonGlacierClient client)
    {
      DeleteArchiveRequest request = new DeleteArchiveRequest()
      {
        VaultName = vaultName,
        ArchiveId = archiveId
      };
      DeleteArchiveResponse response = client.DeleteArchive(request);
    }
  }
```

```
}
```

Löschen eines Amazon Glacier-Archivs mithilfe der REST-API

Sie können die API zum Löschen von Archiven verwenden, um ein Archiv zu löschen.

- Informationen zur API zum Löschen von Archiven finden Sie unter Delete Archive (DELETE archive).
- Weitere Informationen zur Verwendung der REST-API finden Sie unter API-Referenz für Amazon Glacier.

Löschen eines Archivs in Amazon Glacier mithilfe der AWS Command Line Interface

Sie können Archive in Amazon Glacier (Amazon Glacier) mit AWS Command Line Interface (AWS CLI) löschen.

Themen

- (Voraussetzung) Einrichtung des AWS CLI
- Beispiel: Löschen eines Archivs mit dem AWS CLI

(Voraussetzung) Einrichtung des AWS CLI

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:

   Installation des AWS Command Line Interface

   Konfiguration des AWS Command Line Interface
2. Überprüfen Sie Ihr AWS CLI Setup, indem Sie die folgenden Befehle an der Befehlszeile eingeben. Diese Befehle stellen nicht explizit Anmeldeinformationen bereit, daher werden die Anmeldeinformationen des Standardprofils verwendet.

   - Versuchen Sie, den Befehl help zu verwenden.

   ```
   aws help
   ```

- Verwenden Sie den `list-vaults` Befehl, um eine Liste der Amazon Glacier-Tresore für das konfigurierte Konto abzurufen. Ersetzen Sie es *123456789012* durch Ihre AWS-Konto ID.

```
aws glacier list-vaults --account-id 123456789012
```

- Verwenden Sie den `aws configure list` Befehl AWS CLI, um die aktuellen Konfigurationsdaten für zu sehen.

```
aws configure list
```

Beispiel: Löschen eines Archivs mit dem AWS CLI

1. Verwenden Sie den initiate-job-Befehl, um eine Lagerabrufaufgabe zu starten.

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters="{\"Type\":\"inventory-retrieval\"}"
```

Erwartete Ausgabe:

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. Verwenden Sie den describe-job-Befehl, um den Status der vorherigen Abrufaufgabe zu überprüfen.

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

Erwartete Ausgabe:

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
```

```
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. Warten Sie, bis der -Auftrag abgeschlossen wurde.

   Sie müssen warten, bis die Ausgabe des Auftrags zum Download bereit ist. Wenn Sie beim Initiieren des Jobs eine Benachrichtigungskonfiguration für den Tresor festgelegt oder ein Amazon Simple Notification Service (Amazon SNS) -Thema angegeben haben, sendet Amazon Glacier nach Abschluss des Jobs eine Nachricht an das Thema.

   Sie können Benachrichtigungskonfiguration für bestimmte Ereignisse auf dem Tresor einrichten. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier. Amazon Glacier sendet bei jedem Eintreten eines bestimmten Ereignisses eine Nachricht an das angegebene SNS-Thema.

4. Nach Abschluss laden Sie die Abrufaufgabe mit dem get-job-output-Befehl in die Datei „`output.json`“ herunter.

   ```
   aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
    --job-id *** jobid *** output.json
   ```

   Dieser Befehl erzeugt eine Datei mit den folgenden Feldern.

   ```
   {
   "VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
   "InventoryDate":"*** job completion date ***",
   "ArchiveList":[
   {"ArchiveId":"*** archiveid ***",
   "ArchiveDescription":*** archive description (if set) ***,
   "CreationDate":"*** archive creation date ***",
   "Size":"*** archive size (in bytes) ***",
   "SHA256TreeHash":"*** archive hash ***"
   }
   {"ArchiveId":
   ...
   ```

```
]}
```

5. Verwenden Sie den `delete-archive`-Befehl, um jedes Archiv aus einem Tresor zu löschen, bis kein Archiv übrig bleibt.

```
aws glacier delete-archive --vault-name awsexamplevault --account-id 111122223333
 --archive-id *** archiveid ***
```

Verwenden von AWS SDKs mit Amazon Glacier

AWS bietet Ihnen SDKs die Möglichkeit, Anwendungen für Amazon Glacier zu entwickeln. Die SDK-Bibliotheken umfassen die zugrunde liegende Amazon Glacier-API und vereinfachen so Ihre Programmieraufgaben. Beispielsweise müssen Sie für jede an Amazon Glacier gesendete Anfrage eine Signatur hinzufügen, um Ihre Anfragen zu authentifizieren. Wenn Sie die SDK-Bibliotheken verwenden, müssen Sie nur Ihre AWS Sicherheitsanmeldedaten in Ihrem Code angeben, und die Bibliotheken berechnen die erforderliche Signatur und fügen sie in die an Amazon Glacier gesendete Anfrage ein. AWS SDKs Sie stellen Bibliotheken bereit, die der zugrunde liegenden REST-API zugeordnet sind, und stellen Objekte bereit, mit denen Sie auf einfache Weise Anfragen erstellen und Antworten verarbeiten können.

Themen

- AWS SDK-Bibliotheken für Java und .NET
- Amazon Glacier mit einem AWS SDK verwenden
- Verwenden von AWS SDK für Java mit Amazon Glacier
- Verwenden von AWS SDK für .NET mit Amazon Glacier

Das AWS Command Line Interface (AWS CLI) ist ein einheitliches Tool zur Verwaltung Ihres AWS-Services, einschließlich Amazon Glacier. Informationen zum Herunterladen von finden AWS CLI Sie unter AWS Command Line Interface. Eine Liste der Amazon Glacier CLI-Befehle finden Sie in der AWS CLI Befehlsreferenz.

AWS SDK-Bibliotheken für Java und .NET

Die AWS SDKs für Java und .NET bieten Wrapper-Bibliotheken auf hoher und niedriger Ebene.

Beispiele für die Arbeit mit Amazon Glacier finden Sie im AWS SDK für Java und AWS SDK für .NET überall in diesem Entwicklerhandbuch.

Was ist die Low-Level-&API?

Die Low-Level-Wrapper-Bibliotheken bilden eng die zugrunde liegende REST-API (API-Referenz für Amazon Glacier) ab, die von Amazon Glacier unterstützt wird. Für jeden Amazon Glacier-REST-Vorgang bietet die Low-Level-API eine entsprechende Methode, ein Anforderungsobjekt, mit dem

Sie Anforderungsinformationen bereitstellen können, und ein Antwortobjekt, mit dem Sie die Amazon Glacier-Antwort verarbeiten können. Die Low-Level-Wrapper-Bibliotheken sind die vollständigste Implementierung der zugrunde liegenden Amazon Glacier-Operationen.

Weitere Informationen zu diesen SDK-Bibliotheken finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier und Verwenden von AWS SDK für .NET mit Amazon Glacier.

Was ist die High-Level-&API?

Zur weiteren Vereinfachung der Anwendungsentwicklung bieten diese Bibliotheken für einige Operationen eine Abstraktion auf höherer Ebene. Beispiel:

- Hochladen eines Archivs: Zum Hochladen eines Archivs mit der Low-Level-API müssen Sie zusätzlich zum Dateinamen und dem Namen des Tresors, in dem Sie das Archiv speichern möchten, eine Prüfsumme (SHA-256-Struktur-Hash) der Nutzlast angeben. Die High-Level-API berechnet jedoch die Prüfsumme für Sie.
- Herunterladen eines Archivs oder eines Tresorbestands: Zum Herunterladen eines Archivs mithilfe der Low-Level-API initiieren Sie zunächst einen Auftrag, warten, bis der Auftrag abgeschlossen ist, und rufen dann die Auftragsausgabe ab. Sie müssen zusätzlichen Code schreiben, um ein Amazon Simple Notification Service (Amazon SNS) -Thema einzurichten, damit Amazon Glacier Sie benachrichtigt, wenn der Job abgeschlossen ist. Sie benötigen außerdem einen Abfragemechanismus, der überprüft, ob eine Nachricht über den Abschluss des Auftrags an das Thema übermittelt wurde. Die High-Level-API stellt zum Herunterladen eines Archivs eine Methode bereit, die alle diese Schritte erledigt. Sie geben lediglich eine Archiv-ID und den Pfad zu dem Ordner an, in dem Sie die heruntergeladenen Daten speichern möchten.

Weitere Informationen zu diesen SDK-Bibliotheken finden Sie unter Verwenden von AWS SDK für Java mit Amazon Glacier und Verwenden von AWS SDK für .NET mit Amazon Glacier.

Situationen für die Verwendung der High-Level- und der Low-Level-API

Wenn die High-Level-API Methoden bietet, die Sie zum Ausführen der Operation benötigen, ist es in der Regel ratsam, die High-Level-API zu verwenden, weil damit der gesamte Vorgang vereinfacht wird. Wenn die High-Level-API keine vorteilhaften Funktionen bietet, können Sie die Low-Level-API verwenden. Die Low-Level-API ermöglicht die differenzierte Steuerung der Operation, z. B. mittels einer Logik für Wiederholversuche bei Fehlern. Beim Hochladen eines Archivs verwendet die High-Level-API beispielsweise die Dateigröße, um zu bestimmen, ob das Archiv in einer

einzigen Operation oder mithilfe der API für mehrteilige Uploads hochgeladen wird. Für den Fall, dass der Hochladevorgang fehlschlägt, verfügt die API ebenfalls über eine integrierte Logik für Wiederholversuche. Falls für Ihre Anwendung eine differenzierte Steuerung dieser Entscheidungen erforderlich ist, können Sie die Low-Level-API verwenden.

Amazon Glacier mit einem AWS SDK verwenden

AWS Software Development Kits (SDKs) sind für viele beliebte Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++ Codebeispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK für Go	AWS SDK für Go Code-Beispiele
AWS SDK für Java	AWS SDK für Java Code-Beispiele
AWS SDK für JavaScript	AWS SDK für JavaScript Code-Beispiele
AWS SDK für Kotlin	AWS SDK für Kotlin Code-Beispiele
AWS SDK für .NET	AWS SDK für .NET Code-Beispiele
AWS SDK für PHP	AWS SDK für PHP Code-Beispiele
AWS -Tools für PowerShell	AWS -Tools für PowerShell Code-Beispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3) Code-Beispiele
AWS SDK für Ruby	AWS SDK für Ruby Code-Beispiele
AWS SDK für Rust	AWS SDK für Rust Code-Beispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Code-Beispiele
AWS SDK für Swift	AWS SDK für Swift Code-Beispiele

Spezifische Beispiele für Amazon Glacier finden Sie unterCodebeispiele für Amazon Glacier mit AWS SDKs.

> **Beispiel für die Verfügbarkeit**
>
> Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

Verwenden von AWS SDK für Java mit Amazon Glacier

Das AWS SDK für Java bietet sowohl High-Level als auch Low-Level APIs für Amazon Glacier (Amazon Glacier), wie unter beschriebenVerwenden von AWS SDKs mit Amazon Glacier. Informationen zum Herunterladen von finden Sie unter Amazon SDK for Java. AWS SDK für Java

> **Note**
>
> Das AWS SDK für Java bietet Thread-sichere Clients für den Zugriff auf Amazon Glacier. Als bewährte Methode sollten Ihre Anwendungen einen Client erstellen und diesen zwischen den Threads wiederverwenden.

Themen

- Verwenden der Low-Level-API
- Verwenden der High-Level-API
- Ausführen von Java-Beispielen für Amazon Glacier mit Eclipse
- Festlegen des Endpunkts

Verwenden der Low-Level-API

Die `AmazonGlacierClient` Low-Level-Klasse stellt alle Methoden bereit, die den zugrunde liegenden REST-Operationen von Amazon Glacier (API-Referenz für Amazon Glacier) zugeordnet sind. Wenn Sie eine dieser Methoden aufrufen, müssen Sie ein entsprechendes Anforderungsobjekt erstellen und ein Antwortobjekt bereitstellen, in dem die Methode die Amazon Glacier-Antwort auf den Vorgang zurückgeben kann.

Beispielsweise stellt die `AmazonGlacierClient`-Klasse die `createVault`-Methode für die Erstellung eines Tresors zur Verfügung. Diese Methode lässt sich der zugrunde liegenden REST-Operation „Create Vault" zuordnen (siehe Create Vault (PUT vault)). Um diese Methode zu verwenden, müssen Sie Instances des `CreateVaultResult` Objekts erstellen, das die Amazon Glacier-Antwort empfängt, wie im folgenden Java-Codeausschnitt dargestellt:

```
AmazonGlacierClient client = new AmazonGlacierClient(credentials);
client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

CreateVaultRequest request = new CreateVaultRequest()
    .withAccountId("-")
    .withVaultName(vaultName);
CreateVaultResult result = client.createVault(createVaultRequest);
```

Alle Low-Level-Beispiele in diesem Handbuch entsprechen diesem Muster.

> **Note**
>
> Im vorstehenden Codesegment wird beim Erstellen der Anfrage `AccountID` angegeben. Bei Verwendung von ist das AWS SDK für Java`AccountId` in der Anfrage jedoch optional, und daher wird dieser Wert nicht in allen einfachen Beispielen in diesem Handbuch festgelegt. Das `AccountId` ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder die AWS-Konto ID oder optional ein '-' angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche eingeben. Wenn Sie bei der Verwendung AWS SDK für Java die Konto-ID nicht angeben, setzt die Bibliothek die Konto-ID auf '-'.

Verwenden der High-Level-API

Um Ihre Anwendungsentwicklung weiter zu vereinfachen, AWS SDK für Java stellt die die `ArchiveTransferManager` Klasse bereit, die eine Abstraktion auf höherer Ebene für einige Methoden in der Low-Level-API implementiert. Sie stellt nützliche Methoden wie `upload` und `download` für die Archivoperationen bereit.

Beispielsweise wird im folgenden Java-Codeausschnitt die High-Level-Methode `upload` zum Hochladen eines Archivs verwendet.

```
String vaultName = "examplevault";
String archiveToUpload = "c:/folder/exampleArchive.zip";

ArchiveTransferManager atm = new ArchiveTransferManager(client, credentials);
String archiveId = atm.upload(vaultName, "Tax 2012 documents", new
 File(archiveToUpload)).getArchiveId();
```

Beachten Sie, dass alle Operationen, die Sie ausführen, für die AWS Region gelten, die Sie bei der Erstellung des Objekts angegeben haben. `ArchiveTransferManager` Wenn Sie keine Region angeben, wird die AWS AWS SDK für Java Region `us-east-1` als AWS Standardregion festgelegt.

Alle High-Level-Beispiele in diesem Handbuch entsprechen diesem Muster.

> **Note**
>
> Die High-Level-Klasse `ArchiveTransferManager` kann mit einer `AmazonGlacierClient`-Instance oder einer `AWSCredentials`-Instance konstruiert werden.

Ausführen von Java-Beispielen für Amazon Glacier mit Eclipse

Am einfachsten gelingt der Einstieg in die Java-Codebeispiele, indem die neueste AWS Toolkit for Eclipse installiert wird. Informationen zum Installieren oder Aktualisieren des neuesten Toolkits finden Sie unter http://aws.amazon.com/eclipse. Die folgenden Aufgaben führen Sie durch das Erstellen und Testen der Java-Codebeispiele aus diesem Abschnitt.

Allgemeiner Vorgang bei der Erstellung von Java-Codebeispielen

1	Erstellen Sie ein Standard-Anmeldeinformationsprofil für Ihre AWS Anmeldein formationen, wie im AWS SDK für Java Thema Bereitstellen von AWS Anmeldein formationen im Amazon SDK for Java beschrieben.
2	Erstellen Sie ein neues AWS Java-Projekt in Eclipse. Das Projekt ist mit dem AWS SDK für Java vorkonfiguriert.

3	Kopieren Sie den Code aus dem Abschnitt, den Sie gerade lesen, in Ihr Projekt.
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 den Code aus. Stellen Sie sicher, dass das Objekt mit der AWS-Manag ementkonsole erstellt wurde. Weitere Informationen zum AWS-Managementkonsole finden Sie unter http://aws.amazon.com/console/.

Festlegen des Endpunkts

Standardmäßig AWS SDK für Java verwendet der den Endpunkt`https://glacier.us-east-1.amazonaws.com`. Sie können den Endpunkt explizit einrichten, wie in den folgenden Java-Codeausschnitten gezeigt.

Der folgende Codeausschnitt zeigt, wie der Endpunkt in der Low-Level-API auf die Region „USA West (Oregon)“ (`us-west-2`) festgelegt wird.

Example

```
client = new AmazonGlacierClient(credentials);
client.setEndpoint("glacier.us-west-2.amazonaws.com");
```

Der folgende Codeausschnitt zeigt, wie der Endpunkt in der High-Level-API auf die Region „USA West (Oregon)“ festgelegt wird.

```
glacierClient = new AmazonGlacierClient(credentials);
sqsClient = new AmazonSQSClient(credentials);
snsClient = new AmazonSNSClient(credentials);

glacierClient.setEndpoint("glacier.us-west-2.amazonaws.com");
sqsClient.setEndpoint("sqs.us-west-2.amazonaws.com");
snsClient.setEndpoint("sns.us-west-2.amazonaws.com");

ArchiveTransferManager atm = new ArchiveTransferManager(glacierClient, sqsClient,
 snsClient);
```

Eine Liste der unterstützten AWS Regionen und Endpunkte finden Sie unterZugriff auf Amazon Glacier.

Verwenden von AWS SDK für .NET mit Amazon Glacier

Die AWS SDK für .NET API ist verfügbar in`AWSSDK.dll`. Informationen zum Herunterladen von finden Sie unter Beispielcodebibliotheken. AWS SDK für .NET Wie unter beschriebenVerwenden von AWS SDKs mit Amazon Glacier, AWS SDK für .NET bietet der sowohl die High-Level- als auch die APIs Low-Level-Version.

> **Note**
>
> Die Low-Level-API und die High-Level-API bieten Thread-sichere Clients für den Zugriff auf Amazon Glacier. Als bewährte Methode sollten Ihre Anwendungen einen Client erstellen und diesen zwischen den Threads wiederverwenden.

Themen

- Verwenden der Low-Level-API
- Verwenden der High-Level-API
- Ausführen von Codebeispielen
- Festlegen des Endpunkts

Verwenden der Low-Level-API

Die `AmazonGlacierClient` Low-Level-Klasse stellt alle Methoden bereit, die den zugrunde liegenden REST-Vorgängen von Amazon Glacier (Amazon Glacier) (API-Referenz für Amazon Glacier) zugeordnet sind. Wenn Sie eine dieser Methoden aufrufen, müssen Sie ein entsprechendes Anforderungsobjekt erstellen und ein Antwortobjekt bereitstellen, in dem die Methode eine Amazon Glacier-Antwort auf den Vorgang zurückgeben kann.

Beispielsweise stellt die `AmazonGlacierClient`-Klasse die `CreateVault`-Methode für die Erstellung eines Tresors zur Verfügung. Diese Methode lässt sich der zugrunde liegenden REST-Operation „Create Vault” zuordnen (siehe Create Vault (PUT vault)). Um diese Methode zu verwenden, müssen Sie Instances der `CreateVaultResponse` Klassen `CreateVaultRequest` und erstellen, um Anforderungsinformationen bereitzustellen und eine Amazon Glacier-Antwort zu erhalten, wie im folgenden C#-Codeausschnitt dargestellt:

```
AmazonGlacierClient client;
```

```
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USEast1);

CreateVaultRequest request = new CreateVaultRequest()
{
  AccountId = "-",
  VaultName = "*** Provide vault name ***"
};

CreateVaultResponse response = client.CreateVault(request);
```

Alle Low-Level-Beispiele in diesem Handbuch entsprechen diesem Muster.

> **Note**
>
> Im vorstehenden Codesegment wird beim Erstellen der Anfrage `AccountId` angegeben. Bei Verwendung von ist die AWS SDK für .NET Angabe `AccountId` in der Anfrage jedoch optional. Daher wird dieser Wert nicht in allen einfachen Beispielen in diesem Handbuch festgelegt. Das `AccountId` ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder die AWS-Konto ID oder optional ein '-' angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche eingeben. Wenn Sie bei der Verwendung AWS SDK für .NET die Konto-ID nicht angeben, setzt die Bibliothek die Konto-ID auf '-'.

Verwenden der High-Level-API

Um Ihre Anwendungsentwicklung weiter zu vereinfachen, AWS SDK für .NET stellt die die `ArchiveTransferManager` Klasse bereit, die eine Abstraktion auf höherer Ebene für einige Methoden in der Low-Level-API implementiert. Sie stellt nützliche Methoden wie `Upload` und `Download` für die Archivoperationen bereit.

Beispielsweise wird im folgenden C#-Codeausschnitt die High-Level-Methode `Upload` zum Hochladen eines Archivs verwendet.

```
string vaultName = "examplevault";
```

```
string archiveToUpload = "c:\folder\exampleArchive.zip";

var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USEast1);
string archiveId = manager.Upload(vaultName, "archive description",
 archiveToUpload).ArchiveId;
```

Beachten Sie, dass alle Operationen, die Sie ausführen, für die AWS Region gelten, die Sie bei der Erstellung des Objekts angegeben haben. `ArchiveTransferManager` Alle High-Level-Beispiele in diesem Handbuch entsprechen diesem Muster.

> **Note**
>
> Auch die `ArchiveTransferManager`-Klasse der High-Level-API braucht den Low-Level-Client `AmazonGlacierClient`, den Sie entweder explizit übergeben oder von `ArchiveTransferManager` erstellen lassen können.

Ausführen von Codebeispielen

Am einfachsten gelingt der Einstieg in die .NET-Codebeispiele durch die Installation von AWS SDK für .NET. Weitere Informationen finden Sie im Artikel zum Amazon-SDK für .NET.

Im folgenden Verfahren werden die Schritte beschrieben, mit denen Sie die in diesem Handbuch bereitgestellten Codebeispiele testen können.

Allgemeines Vorgehen zum Erstellen der .NET-Codebeispiele (mit Visual Studio)

1	Erstellen Sie ein Anmeldeinformationsprofil für Ihre AWS Anmeldeinformationen, wie im Thema Konfiguration von AWS Anmeldeinformationen für Amazon SDK for .NET beschrieben.
2	Erstellen Sie ein neues Visual-Studio-Projekt mit der Vorlage AWS Empty Project.
3	Ersetzen des Codes in der Projektdatei, `Program.cs` , durch den Code in dem Abschnitt, den Sie gerade lesen.
4	Führen Sie den Code aus. Stellen Sie sicher, dass das Objekt mit der AWS-Manag ementkonsole erstellt wurde. Weitere Informationen zu AWS-Managementkonsole finden Sie unter http://aws.amazon.com/console/.

Festlegen des Endpunkts

Standardmäßig AWS SDK für .NET legt der als Endpunkt die Region USA West (Oregon) fest (`https://glacier.us-west-2.amazonaws.com`). Sie können den Endpunkt auf andere AWS Regionen festlegen, wie in den folgenden C#-Snippets gezeigt.

Der folgende Codeausschnitt zeigt, wie der Endpunkt in der Low-Level-API auf die Region „USA West (Oregon)“ (`us-west-2`) festgelegt wird.

Example

```
AmazonGlacierClient client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);
```

Der folgende Codeausschnitt zeigt, wie der Endpunkt in der High-Level-API auf die Region „USA West (Oregon)“ festgelegt wird.

```
var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
```

Eine aktuelle Liste der unterstützten AWS Regionen und Endpunkte finden Sie unter. Zugriff auf Amazon Glacier

Codebeispiele für Amazon Glacier mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Amazon Glacier mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien anzeigen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie bestimmte Aufgaben ausführen, indem Sie mehrere Funktionen innerhalb eines Service aufrufen oder mit anderen AWS-Services kombinieren.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Grundlegende Beispiele für die Verwendung von Amazon Glacier AWS SDKs
  - Hallo Amazon Glacier
  - Aktionen für Amazon Glacier mit AWS SDKs
    - Verwendung AddTagsToVault mit einem AWS SDK oder CLI
    - Verwendung CreateVault mit einem AWS SDK oder CLI
    - Verwendung DeleteArchive mit einem AWS SDK oder CLI
    - Verwendung DeleteVault mit einem AWS SDK oder CLI
    - Verwendung DeleteVaultNotifications mit einem AWS SDK oder CLI
    - Verwendung DescribeJob mit einem AWS SDK oder CLI
    - Verwendung DescribeVault mit einem AWS SDK oder CLI
    - Verwendung GetJobOutput mit einem AWS SDK oder CLI
    - Verwendung GetVaultNotifications mit einem AWS SDK oder CLI
    - Verwendung InitiateJob mit einem AWS SDK oder CLI
    - Verwendung ListJobs mit einem AWS SDK oder CLI
    - Verwendung ListTagsForVault mit einem AWS SDK oder CLI
    - Verwendung ListVaults mit einem AWS SDK oder CLI
    - Verwendung SetVaultNotifications mit einem AWS SDK oder CLI

    - Verwendung UploadArchive mit einem AWS SDK oder CLI
    - Verwendung UploadMultipartPart mit einem AWS SDK oder CLI
- Szenarien für die Verwendung von Amazon Glacier AWS SDKs
    - Archivieren Sie eine Datei in Amazon Glacier, erhalten Sie Benachrichtigungen und initiieren Sie einen Job mithilfe eines AWS SDK
    - Rufen Sie Amazon Glacier-Archivinhalte ab und löschen Sie das Archiv mithilfe eines AWS SDK

Grundlegende Beispiele für die Verwendung von Amazon Glacier AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie die Grundlagen von Amazon Glacier mit verwenden können AWS SDKs.

Beispiele

- Hallo Amazon Glacier
- Aktionen für Amazon Glacier mit AWS SDKs
    - Verwendung AddTagsToVault mit einem AWS SDK oder CLI
    - Verwendung CreateVault mit einem AWS SDK oder CLI
    - Verwendung DeleteArchive mit einem AWS SDK oder CLI
    - Verwendung DeleteVault mit einem AWS SDK oder CLI
    - Verwendung DeleteVaultNotifications mit einem AWS SDK oder CLI
    - Verwendung DescribeJob mit einem AWS SDK oder CLI
    - Verwendung DescribeVault mit einem AWS SDK oder CLI
    - Verwendung GetJobOutput mit einem AWS SDK oder CLI
    - Verwendung GetVaultNotifications mit einem AWS SDK oder CLI
    - Verwendung InitiateJob mit einem AWS SDK oder CLI
    - Verwendung ListJobs mit einem AWS SDK oder CLI
    - Verwendung ListTagsForVault mit einem AWS SDK oder CLI
    - Verwendung ListVaults mit einem AWS SDK oder CLI
    - Verwendung SetVaultNotifications mit einem AWS SDK oder CLI
    - Verwendung UploadArchive mit einem AWS SDK oder CLI

- Verwendung UploadMultipartPart mit einem AWS SDK oder CLI

Hallo Amazon Glacier

Das folgende Codebeispiel zeigt die ersten Schritte mit Amazon Glacier.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
using Amazon.Glacier;
using Amazon.Glacier.Model;

namespace GlacierActions;

public static class HelloGlacier
{
    static async Task Main()
    {
        var glacierService = new AmazonGlacierClient();

        Console.WriteLine("Hello Amazon Glacier!");
        Console.WriteLine("Let's list your Glacier vaults:");

        // You can use await and any of the async methods to get a response.
        // Let's get the vaults using a paginator.
        var glacierVaultPaginator = glacierService.Paginators.ListVaults(
            new ListVaultsRequest { AccountId = "-" });

        await foreach (var vault in glacierVaultPaginator.VaultList)
        {
            Console.WriteLine($"{vault.CreationDate}:{vault.VaultName}, ARN:
{vault.VaultARN}");
        }
```

```
    }
}
```

- Einzelheiten zur API finden Sie ListVaultsin der AWS SDK für .NET API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Aktionen für Amazon Glacier mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie einzelne Amazon Glacier-Aktionen mit ausführen AWS SDKs. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden können.

Diese Auszüge rufen die Amazon Glacier-API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Sie können Aktionen im Kontext unter Szenarien für die Verwendung von Amazon Glacier AWS SDKs anzeigen.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der Amazon Glacier API-Referenz.

Beispiele

- Verwendung AddTagsToVault mit einem AWS SDK oder CLI
- Verwendung CreateVault mit einem AWS SDK oder CLI
- Verwendung DeleteArchive mit einem AWS SDK oder CLI
- Verwendung DeleteVault mit einem AWS SDK oder CLI
- Verwendung DeleteVaultNotifications mit einem AWS SDK oder CLI
- Verwendung DescribeJob mit einem AWS SDK oder CLI
- Verwendung DescribeVault mit einem AWS SDK oder CLI
- Verwendung GetJobOutput mit einem AWS SDK oder CLI
- Verwendung GetVaultNotifications mit einem AWS SDK oder CLI
- Verwendung InitiateJob mit einem AWS SDK oder CLI
- Verwendung ListJobs mit einem AWS SDK oder CLI
- Verwendung ListTagsForVault mit einem AWS SDK oder CLI

- Verwendung ListVaults mit einem AWS SDK oder CLI
- Verwendung SetVaultNotifications mit einem AWS SDK oder CLI
- Verwendung UploadArchive mit einem AWS SDK oder CLI
- Verwendung UploadMultipartPart mit einem AWS SDK oder CLI

Verwendung **`AddTagsToVault`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `AddTagsToVault` verwendet wird.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /// <summary>
   /// Add tags to the items in an Amazon S3 Glacier vault.
   /// </summary>
   /// <param name="vaultName">The name of the vault to add tags to.</param>
   /// <param name="key">The name of the object to tag.</param>
   /// <param name="value">The tag value to add.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> AddTagsToVaultAsync(string vaultName, string key,
string value)
   {
       var request = new AddTagsToVaultRequest
       {
           Tags = new Dictionary<string, string>
               {
                   { key, value },
               },
           AccountId = "-",
           VaultName = vaultName,
       };

       var response = await _glacierService.AddTagsToVaultAsync(request);
```

```
        return response.HttpStatusCode == HttpStatusCode.NoContent;
    }
```

- Einzelheiten zur API finden Sie AddTagsToVaultin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl fügt zwei Tags zu einem Tresor mit dem Namen `my-vault` hinzu:

```
aws glacier add-tags-to-vault --account-id - --vault-name my-vault --
tags id=1234,date=july2015
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie AddTagsToVaultin der AWS CLI Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateVault`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `CreateVault` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Create an Amazon S3 Glacier vault.
/// </summary>
/// <param name="vaultName">The name of the vault to create.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> CreateVaultAsync(string vaultName)
{
    var request = new CreateVaultRequest
    {
        // Setting the AccountId to "-" means that
        // the account associated with the current
        // account will be used.
        AccountId = "-",
        VaultName = vaultName,
    };

    var response = await _glacierService.CreateVaultAsync(request);

    Console.WriteLine($"Created {vaultName} at: {response.Location}");

    return response.HttpStatusCode == HttpStatusCode.Created;
}
```

- Einzelheiten zur API finden Sie CreateVaultin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl erstellt einen neuen Tresor mit dem Namen `my-vault`:

```
aws glacier create-vault --vault-name my-vault --account-id -
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie CreateVaultin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.CreateVaultRequest;
import software.amazon.awssdk.services.glacier.model.CreateVaultResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateVault {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <vaultName>

                Where:
                   vaultName - The name of the vault to create.
```

```
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String vaultName = args[0];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createGlacierVault(glacier, vaultName);
        glacier.close();
    }

    public static void createGlacierVault(GlacierClient glacier, String
 vaultName) {
        try {
            CreateVaultRequest vaultRequest = CreateVaultRequest.builder()
                    .vaultName(vaultName)
                    .build();

            CreateVaultResponse createVaultResult =
 glacier.createVault(vaultRequest);
            System.out.println("The URI of the new vault is " +
 createVaultResult.location());

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie CreateVaultin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie den Client.

```
const { GlacierClient } = require("@aws-sdk/client-glacier");
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const glacierClient = new GlacierClient({ region: REGION });
export { glacierClient };
```

Erstellen des Tresors.

```
// Load the SDK for JavaScript
import { CreateVaultCommand } from "@aws-sdk/client-glacier";
import { glacierClient } from "./libs/glacierClient.js";

// Set the parameters
const vaultname = "VAULT_NAME"; // VAULT_NAME
const params = { vaultName: vaultname };

const run = async () => {
  try {
    const data = await glacierClient.send(new CreateVaultCommand(params));
    console.log("Success, vault created!");
    return data; // For unit tests.
  } catch (err) {
    console.log("Error");
  }
};
run();
```

- Weitere Informationen finden Sie im AWS SDK für JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie CreateVaultin der AWS SDK für JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
// Load the SDK for JavaScript
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create a new service object
var glacier = new AWS.Glacier({ apiVersion: "2012-06-01" });
// Call Glacier to create the vault
glacier.createVault({ vaultName: "YOUR_VAULT_NAME" }, function (err) {
  if (!err) {
    console.log("Created vault!");
  }
});
```

- Weitere Informationen finden Sie im AWS SDK für JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie CreateVaultin der AWS SDK für JavaScript API-Referenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Erstellt einen neuen Tresor für das Benutzerkonto. Da für den AccountId Parameter - kein Wert angegeben wurde, verwenden die Cmdlets den Standardwert „-“, der das aktuelle Konto angibt.

```
New-GLCVault -VaultName myvault
```

Ausgabe:

```
/01234567812/vaults/myvault
```

- Einzelheiten zur API finden Sie unter CreateVault AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Erstellt einen neuen Tresor für das Benutzerkonto. Da für den AccountId Parameter - kein Wert angegeben wurde, verwenden die Cmdlets den Standardwert „-“, der das aktuelle Konto angibt.

```
New-GLCVault -VaultName myvault
```

Ausgabe:

```
/01234567812/vaults/myvault
```

- Einzelheiten zur API finden Sie unter CreateVault AWS -Tools für PowerShellCmdlet-Referenz (V5).

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource
```

```
    def create_vault(self, vault_name):
        """
        Creates a vault.

        :param vault_name: The name to give the vault.
        :return: The newly created vault.
        """
        try:
            vault = self.glacier_resource.create_vault(vaultName=vault_name)
            logger.info("Created vault %s.", vault_name)
        except ClientError:
            logger.exception("Couldn't create vault %s.", vault_name)
            raise
        else:
            return vault
```

- Einzelheiten zur API finden Sie CreateVaultin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteArchive`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteArchive` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Abrufen von Archivinhalten und Löschen des Archivs

CLI

AWS CLI

So löschen Sie ein Archiv aus einem Tresor

Im folgenden Beispiel für `delete-archive` wird das angegebene Archiv aus `example_vault` entfernt.

```
aws glacier delete-archive \
    --account-id 111122223333  \
    --vault-name example_vault \
    --archive-id Sc0u9ZP8yaWkmh-XGlIvAVprtLhaLCGnNwNl5I5x9HqPIkX5mjc0DrId3Ln-
Gi_k2HzmlIDZUz117KSdVMdMXLuFWi9PJUitxWO73edQ43eTlMWkH0pd9zVSAuV_XXZBVhKhyGhJ7w
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- Einzelheiten zur API finden Sie DeleteArchivein der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.DeleteArchiveRequest;
import software.amazon.awssdk.services.glacier.model.GlacierException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteArchive {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <vaultName> <accountId> <archiveId>

                Where:
```

```
                    vaultName - The name of the vault that contains the archive to
 delete.
                    accountId - The account ID value.
                    archiveId - The archive ID value.
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String vaultName = args[0];
        String accountId = args[1];
        String archiveId = args[2];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deleteGlacierArchive(glacier, vaultName, accountId, archiveId);
        glacier.close();
    }

    public static void deleteGlacierArchive(GlacierClient glacier, String
 vaultName, String accountId,
            String archiveId) {
        try {
            DeleteArchiveRequest delArcRequest = DeleteArchiveRequest.builder()
                    .vaultName(vaultName)
                    .accountId(accountId)
                    .archiveId(archiveId)
                    .build();

            glacier.deleteArchive(delArcRequest);
            System.out.println("The archive was deleted.");

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie DeleteArchivein der AWS SDK for Java 2.x API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def delete_archive(archive):
        """
        Deletes an archive from a vault.

        :param archive: The archive to delete.
        """
        try:
            archive.delete()
            logger.info(
                "Deleted archive %s from vault %s.", archive.id,
 archive.vault_name
            )
        except ClientError:
            logger.exception("Couldn't delete archive %s.", archive.id)
            raise
```

- Einzelheiten zur API finden Sie DeleteArchivein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteVault`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteVault` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Abrufen von Archivinhalten und Löschen des Archivs

CLI

AWS CLI

Der folgende Befehl löscht einen Tresor mit dem Namen `my-vault`:

```
aws glacier delete-vault --vault-name my-vault --account-id -
```

Dieser Befehl erzeugt keine Ausgabe. Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie DeleteVaultin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.DeleteVaultRequest;
```

```
import software.amazon.awssdk.services.glacier.model.GlacierException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteVault {
    public static void main(String[] args) {

        final String usage = """

                Usage:    <vaultName>

                Where:
                   vaultName - The name of the vault to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String vaultName = args[0];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deleteGlacierVault(glacier, vaultName);
        glacier.close();
    }

    public static void deleteGlacierVault(GlacierClient glacier, String
 vaultName) {
        try {
            DeleteVaultRequest delVaultRequest = DeleteVaultRequest.builder()
                    .vaultName(vaultName)
                    .build();

            glacier.deleteVault(delVaultRequest);
```

```
            System.out.println("The vault was deleted!");

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie DeleteVaultin der AWS SDK for Java 2.x API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def delete_vault(vault):
        """
        Deletes a vault.

        :param vault: The vault to delete.
        """
        try:
            vault.delete()
            logger.info("Deleted vault %s.", vault.name)
```

```
        except ClientError:
            logger.exception("Couldn't delete vault %s.", vault.name)
            raise
```

- Einzelheiten zur API finden Sie DeleteVaultin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteVaultNotifications`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteVaultNotifications` verwendet wird.

CLI

AWS CLI

So entfernen Sie die SNS-Benachrichtigungen für einen Tresor

Im folgenden Beispiel für `delete-vault-notifications` werden Benachrichtigungen, die von Amazon Simple Notification Service (Amazon SNS) gesendet wurden, für den angegebenen Tresor entfernt.

```
aws glacier delete-vault-notifications \
    --account-id 111122223333 \
    --vault-name example_vault
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

- Einzelheiten zur API finden Sie DeleteVaultNotificationsin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def stop_notifications(notification):
        """
        Stops notifications to the configured Amazon SNS topic.

        :param notification: The notification configuration to remove.
        """
        try:
            notification.delete()
            logger.info("Notifications stopped.")
        except ClientError:
            logger.exception("Couldn't stop notifications.")
            raise
```

- Einzelheiten zur API finden Sie DeleteVaultNotificationsin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DescribeJob`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeJob` verwendet wird.

CLI

AWS CLI

Der folgende Befehl ruft Informationen über einen Auftrag zum Abrufen von Inventar in einem Tresor mit dem Namen `my-vault` ab:

```
aws glacier describe-job --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW
```

Ausgabe:

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
    "Completed": false,
    "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
    "Action": "InventoryRetrieval",
    "CreationDate": "2015-07-17T20:23:41.616Z",
    "StatusCode": "InProgress"
}
```

Die Auftrags-ID finden Sie in der Ausgabe von `aws glacier initiate-job` und `aws glacier list-jobs`. Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie DescribeJobin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Gibt Details des angegebenen Auftrags zurück. Wenn der Auftrag erfolgreich abgeschlossen wurde, kann das Cmdlet GCJob Read-Output verwendet werden, um den Inhalt des Auftrags (ein Archiv oder eine Inventarliste) in das lokale Dateisystem abzurufen.

```
Get-GLCJob -VaultName myvault -JobId "op1x...JSbthM"
```

Ausgabe:

```
Action                       : ArchiveRetrieval
ArchiveId                    : o9O9j...X-TpIhQJw
ArchiveSHA256TreeHash        : 79f3ea754c02f58...dc57bf4395b
ArchiveSizeInBytes           : 38034480
Completed                    : False
CompletionDate               : 1/1/0001 12:00:00 AM
CreationDate                 : 12/13/2018 11:00:14 AM
InventoryRetrievalParameters :
InventorySizeInBytes         : 0
JobDescription               :
JobId                        : op1x...JSbthM
JobOutputPath                :
OutputLocation               :
RetrievalByteRange           : 0-38034479
SelectParameters             :
SHA256TreeHash               : 79f3ea754c02f58...dc57bf4395b
SNSTopic                     :
StatusCode                   : InProgress
StatusMessage                :
Tier                         : Standard
VaultARN                     : arn:aws:glacier:us-west-2:012345678912:vaults/test
```

- Einzelheiten zur API finden Sie unter DescribeJob AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Gibt Details des angegebenen Auftrags zurück. Wenn der Auftrag erfolgreich abgeschlossen wurde, kann das Cmdlet GCJob Read-Output verwendet werden, um den Inhalt des Auftrags (ein Archiv oder eine Inventarliste) in das lokale Dateisystem abzurufen.

```
Get-GLCJob -VaultName myvault -JobId "op1x...JSbthM"
```

Ausgabe:

```
Action                       : ArchiveRetrieval
ArchiveId                    : o9O9j...X-TpIhQJw
ArchiveSHA256TreeHash        : 79f3ea754c02f58...dc57bf4395b
ArchiveSizeInBytes           : 38034480
Completed                    : False
CompletionDate               : 1/1/0001 12:00:00 AM
CreationDate                 : 12/13/2018 11:00:14 AM
InventoryRetrievalParameters :
InventorySizeInBytes         : 0
JobDescription               :
JobId                        : op1x...JSbthM
JobOutputPath                :
OutputLocation               :
RetrievalByteRange           : 0-38034479
SelectParameters             :
SHA256TreeHash               : 79f3ea754c02f58...dc57bf4395b
SNSTopic                     :
StatusCode                   : InProgress
StatusMessage                :
Tier                         : Standard
VaultARN                     : arn:aws:glacier:us-west-2:012345678912:vaults/test
```

- Einzelheiten zur API finden Sie unter DescribeJob AWS -Tools für PowerShellCmdlet-Referenz (V5).

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
```

```
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def get_job_status(job):
        """
        Gets the status of a job.

        :param job: The job to query.
        :return: The current status of the job.
        """
        try:
            job.load()
            logger.info(
                "Job %s is performing action %s and has status %s.",
                job.id,
                job.action,
                job.status_code,
            )
        except ClientError:
            logger.exception("Couldn't get status for job %s.", job.id)
            raise
        else:
            return job.status_code
```

- Einzelheiten zur API finden Sie DescribeJobin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DescribeVault`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeVault` verwendet wird.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Describe an Amazon S3 Glacier vault.
    /// </summary>
    /// <param name="vaultName">The name of the vault to describe.</param>
    /// <returns>The Amazon Resource Name (ARN) of the vault.</returns>
    public async Task<string> DescribeVaultAsync(string vaultName)
    {
        var request = new DescribeVaultRequest
        {
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.DescribeVaultAsync(request);

        // Display the information about the vault.
        Console.WriteLine($"{response.VaultName}\tARN: {response.VaultARN}");
        Console.WriteLine($"Created on: {response.CreationDate}\tNumber
of Archives: {response.NumberOfArchives}\tSize (in bytes):
{response.SizeInBytes}");
        if (response.LastInventoryDate != DateTime.MinValue)
        {
            Console.WriteLine($"Last inventory: {response.LastInventoryDate}");
        }

        return response.VaultARN;
    }
```

- Einzelheiten zur API finden Sie DescribeVaultin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl ruft Daten über einen Tresor mit dem Namen `my-vault` ab:

```
aws glacier describe-vault --vault-name my-vault --account-id -
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie DescribeVaultin der AWS CLI Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetJobOutput`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetJobOutput` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Abrufen von Archivinhalten und Löschen des Archivs

CLI

AWS CLI

Der folgende Befehl speichert die Ausgabe eines Tresor-Inventarauftrags in einer Datei im aktuellen Verzeichnis mit dem Namen `output.json`:

```
aws glacier get-job-output --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61ds04CnMW output.json
```

Die `job-id` ist in der Ausgabe von `aws glacier list-jobs` verfügbar. Beachten Sie, dass der Name der Ausgabedatei ein Positionsargument ist, dem kein Optionsname

vorangestellt ist. Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

Ausgabe:

```
{
    "status": 200,
    "acceptRanges": "bytes",
    "contentType": "application/json"
}
```

`output.json`:

```
{"VaultARN":"arn:aws:glacier:us-west-2:0123456789012:vaults/
my-vault","InventoryDate":"2015-04-07T00:26:18Z","ArchiveList":
[{"ArchiveId":"kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw","ArchiveDescription":"multipart
 upload
 test","CreationDate":"2015-04-06T22:24:34Z","Size":3145728,"SHA256TreeHash":"9628195fcdb
```

- Einzelheiten zur API finden Sie GetJobOutputin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Lädt den Archivinhalt herunter, dessen Abruf im angegebenen Auftrag geplant war, und speichert den Inhalt in einer Datei auf der Festplatte. Der Download überprüft für Sie die Prüfsumme, sofern eine verfügbar ist. Auf Wunsch kann die gesamte Antwort einschließlich der Prüfsumme durch Angabe von **`-Select '*'`** zurückgegeben werden.

```
Read-GLCJobOutput -VaultName myvault -JobId "HSWjArc...Zq2XLiW" -FilePath "c:
\temp\blue.bin"
```

- Einzelheiten zur API finden Sie unter GetJobOutput AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Lädt den Archivinhalt herunter, dessen Abruf im angegebenen Auftrag geplant war, und speichert den Inhalt in einer Datei auf der Festplatte. Der Download überprüft für Sie die Prüfsumme, sofern eine verfügbar ist. Auf Wunsch kann die gesamte Antwort einschließlich der Prüfsumme durch Angabe von **`-Select '*'`** zurückgegeben werden.

```
Read-GLCJobOutput -VaultName myvault -JobId "HSWjArc...Zq2XLiW" -FilePath "c:
\temp\blue.bin"
```

- Einzelheiten zur API finden Sie unter GetJobOutput AWS -Tools für PowerShellCmdlet-Referenz (V5).

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def get_job_output(job):
        """
        Gets the output of a job, such as a vault inventory or the contents of an
        archive.

        :param job: The job to get output from.
        :return: The job output, in bytes.
```

```
        """
        try:
            response = job.get_output()
            out_bytes = response["body"].read()
            logger.info("Read %s bytes from job %s.", len(out_bytes), job.id)
            if "archiveDescription" in response:
                logger.info(
                    "These bytes are described as '%s'",
response["archiveDescription"]
                )
        except ClientError:
            logger.exception("Couldn't get output for job %s.", job.id)
            raise
        else:
            return out_bytes
```

- Einzelheiten zur API finden Sie GetJobOutputin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetVaultNotifications`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetVaultNotifications` verwendet wird.

CLI

AWS CLI

Der folgende Befehl ruft eine Beschreibung der Benachrichtigungskonfiguration für einen Tresor mit dem Namen `my-vault` ab:

```
aws glacier get-vault-notifications --account-id - --vault-name my-vault
```

Ausgabe:

```
{
    "vaultNotificationConfig": {
```

```
        "Events": [
            "InventoryRetrievalCompleted",
            "ArchiveRetrievalCompleted"
        ],
        "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault"
    }
}
```

Wenn keine Benachrichtigungen für den Tresor konfiguriert wurden, wird ein Fehler zurückgegeben. Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie GetVaultNotificationsin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def get_notification(vault):
        """
        Gets the currently notification configuration for a vault.

        :param vault: The vault to query.
```

```
        :return: The notification configuration for the specified vault.
        """
        try:
            notification = vault.Notification()
            logger.info(
                "Vault %s notifies %s on %s events.",
                vault.name,
                notification.sns_topic,
                notification.events,
            )
        except ClientError:
            logger.exception("Couldn't get notification data for %s.",
vault.name)
            raise
        else:
            return notification
```

- Einzelheiten zur API finden Sie GetVaultNotificationsin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`InitiateJob`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `InitiateJob` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Rufen Sie ein Archiv aus einem Tresor ab. In diesem Beispiel wird die ArchiveTransferManager Klasse verwendet. Einzelheiten zur API finden Sie unter ArchiveTransferManager.

```
    /// <summary>
    /// Download an archive from an Amazon S3 Glacier vault using the Archive
    /// Transfer Manager.
    /// </summary>
    /// <param name="vaultName">The name of the vault containing the object.</
param>
    /// <param name="archiveId">The Id of the archive to download.</param>
    /// <param name="localFilePath">The local directory where the file will
    /// be stored after download.</param>
    /// <returns>Async Task.</returns>
    public async Task<bool> DownloadArchiveWithArchiveManagerAsync(string
 vaultName, string archiveId, string localFilePath)
    {
        try
        {
            var manager = new ArchiveTransferManager(_glacierService);

            var options = new DownloadOptions
            {
                StreamTransferProgress = Progress!,
            };

            // Download an archive.
            Console.WriteLine("Initiating the archive retrieval job and then
 polling SQS queue for the archive to be available.");
            Console.WriteLine("When the archive is available, downloading will
 begin.");
```

```
            await manager.DownloadAsync(vaultName, archiveId, localFilePath,
options);

            return true;
        }
        catch (AmazonGlacierException ex)
        {
            Console.WriteLine(ex.Message);
            return false;
        }
    }

    /// <summary>
    /// Event handler to track the progress of the Archive Transfer Manager.
    /// </summary>
    /// <param name="sender">The object that raised the event.</param>
    /// <param name="args">The argument values from the object that raised the
    /// event.</param>
    static void Progress(object sender, StreamTransferProgressArgs args)
    {
        if (args.PercentDone != _currentPercentage)
        {
            _currentPercentage = args.PercentDone;
            Console.WriteLine($"Downloaded {_currentPercentage}%");
        }
    }
```

- Einzelheiten zur API finden Sie InitiateJobin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Mit dem folgenden Befehl wird ein Auftrag zum Abrufen eines Bestands des Tresors `my-vault` eingeleitet:

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-parameters
 '{"Type": "inventory-retrieval"}'
```

Ausgabe:

```
{
    "location": "/0123456789012/vaults/my-vault/jobs/
zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
    "jobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW"
}
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

Mit dem folgenden Befehl wird ein Auftrag zum Abrufen eines Archivs des Tresors `my-vault` eingeleitet:

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-
parameters file://job-archive-retrieval.json
```

`job-archive-retrieval.json` ist eine JSON-Datei im lokalen Ordner, die den Auftragstyp, die Archiv-ID und einige optionale Parameter angibt:

```
{
  "Type": "archive-retrieval",
  "ArchiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
  "Description": "Retrieve archive on 2015-07-17",
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-topic"
}
```

Archive IDs sind in der Ausgabe von `aws glacier upload-archive` und verfügbar`aws glacier get-job-output`.

Ausgabe:

```
{
    "location": "/011685312445/vaults/mwunderl/jobs/l7IL5-
EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
    "jobId": "l7IL5-EkXy2O5uLYaFdAYOiEY9Ws95fClzIbk-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav"
}
```

Details zum Format der Auftragsparameter finden Sie unter „Einleiten eines Auftrags“ in der API-Referenz zu Amazon Glacier.

- Einzelheiten zur API finden Sie InitiateJobin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Rufen Sie eine Tresorinventur ab.

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.JobParameters;
import software.amazon.awssdk.services.glacier.model.InitiateJobResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import software.amazon.awssdk.services.glacier.model.InitiateJobRequest;
import software.amazon.awssdk.services.glacier.model.DescribeJobRequest;
import software.amazon.awssdk.services.glacier.model.DescribeJobResponse;
import software.amazon.awssdk.services.glacier.model.GetJobOutputRequest;
import software.amazon.awssdk.services.glacier.model.GetJobOutputResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
```

```
public class ArchiveDownload {
    public static void main(String[] args) {

        final String usage = """

                Usage:    <vaultName> <accountId> <path>

                Where:
                   vaultName - The name of the vault.
                   accountId - The account ID value.
                   path - The path where the file is written to.
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String vaultName = args[0];
        String accountId = args[1];
        String path = args[2];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String jobNum = createJob(glacier, vaultName, accountId);
        checkJob(glacier, jobNum, vaultName, accountId, path);
        glacier.close();
    }

    public static String createJob(GlacierClient glacier, String vaultName,
 String accountId) {
        try {
            JobParameters job = JobParameters.builder()
                    .type("inventory-retrieval")
                    .build();

            InitiateJobRequest initJob = InitiateJobRequest.builder()
                    .jobParameters(job)
                    .accountId(accountId)
                    .vaultName(vaultName)
                    .build();

            InitiateJobResponse response = glacier.initiateJob(initJob);
```

```
            System.out.println("The job ID is: " + response.jobId());
            System.out.println("The relative URI path of the job is: " +
response.location());
            return response.jobId();

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);

        }
        return "";
    }

    // Poll S3 Glacier = Polling a Job may take 4-6 hours according to the
    // Documentation.
    public static void checkJob(GlacierClient glacier, String jobId, String name,
String account, String path) {
        try {
            boolean finished = false;
            String jobStatus;
            int yy = 0;

            while (!finished) {
                DescribeJobRequest jobRequest = DescribeJobRequest.builder()
                        .jobId(jobId)
                        .accountId(account)
                        .vaultName(name)
                        .build();

                DescribeJobResponse response = glacier.describeJob(jobRequest);
                jobStatus = response.statusCodeAsString();

                if (jobStatus.compareTo("Succeeded") == 0)
                    finished = true;
                else {
                    System.out.println(yy + " status is: " + jobStatus);
                    Thread.sleep(1000);
                }
                yy++;
            }

            System.out.println("Job has Succeeded");
            GetJobOutputRequest jobOutputRequest = GetJobOutputRequest.builder()
                    .jobId(jobId)
```

```
                    .vaultName(name)
                    .accountId(account)
                    .build();

            ResponseBytes<GetJobOutputResponse> objectBytes =
 glacier.getJobOutputAsBytes(jobOutputRequest);
            // Write the data to a local file.
            byte[] data = objectBytes.asByteArray();
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from a Glacier
 vault");
            os.close();

        } catch (GlacierException | InterruptedException | IOException e) {
            System.out.println(e.getMessage());
            System.exit(1);

        }
    }
}
```

- Einzelheiten zur API finden Sie InitiateJobin der AWS SDK for Java 2.x API-Referenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Startet einen Auftrag zum Abrufen eines Archivs aus dem angegebenen Tresor, dessen Eigentümer der Benutzer ist. Der Status des Jobs kann mit dem GLCJob Cmdlet Get- überprüft werden. Wenn der Auftrag erfolgreich abgeschlossen wurde, kann das Cmdlet GCJob Read-Output verwendet werden, um den Inhalt des Archivs in das lokale Dateisystem abzurufen.

```
Start-GLCJob -VaultName myvault -JobType "archive-retrieval" -JobDescription
 "archive retrieval" -ArchiveId "o9O9j...TX-TpIhQJw"
```

Ausgabe:

```
JobId            JobOutputPath Location
```

```
-----            ------------ --------
op1x...JSbthM                 /012345678912/vaults/test/jobs/
op1xe...I4HqCHkSJSbthM
```

- Einzelheiten zur API finden Sie unter InitiateJob AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Startet einen Auftrag zum Abrufen eines Archivs aus dem angegebenen Tresor, dessen Eigentümer der Benutzer ist. Der Status des Jobs kann mit dem GLCJob Cmdlet Get- überprüft werden. Wenn der Auftrag erfolgreich abgeschlossen wurde, kann das Cmdlet GCJob Read-Output verwendet werden, um den Inhalt des Archivs in das lokale Dateisystem abzurufen.

```
Start-GLCJob -VaultName myvault -JobType "archive-retrieval" -JobDescription
 "archive retrieval" -ArchiveId "o9O9j...TX-TpIhQJw"
```

Ausgabe:

```
JobId            JobOutputPath Location
-----            ------------ --------
op1x...JSbthM                 /012345678912/vaults/test/jobs/
op1xe...I4HqCHkSJSbthM
```

- Einzelheiten zur API finden Sie unter InitiateJob AWS -Tools für PowerShellCmdlet-Referenz (V5).

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Rufen Sie eine Tresorinventur ab.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def initiate_inventory_retrieval(vault):
        """
        Initiates an inventory retrieval job. The inventory describes the
 contents
        of the vault. Standard retrievals typically complete within 3–5 hours.
        When the job completes, you can get the inventory by calling
 get_output().

        :param vault: The vault to inventory.
        :return: The inventory retrieval job.
        """
        try:
            job = vault.initiate_inventory_retrieval()
            logger.info("Started %s job with ID %s.", job.action, job.id)
        except ClientError:
            logger.exception("Couldn't start job on vault %s.", vault.name)
            raise
        else:
            return job
```

Rufen Sie ein Archiv aus einem Tresor ab.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource
```

```
    @staticmethod
    def initiate_archive_retrieval(archive):
        """
        Initiates an archive retrieval job. Standard retrievals typically
complete
        within 3–5 hours. When the job completes, you can get the archive
contents
        by calling get_output().

        :param archive: The archive to retrieve.
        :return: The archive retrieval job.
        """
        try:
            job = archive.initiate_archive_retrieval()
            logger.info("Started %s job with ID %s.", job.action, job.id)
        except ClientError:
            logger.exception("Couldn't start job on archive %s.", archive.id)
            raise
        else:
            return job
```

- Einzelheiten zur API finden Sie InitiateJobin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListJobs`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `ListJobs` verwendet wird.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen
- Abrufen von Archivinhalten und Löschen des Archivs

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// List Amazon S3 Glacier jobs.
    /// </summary>
    /// <param name="vaultName">The name of the vault to list jobs for.</param>
    /// <returns>A list of Amazon S3 Glacier jobs.</returns>
    public async Task<List<GlacierJobDescription>> ListJobsAsync(string
vaultName)
    {
        var request = new ListJobsRequest
        {
            // Using a hyphen "-" for the Account Id will
            // cause the SDK to use the Account Id associated
            // with the current account.
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.ListJobsAsync(request);

        return response.JobList;
    }
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl listet laufende und kürzlich abgeschlossene Aufträge für einen Tresor mit dem Namen `my-vault` auf:

```
aws glacier list-jobs --account-id - --vault-name my-vault
```

Ausgabe:

```
{
    "JobList": [
        {
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-
vault",
            "RetrievalByteRange": "0-3145727",
            "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
            "Completed": false,
            "SHA256TreeHash":
 "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
            "JobId": "l7IL5-EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
            "ArchiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--
zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
            "JobDescription": "Retrieve archive on 2015-07-17",
            "ArchiveSizeInBytes": 3145728,
            "Action": "ArchiveRetrieval",
            "ArchiveSHA256TreeHash":
 "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
            "CreationDate": "2015-07-17T21:16:13.840Z",
            "StatusCode": "InProgress"
        },
        {
            "InventoryRetrievalParameters": {
                "Format": "JSON"
            },
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-
vault",
            "Completed": false,
            "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
```

```
            "Action": "InventoryRetrieval",
            "CreationDate": "2015-07-17T20:23:41.616Z",
            "StatusCode": ""InProgress""
        }
    ]
}
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie ListJobsin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def list_jobs(vault, job_type):
        """
        Lists jobs by type for the specified vault.

        :param vault: The vault to query.
        :param job_type: The type of job to list.
        :return: The list of jobs of the requested type.
        """
        job_list = []
```

```
        try:
            if job_type == "all":
                jobs = vault.jobs.all()
            elif job_type == "in_progress":
                jobs = vault.jobs_in_progress.all()
            elif job_type == "completed":
                jobs = vault.completed_jobs.all()
            elif job_type == "succeeded":
                jobs = vault.succeeded_jobs.all()
            elif job_type == "failed":
                jobs = vault.failed_jobs.all()
            else:
                jobs = []
                logger.warning("%s isn't a type of job I can get.", job_type)
            for job in jobs:
                job_list.append(job)
                logger.info("Got %s %s job %s.", job_type, job.action, job.id)
        except ClientError:
            logger.exception("Couldn't get %s jobs from %s.", job_type,
vault.name)
            raise
        else:
            return job_list
```

- Einzelheiten zur API finden Sie ListJobsin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListTagsForVault`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `ListTagsForVault` verwendet wird.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// List tags for an Amazon S3 Glacier vault.
    /// </summary>
    /// <param name="vaultName">The name of the vault to list tags for.</param>
    /// <returns>A dictionary listing the tags attached to each object in the
    /// vault and its tags.</returns>
    public async Task<Dictionary<string, string>> ListTagsForVaultAsync(string
vaultName)
    {
        var request = new ListTagsForVaultRequest
        {
            // Using a hyphen "-" for the Account Id will
            // cause the SDK to use the Account Id associated
            // with the default user.
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.ListTagsForVaultAsync(request);

        return response.Tags;
    }
```

- Einzelheiten zur API finden Sie ListTagsForVaultin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl listet die Tags auf, die auf einen Tresor mit dem Namen `my-vault` angewendet wurden:

```
aws glacier list-tags-for-vault --account-id - --vault-name my-vault
```

Ausgabe:

```
{
    "Tags": {
        "date": "july2015",
        "id": "1234"
    }
}
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie ListTagsForVaultin der AWS CLI Befehlsreferenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListVaults`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `ListVaults` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// List the Amazon S3 Glacier vaults associated with the current account.
/// </summary>
/// <returns>A list containing information about each vault.</returns>
public async Task<List<DescribeVaultOutput>> ListVaultsAsync()
{
    var glacierVaultPaginator = _glacierService.Paginators.ListVaults(
        new ListVaultsRequest { AccountId = "-" });
    var vaultList = new List<DescribeVaultOutput>();

    await foreach (var vault in glacierVaultPaginator.VaultList)
    {
        vaultList.Add(vault);
    }

    return vaultList;
}
```

- Einzelheiten zur API finden Sie ListVaultsin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl listet die Tresore im Standardkonto und der Standardregion auf:

```
aws glacier list-vaults --account-id -
```

Ausgabe:

```
{
    "VaultList": [
        {
            "SizeInBytes": 3178496,
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-
vault",
            "LastInventoryDate": "2015-04-07T00:26:19.028Z",
            "VaultName": "my-vault",
            "NumberOfArchives": 1,
            "CreationDate": "2015-04-06T21:23:45.708Z"
        }
    ]
}
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie ListVaultsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.model.ListVaultsRequest;
import software.amazon.awssdk.services.glacier.model.ListVaultsResponse;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.DescribeVaultOutput;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListVaults {
    public static void main(String[] args) {
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllVault(glacier);
        glacier.close();
    }

    public static void listAllVault(GlacierClient glacier) {
        boolean listComplete = false;
        String newMarker = null;
        int totalVaults = 0;
        System.out.println("Your Amazon Glacier vaults:");
        try {
            while (!listComplete) {
                ListVaultsResponse response = null;
                if (newMarker != null) {
                    ListVaultsRequest request = ListVaultsRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = glacier.listVaults(request);
                } else {
                    ListVaultsRequest request = ListVaultsRequest.builder()
                            .build();
                    response = glacier.listVaults(request);
                }

                List<DescribeVaultOutput> vaultList = response.vaultList();
                for (DescribeVaultOutput v : vaultList) {
                    totalVaults += 1;
                    System.out.println("* " + v.vaultName());
                }

                // Check for further results.
                newMarker = response.marker();
                if (newMarker == null) {
```

```
                    listComplete = true;
                }
            }

            if (totalVaults == 0) {
                System.out.println("No vaults found.");
            }

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListVaultsin der AWS SDK for Java 2.x API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    def list_vaults(self):
        """
        Lists vaults for the current account.
        """
```

```
        try:
            for vault in self.glacier_resource.vaults.all():
                logger.info("Got vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't list vaults.")
            raise
```

- Einzelheiten zur API finden Sie ListVaultsin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`SetVaultNotifications`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `SetVaultNotifications` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen

CLI

AWS CLI

Der folgende Befehl konfiguriert SNS-Benachrichtigungen für einen Tresor mit dem Namen `my-vault`:

```
aws glacier set-vault-notifications --account-id - --vault-name my-vault --vault-notification-config file://notificationconfig.json
```

`notificationconfig.json` ist eine JSON-Datei im aktuellen Ordner, die ein SNS-Thema und die zu veröffentlichenden Ereignisse angibt:

```
{
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
```

```
  "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

- Einzelheiten zur API finden Sie SetVaultNotificationsin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    def set_notifications(self, vault, sns_topic_arn):
        """
        Sets an Amazon Simple Notification Service (Amazon SNS) topic as a target
        for notifications. Amazon S3 Glacier publishes messages to this topic for
        the configured list of events.

        :param vault: The vault to set up to publish notifications.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of the topic that
                              receives notifications.
        :return: Data about the new notification configuration.
        """
        try:
            notification = self.glacier_resource.Notification("-", vault.name)
            notification.set(
```

```
                vaultNotificationConfig={
                    "SNSTopic": sns_topic_arn,
                    "Events": [
                        "ArchiveRetrievalCompleted",
                        "InventoryRetrievalCompleted",
                    ],
                }
            )
            logger.info(
                "Notifications will be sent to %s for events %s from %s.",
                notification.sns_topic,
                notification.events,
                notification.vault_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't set notifications to %s on %s.", sns_topic_arn,
vault.name
            )
            raise
        else:
            return notification
```

- Einzelheiten zur API finden Sie SetVaultNotificationsin AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UploadArchive`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `UploadArchive` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Archivieren einer Datei, Erhalten von Benachrichtigungen und Initiieren von Aufträgen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /// <summary>
   /// Upload an object to an Amazon S3 Glacier vault.
   /// </summary>
   /// <param name="vaultName">The name of the Amazon S3 Glacier vault to upload
   /// the archive to.</param>
   /// <param name="archiveFilePath">The file path of the archive to upload to
the vault.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<string> UploadArchiveWithArchiveManager(string vaultName,
string archiveFilePath)
   {
       try
       {
           var manager = new ArchiveTransferManager(_glacierService);

           // Upload an archive.
           var response = await manager.UploadAsync(vaultName, "upload archive
test", archiveFilePath);
           return response.ArchiveId;
       }
       catch (AmazonGlacierException ex)
       {
           Console.WriteLine(ex.Message);
           return string.Empty;
       }
   }
```

- Einzelheiten zur API finden Sie UploadArchivein der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

Der folgende Befehl lädt ein Archiv im aktuellen Ordner mit dem Namen `archive.zip` in einen Tresor mit dem Namen `my-vault` hoch:

```
aws glacier upload-archive --account-id - --vault-name my-vault --
body archive.zip
```

Ausgabe:

```
{
    "archiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--
zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
    "checksum":
 "969fb39823836d81f0cc028195fcdbcbbe76cdde932d4646fa7de5f21e18aa67",
    "location": "/0123456789012/vaults/my-vault/archives/
kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw"
}
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

Um ein hochgeladenes Archiv abzurufen, initiieren Sie einen Abrufauftrag mit dem Befehl aws glacier initiate-job.

- Einzelheiten zur API finden Sie UploadArchivein der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.UploadArchiveRequest;
import software.amazon.awssdk.services.glacier.model.UploadArchiveResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import java.io.File;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UploadArchive {

    static final int ONE_MB = 1024 * 1024;

    public static void main(String[] args) {
        final String usage = """

                Usage:   <strPath> <vaultName>\s

                Where:
                   strPath - The path to the archive to upload (for example, C:\
\AWS\\test.pdf).
                   vaultName - The name of the vault.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String strPath = args[0];
```

```
        String vaultName = args[1];
        File myFile = new File(strPath);
        Path path = Paths.get(strPath);
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String archiveId = uploadContent(glacier, path, vaultName, myFile);
        System.out.println("The ID of the archived item is " + archiveId);
        glacier.close();
    }

    public static String uploadContent(GlacierClient glacier, Path path, String
vaultName, File myFile) {
        // Get an SHA-256 tree hash value.
        String checkVal = computeSHA256(myFile);
        try {
            UploadArchiveRequest uploadRequest = UploadArchiveRequest.builder()
                    .vaultName(vaultName)
                    .checksum(checkVal)
                    .build();

            UploadArchiveResponse res = glacier.uploadArchive(uploadRequest,
path);
            return res.archiveId();

        } catch (GlacierException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    private static String computeSHA256(File inputFile) {
        try {
            byte[] treeHash = computeSHA256TreeHash(inputFile);
            System.out.printf("SHA-256 tree hash = %s\n", toHex(treeHash));
            return toHex(treeHash);

        } catch (IOException ioe) {
            System.err.format("Exception when reading from file %s: %s",
inputFile, ioe.getMessage());
            System.exit(-1);
```

```
        } catch (NoSuchAlgorithmException nsae) {
            System.err.format("Cannot locate MessageDigest algorithm for SHA-256:
%s", nsae.getMessage());
            System.exit(-1);
        }
        return "";
    }

    public static byte[] computeSHA256TreeHash(File inputFile) throws
IOException,
            NoSuchAlgorithmException {

        byte[][] chunkSHA256Hashes = getChunkSHA256Hashes(inputFile);
        return computeSHA256TreeHash(chunkSHA256Hashes);
    }

    /**
     * Computes an SHA256 checksum for each 1 MB chunk of the input file. This
     * includes the checksum for the last chunk, even if it's smaller than 1 MB.
     */
    public static byte[][] getChunkSHA256Hashes(File file) throws IOException,
            NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        long numChunks = file.length() / ONE_MB;
        if (file.length() % ONE_MB > 0) {
            numChunks++;
        }

        if (numChunks == 0) {
            return new byte[][] { md.digest() };
        }

        byte[][] chunkSHA256Hashes = new byte[(int) numChunks][];
        FileInputStream fileStream = null;

        try {
            fileStream = new FileInputStream(file);
            byte[] buff = new byte[ONE_MB];

            int bytesRead;
            int idx = 0;

            while ((bytesRead = fileStream.read(buff, 0, ONE_MB)) > 0) {
```

```
                md.reset();
                md.update(buff, 0, bytesRead);
                chunkSHA256Hashes[idx++] = md.digest();
            }

            return chunkSHA256Hashes;

        } finally {
            if (fileStream != null) {
                try {
                    fileStream.close();
                } catch (IOException ioe) {
                    System.err.printf("Exception while closing %s.\n %s",
file.getName(),
                            ioe.getMessage());
                }
            }
        }
    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1 MB chunk
     * checksums.
     */
    public static byte[] computeSHA256TreeHash(byte[][] chunkSHA256Hashes)
            throws NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        byte[][] prevLvlHashes = chunkSHA256Hashes;
        while (prevLvlHashes.length > 1) {
            int len = prevLvlHashes.length / 2;
            if (prevLvlHashes.length % 2 != 0) {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];
            int j = 0;
            for (int i = 0; i < prevLvlHashes.length; i = i + 2, j++) {

                // If there are at least two elements remaining.
                if (prevLvlHashes.length - i > 1) {

                    // Calculate a digest of the concatenated nodes.
                    md.reset();
```

```
                    md.update(prevLvlHashes[i]);
                    md.update(prevLvlHashes[i + 1]);
                    currLvlHashes[j] = md.digest();

                } else { // Take care of the remaining odd chunk
                    currLvlHashes[j] = prevLvlHashes[i];
                }
            }

            prevLvlHashes = currLvlHashes;
        }

        return prevLvlHashes[0];
    }

    /**
     * Returns the hexadecimal representation of the input byte array
     */
    public static String toHex(byte[] data) {
        StringBuilder sb = new StringBuilder(data.length * 2);
        for (byte datum : data) {
            String hex = Integer.toHexString(datum & 0xFF);

            if (hex.length() == 1) {
                // Append leading zero.
                sb.append("0");
            }
            sb.append(hex);
        }
        return sb.toString().toLowerCase();
    }
}
```

- Einzelheiten zur API finden Sie UploadArchivein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie den Client.

```
const { GlacierClient } = require("@aws-sdk/client-glacier");
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const glacierClient = new GlacierClient({ region: REGION });
export { glacierClient };
```

Hochladen des Archivs.

```
// Load the SDK for JavaScript
import { UploadArchiveCommand } from "@aws-sdk/client-glacier";
import { glacierClient } from "./libs/glacierClient.js";

// Set the parameters
const vaultname = "VAULT_NAME"; // VAULT_NAME

// Create a new service object and buffer
const buffer = new Buffer.alloc(2.5 * 1024 * 1024); // 2.5MB buffer
const params = { vaultName: vaultname, body: buffer };

const run = async () => {
  try {
    const data = await glacierClient.send(new UploadArchiveCommand(params));
    console.log("Archive ID", data.archiveId);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error uploading archive!", err);
  }
};
```

```
run();
```

- Weitere Informationen finden Sie im AWS SDK für JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie UploadArchivein der AWS SDK für JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
// Load the SDK for JavaScript
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create a new service object and buffer
var glacier = new AWS.Glacier({ apiVersion: "2012-06-01" });
buffer = Buffer.alloc(2.5 * 1024 * 1024); // 2.5MB buffer

var params = { vaultName: "YOUR_VAULT_NAME", body: buffer };
// Call Glacier to upload the archive.
glacier.uploadArchive(params, function (err, data) {
  if (err) {
    console.log("Error uploading archive!", err);
  } else {
    console.log("Archive ID", data.archiveId);
  }
});
```

- Weitere Informationen finden Sie im AWS SDK für JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie UploadArchivein der AWS SDK für JavaScript API-Referenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Lädt eine einzelne Datei in den angegebenen Tresor hoch und gibt die Archiv-ID und die berechnete Prüfsumme zurück.

```
Write-GLCArchive -VaultName myvault -FilePath c:\temp\blue.bin
```

Ausgabe:

```
FilePath                  ArchiveId             Checksum
--------                  ---------             --------
C:\temp\blue.bin          o9O9jUUs...TTX-TpIhQJw 79f3e...f4395b
```

Beispiel 2: Lädt den Inhalt einer Ordnerhierarchie in den angegebenen Tresor im Benutzerkonto hoch. Für jede hochgeladene Datei gibt das Cmdlet den Dateinamen, die entsprechende Archiv-ID und die berechnete Prüfsumme des Archivs aus.

```
Write-GLCArchive -VaultName myvault -FolderPath . -Recurse
```

Ausgabe:

```
FilePath                  ArchiveId             Checksum
--------                  ---------             --------
C:\temp\blue.bin          o9O9jUUs...TTX-TpIhQJw 79f3e...f4395b
C:\temp\green.bin         qXAfOdSG...czo729UHXrw d50a1...9184b9
C:\temp\lum.bin           39aNifP3...q9nb8nZkFIg 28886...5c3e27
C:\temp\red.bin           vp7E6rU_...Ejk_HhjAxKA e05f7...4e34f5
C:\temp\Folder1\file1.txt _eRINlip...5Sxy7dD2BaA d0d2a...c8a3ba
C:\temp\Folder2\file2.iso -Ix3jlmu...iXiDh-XfOPA 7469e...3e86f1
```

- Einzelheiten zur API finden Sie unter UploadArchive AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Lädt eine einzelne Datei in den angegebenen Tresor hoch und gibt die Archiv-ID und die berechnete Prüfsumme zurück.

```
Write-GLCArchive -VaultName myvault -FilePath c:\temp\blue.bin
```

Ausgabe:

```
FilePath                    ArchiveId             Checksum
--------                    ---------             --------
C:\temp\blue.bin            o9O9jUUs...TTX-TpIhQJw 79f3e...f4395b
```

Beispiel 2: Lädt den Inhalt einer Ordnerhierarchie in den angegebenen Tresor im Benutzerkonto hoch. Für jede hochgeladene Datei gibt das Cmdlet den Dateinamen, die entsprechende Archiv-ID und die berechnete Prüfsumme des Archivs aus.

```
Write-GLCArchive -VaultName myvault -FolderPath . -Recurse
```

Ausgabe:

```
FilePath                    ArchiveId             Checksum
--------                    ---------             --------
C:\temp\blue.bin            o9O9jUUs...TTX-TpIhQJw 79f3e...f4395b
C:\temp\green.bin           qXAfOdSG...czo729UHXrw d50a1...9184b9
C:\temp\lum.bin             39aNifP3...q9nb8nZkFIg 28886...5c3e27
C:\temp\red.bin             vp7E6rU_...Ejk_HhjAxKA e05f7...4e34f5
C:\temp\Folder1\file1.txt   _eRINlip...5Sxy7dD2BaA d0d2a...c8a3ba
C:\temp\Folder2\file2.iso   -Ix3jlmu...iXiDh-XfOPA 7469e...3e86f1
```

- Einzelheiten zur API finden Sie unter UploadArchive AWS -Tools für PowerShellCmdlet-Referenz (V5).

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
```

```
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def upload_archive(vault, archive_description, archive_file):
        """
        Uploads an archive to a vault.

        :param vault: The vault where the archive is put.
        :param archive_description: A description of the archive.
        :param archive_file: The archive file to put in the vault.
        :return: The uploaded archive.
        """
        try:
            archive = vault.upload_archive(
                archiveDescription=archive_description, body=archive_file
            )
            logger.info(
                "Uploaded %s with ID %s to vault %s.",
                archive_description,
                archive.id,
                vault.name,
            )
        except ClientError:
            logger.exception(
                "Couldn't upload %s to %s.", archive_description, vault.name
            )
            raise
        else:
            return archive
```

- Einzelheiten zur API finden Sie UploadArchivein AWS SDK for Python (Boto3) API Reference.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Amazon Glacier mit einem AWS SDK verwenden Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`UploadMultipartPart`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `UploadMultipartPart` verwendet wird.

CLI

AWS CLI

Der folgende Befehl lädt den ersten, 1 MiB (1024 x 1024 Byte) umfassenden Teil eines Archivs hoch:

```
aws glacier upload-multipart-part --body part1 --range 'bytes
 0-1048575/*' --account-id - --vault-name my-vault --upload-
id 19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ
```

Amazon Glacier benötigt bei der Durchführung von Operationen ein Konto-ID-Argument, Sie können jedoch einen Bindestrich verwenden, um das verwendete Konto anzugeben.

Der Parameter „body“ nimmt einen Pfad zu einer Teildatei im lokalen Dateisystem an. Der Parameter „range“ nimmt einen HTTP-Inhaltsbereich an, der die Byte angibt, die der Teil im fertigen Archiv belegt. Die Upload-ID wird vom Befehl `aws glacier initiate-multipart-upload` zurückgegeben und kann auch mithilfe von `aws glacier list-multipart-uploads` abgerufen werden.

Weitere Informationen zu mehrteiligen Uploads auf Amazon Glacier mithilfe der AWS CLI finden Sie unter Verwenden von Amazon Glacier im AWS CLI-Benutzerhandbuch.

- Einzelheiten zur API finden Sie UploadMultipartPartin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen eines mehrteiligen Uploads mit 1-Megabyte-Blöcken eines Buffer-Objekts.

```
// Create a new service object and some supporting variables
var glacier = new AWS.Glacier({ apiVersion: "2012-06-01" }),
  vaultName = "YOUR_VAULT_NAME",
  buffer = new Buffer(2.5 * 1024 * 1024), // 2.5MB buffer
  partSize = 1024 * 1024, // 1MB chunks,
  numPartsLeft = Math.ceil(buffer.length / partSize),
  startTime = new Date(),
  params = { vaultName: vaultName, partSize: partSize.toString() };

// Compute the complete SHA-256 tree hash so we can pass it
// to completeMultipartUpload request at the end
var treeHash = glacier.computeChecksums(buffer).treeHash;

// Initiate the multipart upload
console.log("Initiating upload to", vaultName);
// Call Glacier to initiate the upload.
glacier.initiateMultipartUpload(params, function (mpErr, multipart) {
  if (mpErr) {
    console.log("Error!", mpErr.stack);
    return;
  }
  console.log("Got upload ID", multipart.uploadId);

  // Grab each partSize chunk and upload it as a part
  for (var i = 0; i < buffer.length; i += partSize) {
    var end = Math.min(i + partSize, buffer.length),
      partParams = {
        vaultName: vaultName,
        uploadId: multipart.uploadId,
        range: "bytes " + i + "-" + (end - 1) + "/*",
        body: buffer.slice(i, end),
      };

    // Send a single part
    console.log("Uploading part", i, "=", partParams.range);
    glacier.uploadMultipartPart(partParams, function (multiErr, mData) {
      if (multiErr) return;
      console.log("Completed part", this.request.params.range);
      if (--numPartsLeft > 0) return; // complete only when all parts uploaded

      var doneParams = {
        vaultName: vaultName,
        uploadId: multipart.uploadId,
```

```
        archiveSize: buffer.length.toString(),
        checksum: treeHash, // the computed tree hash
      };

      console.log("Completing upload...");
      glacier.completeMultipartUpload(doneParams, function (err, data) {
        if (err) {
          console.log("An error occurred while uploading the archive");
          console.log(err);
        } else {
          var delta = (new Date() - startTime) / 1000;
          console.log("Completed upload in", delta, "seconds");
          console.log("Archive ID:", data.archiveId);
          console.log("Checksum:  ", data.checksum);
        }
      });
    });
  }
});
```

- Weitere Informationen finden Sie im AWS SDK für JavaScript -Entwicklerhandbuch.
- Einzelheiten zur API finden Sie UploadMultipartPartin der AWS SDK für JavaScript API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für die Verwendung von Amazon Glacier AWS SDKs

Die folgenden Codebeispiele zeigen Ihnen, wie Sie gängige Szenarien in Amazon Glacier mit implementieren AWS SDKs. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben erledigen können, indem Sie mehrere Funktionen in Amazon Glacier aufrufen oder mit anderen kombinieren AWS-Services. Jedes Szenario enthält einen Link zum vollständigen Quell-Code, wo Sie Anleitungen zum Einrichten und Ausführen des Codes finden.

Szenarien zielen auf eine mittlere Erfahrungsebene ab, um Ihnen zu helfen, Service-Aktionen im Kontext zu verstehen.

Beispiele

- Archivieren Sie eine Datei in Amazon Glacier, erhalten Sie Benachrichtigungen und initiieren Sie einen Job mithilfe eines AWS SDK
- Rufen Sie Amazon Glacier-Archivinhalte ab und löschen Sie das Archiv mithilfe eines AWS SDK

Archivieren Sie eine Datei in Amazon Glacier, erhalten Sie Benachrichtigungen und initiieren Sie einen Job mithilfe eines AWS SDK

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Erstellen Sie einen Amazon Glacier-Tresor.
- Konfigurieren des Tresors, um Benachrichtigungen in einem Amazon-SNS-Thema zu veröffentlichen.
- Hochladen einer Archivdatei in den Tresor
- Initiieren eines Auftrags zum Abrufen eines Archivs

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie eine Klasse, die Amazon Glacier-Operationen umschließt.

```
import argparse
import logging
import os
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


class GlacierWrapper:
```

```
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    def create_vault(self, vault_name):
        """
        Creates a vault.

        :param vault_name: The name to give the vault.
        :return: The newly created vault.
        """
        try:
            vault = self.glacier_resource.create_vault(vaultName=vault_name)
            logger.info("Created vault %s.", vault_name)
        except ClientError:
            logger.exception("Couldn't create vault %s.", vault_name)
            raise
        else:
            return vault


    def list_vaults(self):
        """
        Lists vaults for the current account.
        """
        try:
            for vault in self.glacier_resource.vaults.all():
                logger.info("Got vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't list vaults.")
            raise


    @staticmethod
    def upload_archive(vault, archive_description, archive_file):
        """
        Uploads an archive to a vault.

        :param vault: The vault where the archive is put.
```

```
        :param archive_description: A description of the archive.
        :param archive_file: The archive file to put in the vault.
        :return: The uploaded archive.
        """
        try:
            archive = vault.upload_archive(
                archiveDescription=archive_description, body=archive_file
            )
            logger.info(
                "Uploaded %s with ID %s to vault %s.",
                archive_description,
                archive.id,
                vault.name,
            )
        except ClientError:
            logger.exception(
                "Couldn't upload %s to %s.", archive_description, vault.name
            )
            raise
        else:
            return archive


    @staticmethod
    def initiate_archive_retrieval(archive):
        """
        Initiates an archive retrieval job. Standard retrievals typically
complete
        within 3–5 hours. When the job completes, you can get the archive
contents
        by calling get_output().

        :param archive: The archive to retrieve.
        :return: The archive retrieval job.
        """
        try:
            job = archive.initiate_archive_retrieval()
            logger.info("Started %s job with ID %s.", job.action, job.id)
        except ClientError:
            logger.exception("Couldn't start job on archive %s.", archive.id)
            raise
        else:
            return job
```

```
    @staticmethod
    def list_jobs(vault, job_type):
        """
        Lists jobs by type for the specified vault.

        :param vault: The vault to query.
        :param job_type: The type of job to list.
        :return: The list of jobs of the requested type.
        """
        job_list = []
        try:
            if job_type == "all":
                jobs = vault.jobs.all()
            elif job_type == "in_progress":
                jobs = vault.jobs_in_progress.all()
            elif job_type == "completed":
                jobs = vault.completed_jobs.all()
            elif job_type == "succeeded":
                jobs = vault.succeeded_jobs.all()
            elif job_type == "failed":
                jobs = vault.failed_jobs.all()
            else:
                jobs = []
                logger.warning("%s isn't a type of job I can get.", job_type)
            for job in jobs:
                job_list.append(job)
                logger.info("Got %s %s job %s.", job_type, job.action, job.id)
        except ClientError:
            logger.exception("Couldn't get %s jobs from %s.", job_type,
vault.name)
            raise
        else:
            return job_list


    def set_notifications(self, vault, sns_topic_arn):
        """
        Sets an Amazon Simple Notification Service (Amazon SNS) topic as a target
        for notifications. Amazon S3 Glacier publishes messages to this topic for
        the configured list of events.

        :param vault: The vault to set up to publish notifications.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of the topic that
```

```
                                receives notifications.
        :return: Data about the new notification configuration.
        """
        try:
            notification = self.glacier_resource.Notification("-", vault.name)
            notification.set(
                vaultNotificationConfig={
                    "SNSTopic": sns_topic_arn,
                    "Events": [
                        "ArchiveRetrievalCompleted",
                        "InventoryRetrievalCompleted",
                    ],
                }
            )
            logger.info(
                "Notifications will be sent to %s for events %s from %s.",
                notification.sns_topic,
                notification.events,
                notification.vault_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't set notifications to %s on %s.", sns_topic_arn,
 vault.name
            )
            raise
        else:
            return notification
```

Rufen Sie Funktionen in der Wrapper-Klasse auf, um einen Tresor zu erstellen und eine Datei hochzuladen. Konfigurieren Sie dann den Tresor so, dass er Benachrichtigungen veröffentlicht, und initiieren Sie einen Auftrag zum Abrufen des Archivs.

```
def upload_demo(glacier, vault_name, topic_arn):
    """
    Shows how to:
    * Create a vault.
    * Configure the vault to publish notifications to an Amazon SNS topic.
    * Upload an archive.
    * Start a job to retrieve the archive.
```

```
    :param glacier: A Boto3 Amazon S3 Glacier resource.
    :param vault_name: The name of the vault to create.
    :param topic_arn: The ARN of an Amazon SNS topic that receives notification
of
                      Amazon S3 Glacier events.
    """
    print(f"\nCreating vault {vault_name}.")
    vault = glacier.create_vault(vault_name)
    print("\nList of vaults in your account:")
    glacier.list_vaults()
    print(f"\nUploading glacier_basics.py to {vault.name}.")
    with open("glacier_basics.py", "rb") as upload_file:
        archive = glacier.upload_archive(vault, "glacier_basics.py", upload_file)
    print(
        "\nStarting an archive retrieval request to get the file back from the "
        "vault."
    )
    glacier.initiate_archive_retrieval(archive)
    print("\nListing in progress jobs:")
    glacier.list_jobs(vault, "in_progress")
    print(
        "\nBecause Amazon S3 Glacier is intended for infrequent retrieval, an "
        "archive request with Standard retrieval typically completes within 3-5 "
        "hours."
    )
    if topic_arn:
        notification = glacier.set_notifications(vault, topic_arn)
        print(
            f"\nVault {vault.name} is configured to notify the "
            f"{notification.sns_topic} topic when {notification.events} "
            f"events occur. You can subscribe to this topic to receive "
            f"a message when the archive retrieval completes.\n"
        )
    else:
        print(
            f"\nVault {vault.name} is not configured to notify an Amazon SNS
topic "
            f"when the archive retrieval completes so wait a few hours."
        )
    print("\nRetrieve your job output by running this script with the --retrieve
flag.")
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für Python (Boto3).
  - CreateVault
  - InitiateJob
  - ListJobs
  - ListVaults
  - SetVaultNotifications
  - UploadArchive

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Rufen Sie Amazon Glacier-Archivinhalte ab und löschen Sie das Archiv mithilfe eines AWS SDK

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Jobs für einen Amazon Glacier-Tresor auflisten und den Auftragsstatus abrufen.
- Abrufen der Ausgabe eines abgeschlossenen Auftrags zum Archivabruf
- Löschen eines Archivs
- Löschen eines Tresors

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie eine Klasse, die Amazon Glacier-Operationen umschließt.

```
import argparse
import logging
import os
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def list_jobs(vault, job_type):
        """
        Lists jobs by type for the specified vault.

        :param vault: The vault to query.
        :param job_type: The type of job to list.
        :return: The list of jobs of the requested type.
        """
        job_list = []
        try:
            if job_type == "all":
                jobs = vault.jobs.all()
            elif job_type == "in_progress":
                jobs = vault.jobs_in_progress.all()
            elif job_type == "completed":
                jobs = vault.completed_jobs.all()
            elif job_type == "succeeded":
                jobs = vault.succeeded_jobs.all()
            elif job_type == "failed":
                jobs = vault.failed_jobs.all()
            else:
                jobs = []
                logger.warning("%s isn't a type of job I can get.", job_type)
```

```
            for job in jobs:
                job_list.append(job)
                logger.info("Got %s %s job %s.", job_type, job.action, job.id)
        except ClientError:
            logger.exception("Couldn't get %s jobs from %s.", job_type,
vault.name)
            raise
        else:
            return job_list


    @staticmethod
    def get_job_output(job):
        """
        Gets the output of a job, such as a vault inventory or the contents of an
        archive.

        :param job: The job to get output from.
        :return: The job output, in bytes.
        """
        try:
            response = job.get_output()
            out_bytes = response["body"].read()
            logger.info("Read %s bytes from job %s.", len(out_bytes), job.id)
            if "archiveDescription" in response:
                logger.info(
                    "These bytes are described as '%s'",
response["archiveDescription"]
                )
        except ClientError:
            logger.exception("Couldn't get output for job %s.", job.id)
            raise
        else:
            return out_bytes


    @staticmethod
    def delete_archive(archive):
        """
        Deletes an archive from a vault.

        :param archive: The archive to delete.
        """
        try:
```

```
            archive.delete()
            logger.info(
                "Deleted archive %s from vault %s.", archive.id,
 archive.vault_name
            )
        except ClientError:
            logger.exception("Couldn't delete archive %s.", archive.id)
            raise


    @staticmethod
    def delete_vault(vault):
        """
        Deletes a vault.

        :param vault: The vault to delete.
        """
        try:
            vault.delete()
            logger.info("Deleted vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't delete vault %s.", vault.name)
            raise
```

Rufen Sie Funktionen der Wrapper-Klasse auf, um Archivinhalte aus einem abgeschlossenen Auftrag abzurufen, und löschen Sie dann das Archiv.

```
def retrieve_demo(glacier, vault_name):
    """
    Shows how to:
    * List jobs for a vault and get job status.
    * Get the output of a completed archive retrieval job.
    * Delete an archive.
    * Delete a vault.

    :param glacier: A Boto3 Amazon S3 Glacier resource.
    :param vault_name: The name of the vault to query for jobs.
    """
    vault = glacier.glacier_resource.Vault("-", vault_name)
    try:
        vault.load()
```

```
    except ClientError as err:
        if err.response["Error"]["Code"] == "ResourceNotFoundException":
            print(
                f"\nVault {vault_name} doesn't exist. You must first run this
script "
                f"with the --upload flag to create the vault."
            )
            return
        else:
            raise

    print(f"\nGetting completed jobs for {vault.name}.")
    jobs = glacier.list_jobs(vault, "completed")
    if not jobs:
        print("\nNo completed jobs found. Give it some time and try again
later.")
        return

    retrieval_job = None
    for job in jobs:
        if job.action == "ArchiveRetrieval" and job.status_code == "Succeeded":
            retrieval_job = job
            break
    if retrieval_job is None:
        print(
            "\nNo ArchiveRetrieval jobs found. Give it some time and try again "
            "later."
        )
        return

    print(f"\nGetting output from job {retrieval_job.id}.")
    archive_bytes = glacier.get_job_output(retrieval_job)
    archive_str = archive_bytes.decode("utf-8")
    print("\nGot archive data. Printing the first 10 lines.")
    print(os.linesep.join(archive_str.split(os.linesep)[:10]))

    print(f"\nDeleting the archive from {vault.name}.")
    archive = glacier.glacier_resource.Archive(
        "-", vault.name, retrieval_job.archive_id
    )
    glacier.delete_archive(archive)

    print(f"\nDeleting {vault.name}.")
    glacier.delete_vault(vault)
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für Python (Boto3).
  - DeleteArchive
  - DeleteVault
  - GetJobOutput
  - ListJobs

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Glacier mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Sicherheit in Amazon Glacier

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die auf die Anforderungen der sicherheitssensibelsten Unternehmen zugeschnitten sind.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, auf der AWS Dienste in der ausgeführt AWS Cloud werden. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Die Wirksamkeit unserer Sicherheitsfunktionen wird regelmäßig von externen Prüfern im Rahmen des AWS -Compliance-Programms getestet und überprüft. Informationen zu den Compliance-Programmen, die für Amazon Glacier (Amazon Glacier) gelten, finden Sie unter AWS Services in Scope by Compliance Program.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. 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 hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Nutzung von Amazon Glacier anwenden können. In den folgenden Themen erfahren Sie, wie Sie Amazon Glacier konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, die Ihnen bei der Überwachung und Sicherung Ihrer Amazon Glacier-Ressourcen helfen können.

Topics

- Datenschutz in Amazon Glacier
- Identity and Access Management für Amazon Glacier
- Protokollierung und Überwachung in Amazon Glacier
- Konformitätsprüfung für Amazon Glacier
- Resilienz in Amazon Glacier
- Infrastruktursicherheit in Amazon Glacier

Datenschutz in Amazon Glacier

Amazon Glacier (Amazon Glacier) bietet äußerst langlebigen Cloud-Speicher für Datenarchivierung und Langzeitsicherung. Amazon Glacier wurde für eine Lebensdauer von 99,999999999 Prozent konzipiert und bietet umfassende Sicherheits- und Compliance-Funktionen, mit denen Sie strenge gesetzliche Anforderungen erfüllen können. Amazon Glacier speichert Daten redundant in mehreren AWS Availability Zones (AZ) und auf mehreren Geräten innerhalb jeder AZ. Um die Haltbarkeit zu erhöhen, speichert Amazon Glacier Ihre Daten synchron über mehrere, AZs bevor ein erfolgreicher Upload bestätigt wird.

Weitere Informationen zur AWS globalen Cloud-Infrastruktur finden Sie unter Globale Infrastruktur.

Aus Datenschutzgründen empfehlen wir, dass Sie Ihre AWS-Konto Anmeldeinformationen schützen und einzelnen Benutzern, Gruppen oder Rollen nur die Berechtigungen gewähren, die zur Erfüllung ihrer Aufgaben erforderlich sind.

Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

Themen

- Datenverschlüsselung
- Schlüsselverwaltung
- Richtlinie für den Datenverkehr zwischen Netzwerken

Datenverschlüsselung

Datenschutz bezieht sich auf den Schutz von Daten während der Übertragung (auf dem Weg zu und von Amazon Glacier) und im Ruhezustand (während sie in AWS Rechenzentren gespeichert werden). Sie können Daten während der Übertragung, die direkt auf Amazon Glacier hochgeladen werden, mithilfe von Secure Sockets Layer (SSL) oder clientseitiger Verschlüsselung schützen.

Sie können auch über Amazon S3 auf Amazon Glacier zugreifen. Amazon S3 unterstützt die Lebenszykluskonfiguration in einem Amazon S3 S3-Bucket, sodass Sie Objekte zur Archivierung in die Amazon Glacier-Speicherklasse übertragen können. Daten, die über Lebenszyklusrichtlinien zwischen Amazon S3 und Amazon Glacier übertragen werden, werden mit SSL verschlüsselt.

In Amazon Glacier gespeicherte Daten werden automatisch serverseitig mit dem 256-Bit-Advanced Encryption Standard (AES-256) verschlüsselt, wobei die Schlüssel von verwaltet werden. AWS Wenn Sie es vorziehen, Ihre eigenen Schlüssel zu verwalten, können Sie vor dem Speichern von Daten in Amazon Glacier auch eine clientseitige Verschlüsselung verwenden. Weitere Informationen über die Standardverschlüsselungsfunktion von Amazon S3 finden Sie im Artikel zur Amazon-S3-Standardverschlüsselung im Benutzerhandbuch für Amazon Simple Storage Service.

Schlüsselverwaltung

Die serverseitige Verschlüsselung befasst sich mit der Datenverschlüsselung im Ruhezustand, d. h. Amazon Glacier verschlüsselt Ihre Daten, während es sie in seine Rechenzentren schreibt, und entschlüsselt sie für Sie, wenn Sie darauf zugreifen. Wenn Sie Ihre Anfrage authentifizieren und Zugriffsberechtigungen besitzen, gibt es in Bezug auf die Art und Weise, wie Sie auf verschlüsselte oder nicht verschlüsselte Daten zugreifen, keinen Unterschied.

In Amazon Glacier gespeicherte Daten werden automatisch serverseitig mit AES-256 verschlüsselt, wobei Schlüssel verwendet werden, die von verwaltet werden. AWS Als zusätzliche Sicherheitsmaßnahme wird der Schlüssel selbst mit einem Stammschlüssel AWS verschlüsselt, den wir regelmäßig wechseln.

Richtlinie für den Datenverkehr zwischen Netzwerken

Der Zugriff auf Amazon Glacier über das Netzwerk erfolgt über AWS Published APIs. Clients müssen Transport Layer Security (TLS) 1.2 unterstützen. Wir empfehlen TLS 1.3 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 Systeme wie Java 7 und höher unterstützen diese Modi. Außerdem müssen Sie die Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signieren, die einem IAM-Prinzipal zugeordnet sind. Sie können auch AWS -Security-Token-Service (AWS STS) verwenden um temporäre Sicherheitsanmeldeinformationen zu generieren.

VPC-Endpunkte

Ein Virtual Private Cloud (VPC) -Endpunkt ermöglicht es Ihnen, Ihre VPC privat mit unterstützten AWS Diensten und VPC-Endpunktdiensten zu verbinden, die bereitgestellt werden, AWS PrivateLink ohne dass ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder eine Verbindung erforderlich ist. AWS Direct Connect Obwohl Amazon Glacier VPC-Endpunkte nicht direkt unterstützt, können Sie die VPC-Endpunkte von Amazon Simple Storage Service (Amazon S3) nutzen, wenn Sie auf Amazon Glacier als in Amazon S3 integrierte Speicherebene zugreifen.

Weitere Informationen zur Amazon S3 S3-Lebenszykluskonfiguration und zur Übertragung von Objekten in die Amazon Glacier-Speicherklasse finden Sie unter Object Lifecycle Management and Transitioning Objects im Amazon Simple Storage Service-Benutzerhandbuch. Weitere Informationen zu VPC-Endpunkten finden Sie unter VPC-Endpunkte im Amazon-VPC-Benutzerhandbuch.

Identity and Access Management für Amazon Glacier

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Amazon Glacier-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- So funktioniert Amazon Glacier mit IAM
- Beispiele für identitätsbasierte Richtlinien für Amazon Glacier
- Beispiele für ressourcenbasierte Richtlinien für Amazon Glacier
- Problembehandlung bei Identität und Zugriff auf Amazon Glacier
- Referenztabelle für -API-Berechtigungen

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von Ihrer Rolle ab:

- Servicebenutzer – Fordern Sie von Ihrem Administrator Berechtigungen an, wenn Sie nicht auf Features zugreifen können (siehe Problembehandlung bei Identität und Zugriff auf Amazon Glacier).
- Serviceadministrator – Bestimmen Sie den Benutzerzugriff und stellen Sie Berechtigungsanfragen (siehe So funktioniert Amazon Glacier mit IAM).
- IAM-Administrator – Schreiben Sie Richtlinien zur Zugriffsverwaltung (siehe Beispiele für identitätsbasierte Richtlinien für Amazon Glacier).

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen sich als IAM-Benutzer authentifizieren oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich als föderierte Identität anmelden, indem Sie Anmeldeinformationen aus einer Identitätsquelle wie AWS IAM Identity Center (IAM Identity Center), Single Sign-On-Authentifizierung oder Anmeldeinformationen verwenden. Google/Facebook Weitere Informationen zum Anmelden finden Sie unter So melden Sie sich bei Ihrem AWS-Konto an im Benutzerhandbuch für AWS-Anmeldung .

AWS Bietet für den programmatischen Zugriff ein SDK und eine CLI zum kryptografischen Signieren von Anfragen. Weitere Informationen finden Sie unter AWS Signature Version 4 for API requests im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, dem sogenannten AWS-Konto Root-Benutzer, der vollständigen Zugriff auf alle AWS-Services Ressourcen hat. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Eine Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Tasks that require root user credentials im IAM-Benutzerhandbuch.

Verbundidentität

Es hat sich bewährt, dass menschliche Benutzer für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden müssen.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensverzeichnis, Ihrem Directory Service Web-Identitätsanbieter oder der AWS-Services mithilfe von Anmeldeinformationen aus einer Identitätsquelle zugreift. Verbundene Identitäten übernehmen Rollen, die temporäre Anmeldeinformationen bereitstellen.

Für die zentrale Zugriffsverwaltung empfehlen wir AWS IAM Identity Center. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität mit bestimmten Berechtigungen für eine einzelne Person oder Anwendung. Wir empfehlen die Verwendung temporärer Anmeldeinformationen anstelle

von IAM-Benutzern mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erfordern, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um AWS mithilfe temporärer Anmeldeinformationen darauf zugreifen zu können.

Eine IAM-Gruppe spezifiziert eine Sammlung von IAM-Benutzern und erleichtert die Verwaltung von Berechtigungen für große Gruppen von Benutzern. Weitere Informationen finden Sie unter Anwendungsfälle für IAM-Benutzer im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität mit spezifischen Berechtigungen, die temporäre Anmeldeinformationen bereitstellt. Sie können eine Rolle übernehmen, indem Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) wechseln oder indem Sie eine AWS Oder-API-Operation AWS CLI aufrufen. Weitere Informationen finden Sie unter Methoden, um eine Rolle zu übernehmen im IAM-Benutzerhandbuch.

IAM-Rollen sind nützlich für den Verbundbenutzer-Zugriff, temporäre IAM-Benutzerberechtigungen, kontoübergreifenden Zugriff, serviceübergreifenden Zugriff und Anwendungen, die auf Amazon EC2 laufen. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie definiert Berechtigungen, wenn sie mit einer Identität oder Ressource verknüpft sind. AWS bewertet diese Richtlinien, wenn ein Principal eine Anfrage stellt. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Mit Hilfe von Richtlinien legen Administratoren fest, wer Zugriff auf was hat, indem sie definieren, welches Prinzipal welche Aktionen auf welchen Ressourcenund unter welchen Bedingungendurchführen darf.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator erstellt IAM-Richtlinien und fügt sie zu Rollen hinzu, die die Benutzer dann übernehmen können. IAM-Richtlinien definieren Berechtigungen unabhängig von der Methode, die zur Ausführung der Operation verwendet wird.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität (Benutzer, Gruppe oder Rolle) anfügen können. Diese Richtlinien steuern, welche Aktionen Identitäten für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können Inline-Richtlinien (direkt in eine einzelne Identität eingebettet) oder verwaltete Richtlinien (eigenständige Richtlinien, die mit mehreren Identitäten verbunden sind) sein. Informationen dazu, wie Sie zwischen verwalteten und Inline-Richtlinien wählen, finden Sie unter Choose between managed policies and inline policies im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele hierfür sind Vertrauensrichtlinien für IAM-Rollen und Amazon S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben.

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Weitere Richtlinientypen

AWS unterstützt zusätzliche Richtlinientypen, mit denen die maximalen Berechtigungen festgelegt werden können, die durch gängigere Richtlinientypen gewährt werden:

- Berechtigungsgrenzen – Eine Berechtigungsgrenze legt die maximalen Berechtigungen fest, die eine identitätsbasierte Richtlinie einer IAM-Entität erteilen kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im -IAM-Benutzerhandbuch.
- Richtlinien zur Dienstkontrolle (SCPs) — Geben Sie die maximalen Berechtigungen für eine Organisation oder Organisationseinheit in an AWS Organizations. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Richtlinien zur Ressourcenkontrolle (RCPs) — Legen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten fest. Weitere Informationen finden Sie im AWS Organizations Benutzerhandbuch unter Richtlinien zur Ressourcenkontrolle (RCPs).

- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die als Parameter übergeben werden, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn für eine Anfrage mehrere Arten von Richtlinien gelten, sind die daraus resultierenden Berechtigungen schwieriger zu verstehen. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie unter Bewertungslogik für Richtlinien im IAM-Benutzerhandbuch.

So funktioniert Amazon Glacier mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Amazon Glacier zu verwalten, sollten Sie sich darüber informieren, welche IAM-Funktionen für Amazon Glacier verfügbar sind.

IAM-Funktionen, die Sie mit Amazon Glacier verwenden können

IAM-Feature	Amazon Glacier-Unterstützung
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Ja
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Richtlinienbedingungsschlüssel (services pezifisch)	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Nein
Temporäre Anmeldeinformationen	Ja
Hauptberechtigungen	Nein
Servicerollen	Nein

IAM-Feature	Amazon Glacier-Unterstützung
Serviceverknüpfte Rollen	Nein

Einen allgemeinen Überblick darüber, wie Amazon Glacier und andere AWS Services mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Services, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für Amazon Glacier

Unterstützt Richtlinien auf Identitätsbasis: Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für Amazon Glacier

Beispiele für identitätsbasierte Richtlinien von Amazon Glacier finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Amazon Glacier

Ressourcenbasierte Richtlinien innerhalb von Amazon Glacier

Unterstützt ressourcenbasierte Richtlinien: Ja

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein

bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Der Amazon Glacier-Service unterstützt nur eine Art von ressourcenbasierter Richtlinie, die als Tresorrichtlinie bezeichnet wird und an einen Tresor angehängt ist. Diese Richtlinie definiert, welche Prinzipale Aktionen für den Tresor ausführen können.

Amazon Glacier Vault-Richtlinien verwalten Berechtigungen auf folgende Weise:

- Verwalten von Benutzerberechtigungen in einem Konto mit einer einzelnen Tresorrichtlinie anstelle von mehreren einzelnen Benutzerrichtlinien.
- Verwalten von kontoübergreifenden Berechtigungen als Alternative zur Verwendung von IAM-Rollen.

Beispiele für ressourcenbasierte Richtlinien in Amazon Glacier

Beispiele für ressourcenbasierte Richtlinien von Amazon Glacier finden Sie unter. Beispiele für ressourcenbasierte Richtlinien für Amazon Glacier

Richtlinienmaßnahmen für Amazon Glacier

Unterstützt Richtlinienaktionen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Nehmen Sie Aktionen in eine Richtlinie auf, um Berechtigungen zur Ausführung des zugehörigen Vorgangs zu erteilen.

Eine Liste der Amazon Glacier-Aktionen finden Sie unter Von Amazon Glacier definierte Aktionen in der Service Authorization Reference.

Richtlinienaktionen in Amazon Glacier verwenden das folgende Präfix vor der Aktion:

```
glacier
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
            "glacier:CreateVault",
            "glacier:DescribeVault",
            "glacier:ListVaults"
         ]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "glacier:GetVault*"
```

Beispiele für identitätsbasierte Richtlinien von Amazon Glacier finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Amazon Glacier

Richtlinienressourcen für Amazon Glacier

Unterstützt Richtlinienressourcen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Als Best Practice geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der Amazon Glacier-Ressourcentypen und ihrer Eigenschaften ARNs finden Sie unter Von Amazon Glacier definierte Ressourcen in der Service Authorization Reference. Informationen zu

den Aktionen, für die Sie den ARN der einzelnen Ressourcen angeben können, finden Sie unter Von Amazon Glacier definierte Aktionen.

In Amazon Glacier ist die primäre Ressource ein Tresor. Amazon Glacier unterstützt Richtlinien nur auf Tresorebene. Das heißt, in einer IAM-Richtlinie kann der von Ihnen angegebene `Resource` Wert ein bestimmter Tresor oder eine Reihe von Tresoren in einer bestimmten AWS Region sein. Amazon Glacier unterstützt keine Berechtigungen auf Archivebene.

`Resource`Gibt für alle Amazon Glacier-Aktionen den Tresor an, für den Sie die Berechtigungen erteilen möchten. Diesen Ressourcen sind eindeutige Amazon-Ressourcennamen (ARNs) zugeordnet, wie in der folgenden Tabelle dargestellt, und Sie können ein Platzhalterzeichen (*) im ARN verwenden, um Tresornamen abzugleichen, die mit demselben Präfix beginnen.

Amazon Glacier bietet eine Reihe von Vorgängen für die Arbeit mit den Amazon Glacier-Ressourcen. Weitere Informationen zu den verfügbaren Operationen finden Sie unter API-Referenz für Amazon Glacier.

Einige Amazon Glacier API-Aktionen unterstützen mehrere Ressourcen. `glacier:AddTagsToVault` greift beispielsweise auf „examplevault1“ und „examplevault2“ zu, sodass ein Prinzipal über Berechtigungen für den Zugriff auf beide Ressourcen verfügen muss. Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie sie ARNs durch Kommas.

```
"Resource": [
""arn:aws:glacier:us-west-2:123456789012:vaults/examplevault1",",
""arn:aws:glacier:us-west-2:123456789012:vaults/examplevault2","
]
```

Schlüssel für Richtlinienbedingungen für Amazon Glacier

Unterstützt servicespezifische Richtlinienbedingungsschlüssel: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Condition` gibt an, wann Anweisungen auf der Grundlage definierter Kriterien ausgeführt werden. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren

verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Eine Liste der Amazon Glacier-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Glacier in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von Amazon Glacier definierte Aktionen.

Beispiele für die Verwendung von Glacier-spezifischen Bedingungsschlüsseln finden Sie unter Tresorverriegelungs-Richtlinien.

ACLs auf Amazon Glacier

Unterstützt ACLs: Nein

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

ABAC mit Amazon Glacier

Unterstützt ABAC (Tags in Richtlinien): Nein

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen, auch als Tags bezeichnet, definiert werden. Sie können Tags an IAM-Entitäten und AWS -Ressourcen anhängen und dann ABAC-Richtlinien entwerfen, um Operationen zu ermöglichen, wenn das Tag des Prinzipals mit dem Tag auf der Ressource übereinstimmt.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Definieren von Berechtigungen mit ABAC-Autorisierung im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Temporäre Anmeldeinformationen mit Amazon Glacier verwenden

Unterstützt temporäre Anmeldeinformationen: Ja

Temporäre Anmeldeinformationen ermöglichen kurzfristigen Zugriff auf AWS Ressourcen und werden automatisch erstellt, wenn Sie einen Verbund verwenden oder die Rollen wechseln. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Anmeldeinformationen in IAM und AWS-Services , die mit IAM funktionieren im IAM-Benutzerhandbuch.

Serviceübergreifende Prinzipalberechtigungen für Amazon Glacier

Unterstützt Forward Access Sessions (FAS): Nein

Forward Access Sessions (FAS) verwenden die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. Einzelheiten zu den Richtlinien für FAS-Anforderungen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für Amazon Glacier

Unterstützt Servicerollen: Nein

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle kann die Funktionalität von Amazon Glacier beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn Amazon Glacier Sie dazu anleitet.

Servicebezogene Rollen für Amazon Glacier

Unterstützt serviceverknüpfte Rollen: Ja

Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen.

Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie in der Tabelle nach einem Service mit einem `Yes` in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

Beispiele für identitätsbasierte Richtlinien für Amazon Glacier

Standardmäßig sind Benutzer und Rollen nicht berechtigt, Amazon Glacier-Ressourcen zu erstellen oder zu ändern. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

Einzelheiten zu den von Amazon Glacier definierten Aktionen und Ressourcentypen, einschließlich des Formats ARNs für die einzelnen Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Glacier in der Service Authorization Reference.

Die folgende Beispielrichtlinie gewährt Berechtigungen für drei Amazon Glacier-Tresor-bezogene Aktionen (`glacier:CreateVault`, `glacier:DescribeVault` und`glacier:ListVaults`) für eine Ressource. Dabei wird der Amazon-Ressourcenname (ARN) verwendet, der alle Tresore in der Region identifiziert. `us-west-2` AWS ARNs Ressourcen eindeutig identifizieren. AWS Weitere Informationen zur ARNs Verwendung mit Amazon Glacier finden Sie unterRichtlinienressourcen für Amazon Glacier.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
         "glacier:CreateVault",
```

```
            "glacier:DescribeVault",
            "glacier:ListVaults"
            ],
            "Resource": "arn:aws:glacier:us-west-2:123456789012:vaults/*"
        }
    ]
}
```

Die Richtlinie gewährt Berechtigungen zum Erstellen, Auflisten und Abrufen von Tresorbeschreibungen in der Region `us-west-2`. Das Platzhalterzeichen (*) am Ende des ARN bedeutet, dass diese Anweisung jedem Tresornamen entsprechen kann.

> **Important**
>
> Wenn Sie Berechtigungen zum Erstellen eines Tresors mit der `glacier:CreateVault`-Operation gewähren, müssen Sie das Platzhalterzeichen (*) verwenden, da Sie den Tresornamen nicht kennen, bis Sie den Tresor erstellt haben.

Themen

- Best Practices für Richtlinien
- Verwenden der Amazon Glacier-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Beispiele für vom Kunden verwaltete Richtlinien

Best Practices für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand Amazon Glacier-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder diese löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Wenn Sie identitätsbasierte Richtlinien erstellen oder bearbeiten, befolgen Sie diese Richtlinien und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien

definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien oder Von AWS verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.

- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung mit IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Best Practices für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der Amazon Glacier-Konsole

Um auf die Amazon Glacier-Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den Amazon Glacier-Ressourcen in Ihrem aufzulisten und anzuzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Die Amazon Glacier-Konsole bietet eine integrierte Umgebung, in der Sie Amazon Glacier-Tresore erstellen und verwalten können. Den von Ihnen erstellten IAM-Identitäten müssen mindestens Berechtigungen für die `glacier:ListVaults` Aktion zum Anzeigen der Amazon Glacier-Konsole erteilt werden, wie im folgenden Beispiel gezeigt.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
         {
            "Action": [
            "glacier:ListVaults"
            ],
            "Effect": "Allow",
            "Resource": "*"
         }
      ]
   }
```

AWS adressiert viele gängige Anwendungsfälle durch die Bereitstellung eigenständiger IAM-Richtlinien, die von erstellt und verwaltet werden. AWS Die verwalteten Richtlinien erteilen die erforderlichen Berechtigungen für viele häufige Anwendungsfälle, sodass Sie nicht mühsam ermitteln müssen, welche Berechtigungen erforderlich sind. Weitere Informationen finden Sie unter AWS - verwaltete Richtlinien im IAM-Benutzerhandbuch.

Die folgenden AWS verwalteten Richtlinien, die Sie Benutzern in Ihrem Konto zuordnen können, sind spezifisch für Amazon Glacier:

- AmazonGlacierReadOnlyAccess— Gewährt nur Lesezugriff auf Amazon Glacier über die AWS-Managementkonsole.
- AmazonGlacierFullAccess— Gewährt vollen Zugriff auf Amazon Glacier über die AWS-Managementkonsole.

Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für Amazon Glacier-API-Aktionen und -Ressourcen zu gewähren. Sie können diese benutzerdefinierten Richtlinien an die benutzerdefinierten IAM-Rollen anhängen, die Sie für Ihre Amazon Glacier-Tresore erstellen.

Beide im nächsten Abschnitt erörterten Amazon Glacier AWS Managed-Richtlinien gewähren Berechtigungen für`glacier:ListVaults`.

Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
```

```
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Beispiele für vom Kunden verwaltete Richtlinien

In diesem Abschnitt finden Sie Beispielbenutzerrichtlinien, die Berechtigungen für verschiedene Amazon Glacier-Aktionen gewähren. Diese Richtlinien gelten, wenn Sie die Amazon Glacier REST API, Amazon SDKs, die oder AWS CLI, falls zutreffend, die Amazon Glacier-Managementkonsole verwenden.

> **Note**
>
> Alle Beispiele verwenden die Region USA West (Oregon) (`us-west-2`) und enthalten ein fiktives Konto. IDs

Beispiele

- Beispiel 1: Einem Benutzer das Herunterladen von Archiven aus einem Tresor erlauben
- Beispiel 2: Einem Benutzer erlauben, einen Tresor zu erstellen und Benachrichtigungen zu konfigurieren
- Beispiel 3: Einem Benutzer das Hochladen von Archiven zu einem bestimmten Tresor erlauben
- Beispiel 4: Einem Benutzer volle Berechtigungen für einen bestimmten Tresor erteilen

Beispiel 1: Einem Benutzer das Herunterladen von Archiven aus einem Tresor erlauben

Wenn Sie ein Archiv herunterladen möchten, initiieren Sie zuerst einen Auftrag, um das Archiv abzurufen. Sie können die Daten herunterladen, nachdem der Abrufauftrag abgeschlossen wurde. Die folgende Beispielrichtlinie gewährt Berechtigungen für die `glacier:InitiateJob`-Aktion, um einen Auftrag zu initiieren (der es dem Benutzer ermöglicht, einen Archiv- oder Tresorbestand aus dem Tresor abzurufen), und Berechtigungen für die `glacier:GetJobOutput`-Aktion, um die abgerufenen Daten herunterzuladen. Die Richtlinie gewährt zudem die Berechtigungen zur Durchführung der `glacier:DescribeJob`-Aktion, sodass der Benutzer den Auftragsstatus abrufen kann. Weitere Informationen finden Sie unter Initiate Job (POST jobs).

Mit der Richtlinie werden diese Berechtigungen für einen Tresor namens `examplevault`verliehen. Sie können den Vault-ARN von der Amazon Glacier-Konsole oder programmgesteuert abrufen, indem Sie entweder die Describe Vault (GET vault) oder die List Vaults (GET vaults) API-Aktionen aufrufen.

Beispiel 2: Einem Benutzer erlauben, einen Tresor zu erstellen und Benachrichtigungen zu konfigurieren

Die folgende Beispielrichtlinie gewährt die Berechtigungen zur Erstellung eines Tresor in der Region „us-west-2“, wie im `Resource`-Element angegeben, und zur Konfiguration von Benachrichtigungen. Weitere Informationen zur Arbeit mit Benachrichtigungen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier. Die Richtlinie gewährt auch die Erlaubnis, Tresore in der AWS Region aufzulisten und eine spezifische Tresorbeschreibung abzurufen.

> **Important**
>
> Wenn Sie Berechtigungen zum Erstellen eines Tresors mit der `glacier:CreateVault`-Operation gewähren, müssen Sie das Platzhalterzeichen (*) im `Resource`-Wert verwenden, da Sie den Tresornamen nicht kennen, bis Sie den Tresor erstellt haben.

Beispiel 3: Einem Benutzer das Hochladen von Archiven zu einem bestimmten Tresor erlauben

Die folgende Beispielrichtlinie gewährt die Berechtigungen zum Hochladen von Archiven in einen bestimmten Tresor in der Region „us-west-2“. Diese Berechtigungen ermöglichen es einem Benutzer, ein Archiv mithilfe der API-Operation Upload Archive (POST archive) auf einmal oder mittels der API-Operation Mehrteiligen Upload initiieren (POST multipart uploads) in Teilen hochzuladen.

Beispiel 4: Einem Benutzer volle Berechtigungen für einen bestimmten Tresor erteilen

Die folgende Beispielrichtlinie gewährt Berechtigungen für alle Amazon Glacier-Aktionen in einem Tresor mit dem Namen`examplevault`.

Beispiele für ressourcenbasierte Richtlinien für Amazon Glacier

Einem Amazon Glacier-Tresor können eine Tresorzugriffsrichtlinie und eine Vault Lock-Richtlinie zugeordnet sein. Eine Amazon Glacier-Richtlinie für den Tresorzugriff ist eine ressourcenbasierte Richtlinie, mit der Sie die Berechtigungen für Ihren Tresor verwalten können. Eine Tresorverriegelungs-Richtlinie ist eine Tresorzugriffsrichtlinie, die gesperrt werden kann. Nachdem Sie eine Tresorverriegelungs-Richtlinie gesperrt haben, kann die Richtlinie nicht geändert werden. Mit einer Tresorverriegelungsrichtlinie können Sie Compliance-Steuerung erzwingen.

Themen

- Richtlinien für den Tresorzugriff
- Tresorverriegelungs-Richtlinien

Richtlinien für den Tresorzugriff

Eine Amazon Glacier-Richtlinie für den Tresorzugriff ist eine ressourcenbasierte Richtlinie, mit der Sie die Berechtigungen für Ihren Tresor verwalten können.

Sie können eine Tresorzugriffsrichtlinie für jeden Tresor erstellen, um Berechtigungen zu verwalten. Sie können die Berechtigungen in einer Tresorzugriffsrichtlinie jederzeit ändern. Amazon Glacier unterstützt auch eine Vault Lock-Richtlinie für jeden Tresor, die nach dem Sperren nicht mehr geändert werden kann. Weitere Informationen zur Arbeit mit Tresorverriegelungs-Richtlinien finden Sie unter Tresorverriegelungs-Richtlinien.

Beispiele

- Beispiel 1: Kontenübergreifende Berechtigungen für bestimmte Amazon Glacier-Aktionen gewähren
- Beispiel 2: Berechtigungen über Konten hinweg für MFA-Löschoperationen erteilen

Beispiel 1: Kontenübergreifende Berechtigungen für bestimmte Amazon Glacier-Aktionen gewähren

Die folgende Beispielrichtlinie gewährt zwei Personen kontoübergreifende Berechtigungen AWS-Konten für eine Reihe von Amazon Glacier-Vorgängen in einem Tresor mit dem Namen`examplevault`.

> **Note**
>
> Dem Konto, das den Tresor besitzt, werden alle mit dem Tresor verbundenen Kosten in Rechnung gestellt. Alle Kosten für Anfragen, Datentransfer und Abruf, die von berechtigten externen Konten getätigt werden, werden dem Konto in Rechnung gestellt, das den Tresor besitzt.

Beispiel 2: Berechtigungen über Konten hinweg für MFA-Löschoperationen erteilen

Sie können die Multi-Faktor-Authentifizierung (MFA) verwenden, um Ihre Amazon Glacier-Ressourcen zu schützen. Um ein zusätzliches Maß an Sicherheit zu bieten, verlangt MFA, dass Benutzer den physischen Besitz eines MFA-Geräts anhand eines gültigen MFA-Codes nachweisen. Weitere Informationen zum Konfigurieren des MFA-Zugriffs finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Die Beispielrichtlinie gewährt einer Person AWS-Konto mit temporären Anmeldeinformationen die Erlaubnis, Archive aus einem Tresor namens examplevault zu löschen, vorausgesetzt, die Anfrage ist mit einem MFA-Gerät authentifiziert. Die Richtlinie verwendet den Bedingungsschlüssel `aws:MultiFactorAuthPresent`, um diese zusätzliche Anforderung anzugeben. Weitere Informationen finden Sie im Artikel zu verfügbaren Schlüsseln für Bedingungen im IAM-Benutzerhandbuch.

Tresorverriegelungs-Richtlinien

Einem Amazon Glacier (Amazon Glacier) -Tresor kann eine ressourcenbasierte Tresorzugriffsrichtlinie und eine Vault Lock-Richtlinie zugeordnet sein. Eine Tresorverriegelungs-Richtlinie ist eine Tresorzugriffsrichtlinie, die Sie sperren können. Die Verwendung einer Tresorverriegelungsrichtlinie kann Ihnen dabei helfen, Anforderungen für gesetzliche Vorschriften und Compliance durchzusetzen. Amazon Glacier bietet eine Reihe von API-Vorgängen, mit denen Sie die Vault Lock-Richtlinien verwalten können, siehe Sperren eines Tresors mithilfe der Amazon Glacier-API.

Nehmen wir als ein Beispiel für eine Tresorverriegelungs-Richtlinie an, dass Sie verpflichtet sind, Archive für ein Jahr aufzubewahren, bevor Sie sie löschen dürfen. Um diese Anforderung zu implementieren, können Sie eine Tresorverriegelungs-Richtlinie erstellen, die Benutzern die Berechtigung verweigert, ein Archiv zu löschen, wenn dieses Archiv noch nicht ein Jahr lang existiert. Sie können diese Richtlinie testen, bevor Sie sie sperren. Nachdem Sie die Richtlinie gesperrt haben, kann sie nicht mehr geändert werden. Weitere Informationen über den Verriegelungsprozess finden Sie unter Tresorverriegelungs-Richtlinien. Wenn Sie andere Benutzerberechtigungen verwalten möchten, die geändert werden können, können Sie die Tresorzugriffsrichtlinie verwenden (siehe Richtlinien für den Tresorzugriff).

Sie können die Amazon Glacier-API SDKs AWS CLI, Amazon oder die Amazon Glacier-Konsole verwenden, um Vault Lock-Richtlinien zu erstellen und zu verwalten. Eine Liste der Amazon Glacier-Aktionen, die für vault-ressourcenbasierte Richtlinien zulässig sind, finden Sie unter. Referenztabelle für -API-Berechtigungen

Beispiele

- Beispiel 1: Berechtigungen zum Löschen von Archiven, die weniger als 365 Tage alt sind, verweigern
- Beispiel 2: Berechtigungen zum Löschen basierend auf einem Tag verweigern

Beispiel 1: Berechtigungen zum Löschen von Archiven, die weniger als 365 Tage alt sind, verweigern

Angenommen, Sie haben eine gesetzliche Verpflichtung, Archive bis zu einem Jahr aufzubewahren, bevor Sie sie löschen können. Sie können diese Anforderung durchsetzen, indem Sie die folgende Tresorverriegelungs-Richtlinie implementieren. Die Richtlinie verweigert die Aktion `glacier:DeleteArchive` auf dem Tresor „examplevault“, wenn das zu löschende Archiv weniger als ein Jahr alt ist. Die Richtlinie verwendet den Amazon Glacier-spezifischen Bedingungsschlüssel, `ArchiveAgeInDays` um die einjährige Aufbewahrungspflicht durchzusetzen.

Beispiel 2: Berechtigungen zum Löschen basierend auf einem Tag verweigern

Angenommen, Sie haben eine zeitabhängige Aufbewahrungsregel, nach der ein Archiv gelöscht werden kann, wenn es weniger als ein Jahr alt ist. Nehmen Sie gleichzeitig an, dass Sie Ihre Archive rechtssicher aufbewahren müssen, um auf unbestimmte Zeit eine Löschung oder Änderung während einer gerichtlichen Untersuchung zu verhindern. In diesem Fall hat die Pflicht zur rechtssicheren Aufbewahrung Vorrang vor der zeitabhängigen Aufbewahrungsregel, die in der Tresorverriegelungs-Richtlinie spezifiziert wurde.

Um diese beiden Regeln in Kraft zu setzen, enthält die folgende Beispielrichtlinie zwei Anweisungen:

- Die erste Anweisung verweigert allen Benutzern die Berechtigung zum Löschen und verriegelt den Tresor. Diese Verriegelung erfolgt mithilfe des Tags `LegalHold`.
- Die zweite Anweisung gewährt Berechtigungen zum Löschen, wenn das Archiv weniger als 365 Tage alt ist. Aber selbst wenn Archive weniger als 365 Tage alt sind, kann niemand sie löschen, wenn die Bedingung in der ersten Anweisung erfüllt ist.

Problembehandlung bei Identität und Zugriff auf Amazon Glacier

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit Amazon Glacier und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion in Amazon Glacier durchzuführen
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Amazon Glacier-Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion in Amazon Glacier durchzuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `glacier:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 glacier:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `glacier:`*`GetWidget`*-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion durchzuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Amazon Glacier übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Service zu übergeben, anstatt eine neue Servicerolle oder eine dienstbezogene Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amazon Glacier auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Amazon Glacier-Ressourcen ermöglichen

Sie können eine Rolle erstellen, mit der Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation auf Ihre Ressourcen zugreifen können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob Amazon Glacier diese Funktionen unterstützt, finden Sie unterSo funktioniert Amazon Glacier mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Referenztabelle für -API-Berechtigungen

Wenn Sie So funktioniert Amazon Glacier mit IAM einrichten und eine Berechtigungsrichtlinie für eine IAM-Identität (identitätsbasierte Richtlinien) oder eine Ressource (ressourcenbasierte Richtlinien) verfassen, können Sie die folgende Tabelle als Referenz verwenden. Die enthält jeden Amazon Glacier-API-Vorgang, die entsprechenden Aktionen, für die Sie Berechtigungen zur Ausführung der Aktion erteilen können, und die AWS Ressource, für die Sie die Berechtigungen erteilen können.

Die Aktionen geben Sie im `Action`-Element und den Wert für die Ressource im `Resource`-Element der Richtlinie an. Außerdem können Sie mithilfe des `Condition`-Elements der IAM-Richtliniensyntax die Bedingungen angeben, unter denen die Richtlinie wirksam werden soll.

Um eine Aktion anzugeben, verwenden Sie das Präfix `glacier:` gefolgt vom Namen der API-Operation (z. B. `glacier:CreateVault`). Für die meisten Amazon Glacier-Aktionen `Resource` ist dies der Tresor, für den Sie die Berechtigungen erteilen möchten. Sie geben einen Tresor mit dem Tresor-ARN als `Resource`-Wert an. Bedingungen werden mithilfe vordefinierter Bedingungsschlüssel formuliert. Weitere Informationen finden Sie unter Ressourcenbasierte Richtlinien innerhalb von Amazon Glacier.

In der folgenden Tabelle werden Aktionen aufgeführt, die bei identitäts- und ressourcenbasierten Richtlinien verwendet werden können.

> **Note**
>
> Einige Aktionen können nur bei identitätsbasierten Richtlinien verwendet werden. Diese Aktionen werden mit einem Sternchen (*) nach dem Namen der API-Operation in der ersten Spalte gekennzeichnet.

Amazon Glacier-API und erforderliche Berechtigungen für Aktionen

Abort Multipart Upload (DELETE uploadID)

Erforderliche Berechtigungen (API-Aktionen): `glacier:AbortMultipartUpload`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Abort Vault Lock (DELETE lock-policy)

Erforderliche Berechtigungen (API-Aktionen): `glacier:AbortVaultLock`

Ressourcen:

Amazon Glacier-Zustandsschlüssel:

Add Tags To Vault (POST tags add)

Erforderliche Berechtigungen (API-Aktionen):`glacier:AddTagsToVault`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Complete Multipart Upload (POST uploadID)

Erforderliche Berechtigungen (API-Aktionen):`glacier:CompleteMultipartUpload`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Complete Vault Lock (POST lockId)

Erforderliche Berechtigungen (API-Aktionen):`glacier:CompleteVaultLock`

Ressourcen:

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Create Vault (PUT vault) *

Erforderliche Berechtigungen (API-Aktionen):`glacier:CreateVault`

Ressourcen:

Amazon Glacier-Zustandsschlüssel:

Delete Archive (DELETE archive)

Erforderliche Berechtigungen (API-Aktionen):`glacier:DeleteArchive`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:`glacier:ArchiveAgeInDays`, `glacier:ResourceTag/`*`TagKey`*

Delete Vault (DELETE vault)

Erforderliche Berechtigungen (API-Aktionen):`glacier:DeleteVault`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Delete Vault Access Policy (DELETE access-policy)

Erforderliche Berechtigungen (API-Aktionen):`glacier:DeleteVaultAccessPolicy`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Delete Vault Notifications (DELETE notification-configuration)

Erforderliche Berechtigungen (API-Aktionen):`glacier:DeleteVaultNotifications`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/TagKey`

Describe Job (GET JobID)

Erforderliche Berechtigungen (API-Aktionen):`glacier:DescribeJob`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Describe Vault (GET vault)

Erforderliche Berechtigungen (API-Aktionen):`glacier:DescribeVault`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Get Data Retrieval Policy (GET policy) *

Erforderliche Berechtigungen (API-Aktionen):`glacier:GetDataRetrievalPolicy`

Ressourcen: `arn:aws:glacier:region:account-id:policies/retrieval-limit-policy`

Amazon Glacier-Zustandsschlüssel:

Get Job Output (GET output)

Erforderliche Berechtigungen (API-Aktionen):`glacier:GetJobOutput`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Get Vault Access Policy (GET access-policy)

Erforderliche Berechtigungen (API-Aktionen):`glacier:GetVaultAccessPolicy`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Get Vault Lock (GET lock-policy)

Erforderliche Berechtigungen (API-Aktionen):`glacier:GetVaultLock`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Get Vault Notifications (GET notification-configuration)

Erforderliche Berechtigungen (API-Aktionen):`glacier:GetVaultNotifications`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:

Initiate Job (POST jobs)

Erforderliche Berechtigungen (API-Aktionen):`glacier:InitiateJob`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:`glacier:ArchiveAgeInDays`, `glacier:ResourceTag/`*`TagKey`*

Mehrteiligen Upload initiieren (POST multipart uploads)

Erforderliche Berechtigungen (API-Aktionen):`glacier:InitiateMultipartUpload`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/TagKey`

Initiate Vault Lock (POST lock-policy)

Erforderliche Berechtigungen (API-Aktionen):`glacier:InitiateVaultLock`

Ressourcen:

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/TagKey`

Aufträge auflisten (GET jobs)

Erforderliche Berechtigungen (API-Aktionen):`glacier:ListJobs`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel:

List Multipart Uploads (GET multipart-uploads)

Erforderliche Berechtigungen (API-Aktionen):`glacier:ListMultipartUploads`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel:

List Parts (GET uploadID)

Erforderliche Berechtigungen (API-Aktionen):`glacier:ListParts`

Ressourcen: `arn:aws:glacier:region:account-id:vaults/vault-name`, `arn:aws:glacier:region:account-id:vaults/example*`, `arn:aws:glacier:region:account-id:vaults/*`

Amazon Glacier-Zustandsschlüssel:

List Tags For Vault (GET tags)

Erforderliche Berechtigungen (API-Aktionen):`glacier:ListTagsForVault`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel:

List Vaults (GET vaults)

Erforderliche Berechtigungen (API-Aktionen):`glacier:ListVaults`

Ressourcen:

Amazon Glacier-Zustandsschlüssel:

Remove Tags From Vault (POST tags remove)

Erforderliche Berechtigungen (API-Aktionen):`glacier:RemoveTagsFromVault`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Set Data Retrieval Policy (PUT policy) *

Erforderliche Berechtigungen (API-Aktionen):`glacier:SetDataRetrievalPolicy`

Ressourcen:`arn:aws:glacier:`*`region`*`:`*`account-id`*`:policies/retrieval-limit-policy`

Amazon Glacier-Zustandsschlüssel:

Set Vault Access Policy (PUT access-policy)

Erforderliche Berechtigungen (API-Aktionen):`glacier:SetVaultAccessPolicy`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Set Vault Notification Configuration (PUT notification-configuration)

Erforderliche Berechtigungen (API-Aktionen):`glacier:SetVaultNotifications`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Upload Archive (POST archive)

Erforderliche Berechtigungen (API-Aktionen):`glacier:UploadArchive`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Upload Part (PUT uploadID)

Erforderliche Berechtigungen (API-Aktionen):`glacier:UploadMultipartPart`

Ressourcen: `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/vault-name`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/example*`, `arn:aws:glacier:`*`region`*`:`*`account-id`*`:vaults/*`

Amazon Glacier-Zustandsschlüssel: `glacier:ResourceTag/`*`TagKey`*

Protokollierung und Überwachung in Amazon Glacier

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon Glacier (Amazon Glacier) und Ihrer AWS Lösungen. Sie sollten Überwachungsdaten aus allen Teilen Ihrer AWS Lösung sammeln, damit Sie die Ursache eines Fehlers, falls einer auftritt, leichter identifizieren und debuggen können. AWS bietet die folgenden Tools zur Überwachung Ihrer Amazon Glacier-Ressourcen und zur Reaktion auf potenzielle Vorfälle:

CloudWatch Amazon-Alarme

Wenn Sie Amazon Glacier über Amazon S3 verwenden, können Sie Amazon CloudWatch Alarms verwenden, um eine einzelne Metrik über einen von Ihnen angegebenen Zeitraum zu beobachten. Wenn die Metrik einen bestimmten Schwellenwert überschreitet, wird eine Benachrichtigung an ein Amazon SNS SNS-Thema oder eine AWS Auto Scaling Richtlinie gesendet. CloudWatch Alarme lösen keine Aktionen aus, da sie sich in einem bestimmten Status befinden. Der Status muss sich stattdessen geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Monitoring Metrics with Amazon CloudWatch.

AWS CloudTrail Logs

CloudTrail bietet eine Aufzeichnung der Aktionen, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon Glacier ausgeführt wurden. CloudTrail erfasst alle API-Aufrufe für Amazon Glacier als Ereignisse, einschließlich Aufrufe von der Amazon Glacier-Konsole und von Codeaufrufen an Amazon Glacier APIs. Weitere Informationen finden Sie unter Protokollieren von Amazon Glacier-API-Aufrufen mit AWS CloudTrail.

AWS Trusted Advisor

Trusted Advisor stützt sich auf bewährte Verfahren, die wir bei der Betreuung von Hunderttausenden von AWS Kunden gelernt haben. Trusted Advisor untersucht Ihre AWS Umgebung und gibt dann Empfehlungen, wenn Möglichkeiten bestehen, Geld zu sparen, die Systemverfügbarkeit und -leistung zu verbessern oder Sicherheitslücken zu schließen. Alle AWS Kunden haben Zugriff auf fünf Trusted Advisor Schecks. Kunden mit einem Business- oder Enterprise-Supportplan können alle Trusted Advisor Schecks einsehen.

Weitere Informationen finden Sie unter AWS Trusted Advisor im Support -Benutzerhandbuch.

Konformitätsprüfung für Amazon Glacier

Die Sicherheit und Konformität von Amazon Glacier (Amazon Glacier) wird von externen Prüfern im Rahmen mehrerer AWS Compliance-Programme bewertet, darunter die 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 unter AWS Services in Scope by Compliance Program eine häufig aktualisierte Liste der AWS Services im Rahmen bestimmter Compliance-Programme.

Prüfberichte von Drittanbietern stehen Ihnen zum Herunterladen zur Verfügung unter AWS Artifact. Weitere Informationen finden Sie im AWS Artifact Benutzerhandbuch unter Herunterladen von Berichten in AWS Artifact.

Weitere Informationen zu AWS Compliance-Programmen finden Sie unter AWS Compliance-Programme.

Ihre Compliance-Verantwortung bei der Nutzung von Amazon Glacier richtet sich nach der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften. Falls Ihre Nutzung von Amazon Glacier der Einhaltung von Standards wie HIPAA, PCI oder FedRAMP unterliegt, AWS bietet Ihnen folgende Ressourcen:

- Amazon Glacier Tresorschlossermöglicht Ihnen die einfache Implementierung und Durchsetzung von Compliance-Kontrollen für einzelne Amazon Glacier-Tresore mit einer Tresorsperrrichtlinie. Sie können in einer Tresorverriegelungs-Richtlinie beispielsweise eine WORM-Bedingung (write once, read many) festlegen und die Richtlinie vor zukünftigen Änderungen schützen. Nachdem die Richtlinie gesperrt ist, kann sie nicht mehr geändert werden. Vault Lock-Richtlinien können Ihnen dabei helfen, gesetzliche Rahmenbedingungen wie SEC17a -4 und HIPAA einzuhalten.
- In den Schnellstartanleitungen zu Sicherheit und Compliance werden architektonische Überlegungen und Schritte zur Implementierung von sicherheits- und Compliance-orientierten Basisumgebungen erörtert. AWS
- Architecting for HIPAA Security and Compliance beschreibt, wie Unternehmen sie dabei unterstützen, die HIPAA-Anforderungen AWS zu erfüllen.

- Das AWS Well-Architected Tool (AWS WA Tool) ist ein Service in der Cloud, der Ihnen einen konsistenten Prozess zur Überprüfung und Messung Ihrer Architektur anhand von AWS Best Practices bietet. Das AWS WA Tool bietet Empfehlungen, wie Sie Ihre Workloads zuverlässiger, sicherer, effizienter und kostengünstiger gestalten können.
- AWS Die Compliance-Ressourcen bieten verschiedene Arbeitsmappen und Leitfäden, die möglicherweise auf Ihre Branche und Ihren Standort zutreffen.
- AWS Config hilft Ihnen bei der Bewertung, zu welchem Grad die Konfiguration Ihrer Ressourcen den internen Vorgehensweisen, Branchenrichtlinien und Vorschriften entspricht.
- AWS Security Hub bietet Ihnen einen umfassenden Überblick über Ihren Sicherheitsstatus AWS und hilft Ihnen dabei, Ihre Einhaltung der Sicherheitsstandards und Best Practices der Sicherheitsbranche zu überprüfen.

Resilienz in Amazon Glacier

Die AWS globale Infrastruktur basiert auf 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. Amazon Glacier speichert Daten redundant auf mehreren Geräten, die sich über mindestens drei Availability Zones erstrecken. Um die Haltbarkeit zu erhöhen, speichert Amazon Glacier Ihre Daten synchron über mehrere, AZs bevor ein erfolgreicher Upload bestätigt wird.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Infrastruktursicherheit in Amazon Glacier

Als verwalteter Service ist Amazon Glacier (Amazon Glacier) durch die AWS globalen Netzwerksicherheitsverfahren geschützt, die in Amazon Web Services: Überblick über Sicherheitsprozesse beschrieben sind.

Der Zugriff auf Amazon Glacier über das Netzwerk erfolgt über AWS Published APIs. Clients müssen Transport Layer Security (TLS) 1.2 unterstützen. Wir empfehlen TLS 1.3 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 Systeme wie Java 7 und höher unterstützen diese Modi. Darüber hinaus müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert werden, der einem -Prinzipal zugeordnet ist. Sie können auch AWS -Security-Token-Service (AWS STS) verwenden, um temporäre Sicherheitsanmeldeinformationen zur Signierung von Anforderungen zu generieren.

VPC-Endpunkte

Ein Virtual Private Cloud (VPC) -Endpunkt ermöglicht es Ihnen, Ihre VPC privat mit unterstützten AWS Diensten und VPC-Endpunktdiensten zu verbinden, die bereitgestellt werden, AWS PrivateLink ohne dass ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder eine Verbindung erforderlich ist. AWS Direct Connect Obwohl Amazon Glacier VPC-Endpunkte nicht direkt unterstützt, können Sie Amazon S3-VPC-Endpunkte nutzen, wenn Sie auf Amazon Glacier als in Amazon S3 integrierte Speicherebene zugreifen.

Weitere Informationen zur Amazon S3 S3-Lebenszykluskonfiguration und zur Übertragung von Objekten in die Amazon Glacier-Speicherklasse finden Sie unter Object Lifecycle Management and Transitioning Objects im Amazon Simple Storage Service-Benutzerhandbuch. Weitere Informationen zu VPC-Endpunkten finden Sie unter VPC-Endpunkte im Amazon-VPC-Benutzerhandbuch.

Amazon Glacier-Richtlinien für den Datenabruf

Mit den Datenabrufrichtlinien von Amazon Glacier können Sie auf einfache Weise Datenabrufkontingente festlegen und die Datenabrufaktivitäten für jeden von Ihnen AWS-Konto verwalten. AWS-Region Weitere Informationen zu den Gebühren für den Datenabruf von Amazon Glacier finden Sie unter Amazon Glacier-Preise.

> **Important**
>
> Eine Datenabruf-Richtlinie gilt nur für Standard-Abrufe und verwaltet Abrufanforderungen, die direkt an Amazon Glacier gestellt werden.
> Weitere Informationen zu den Amazon Glacier-Speicherklassen finden Sie unter Speicherklassen für die Archivierung von Objekten und Transitioning Objects im Amazon Simple Storage Service-Benutzerhandbuch.

Themen

- Auswahl einer Amazon Glacier-Richtlinie für den Datenabruf
- Verwenden der Amazon Glacier-Konsole zum Einrichten einer Richtlinie für den Datenabruf
- Verwenden der Amazon Glacier-API zum Einrichten einer Datenabruf-Richtlinie

Auswahl einer Amazon Glacier-Richtlinie für den Datenabruf

Sie können zwischen drei Arten von Amazon Glacier-Richtlinien für den Datenabruf wählen: Kein Abruflimit, Nur kostenloses Kontingent und Max. Abrufrate.

„Kein Abruflimit“ ist die Standardrichtlinie für Datenabrufe. Wenn Sie die Richtlinie „Kein Abruflimit“ verwenden, wird kein Abrufkontingent festgelegt und alle gültigen Datenabrufanforderungen werden akzeptiert.

Wenn Sie ein kostenloses Kontingent nutzen, können Sie Ihre Abrufe innerhalb Ihres täglichen AWS kostenlosen Kontingents halten und es fallen keine Kosten für den Datenabruf an. Wenn Sie mehr Daten abrufen möchten, als in Ihrem AWS kostenlosen Kontingent enthalten sind, können Sie im Rahmen der Richtlinie „Max. Abrufrate“ ein Kontingent für die maximale Abrufrate festlegen. bytes-per-hour Die Richtlinie „Maximale Abrufrate“ stellt sicher, dass die maximale Abrufrate aller

Abrufaufträge in Ihrem Konto das von Ihnen festgelegte Kontingent nicht AWS-Region überschreitet. bytes-per-hour

Sowohl bei Verwendung der Richtlinie „Nur kostenloses Kontingent“ als auch der Richtlinie „Maximale Abrufrate“ werden Datenabrufanforderungen, durch die die angegebenen Abruflimits überschritten würden, nicht akzeptiert. Wenn Sie ein kostenloses Kontingent verwenden, lehnt Amazon Glacier synchron Abrufanfragen ab, die Ihr AWS kostenloses Kontingent überschreiten. Wenn Sie eine Richtlinie für die maximale Abrufrate verwenden, lehnt Amazon Glacier Abrufanforderungen ab, die dazu führen, dass die maximale Abrufrate der laufenden Aufträge das in der Richtlinie festgelegte bytes-per-hour Kontingent überschreitet. Diese Richtlinien erleichtern die Verwaltung der Datenabrufkosten.

Es folgen einige nützliche Fakten zu den Richtlinien für den Datenabruf:

- Die Richtlinieneinstellungen für den Datenabruf ändern nicht den Zeitraum von 3 bis 5 Stunden, der benötigt wird, um Daten von Amazon Glacier mithilfe von Standardabrufen abzurufen.
- Die Festlegung einer neuen Richtlinie für den Datenabruf wirkt sich nicht auf zuvor akzeptierte Abrufaufträge aus, die bereits in Bearbeitung sind.
- Wenn eine Anforderung für einen Abrufauftrag wegen einer Richtlinie für den Datenabruf abgelehnt wird, werden Ihnen weder der Auftrag noch die Anforderung in Rechnung gestellt.
- Sie können jeweils eine Richtlinie für den Datenabruf festlegen AWS-Region, die alle Datenabrufaktivitäten unter Ihrem Konto regelt. AWS-Region Eine Richtlinie für den Datenabruf ist spezifisch für ein bestimmtes Thema, AWS-Region da die Kosten für den Datenabruf von Person zu Person unterschiedlich sind. AWS-Regionen Weitere Informationen finden Sie unter Amazon Glacier-Preise.

Richtlinie „Free Tier Only”

Sie können als Richtlinie für den Datenabruf die Option „Nur kostenloses Kontingent“ festlegen, um sicherzustellen, dass Ihre Datenabrufe stets im Rahmen Ihres AWS kostenlosen Kontingents bleiben, sodass Ihnen keine Gebühren für den Datenabruf entstehen. Wenn eine Abrufanforderung zurückgewiesen wird, erhalten Sie eine Fehlermeldung mit dem Hinweis, dass die Anforderung aufgrund der aktuellen Richtlinie für den Datenabruf abgelehnt wurde.

Sie können die Richtlinie für den Datenabruf für jede Region auf „Nur kostenloses Kontingent“ festlegen. Sobald die Richtlinie festgelegt wurde, können Sie an einem Tag nicht mehr Daten abrufen, als Ihnen für den Abruf im Rahmen des kostenlosen AWS -Kontingents für diese AWS-

Region pro Tag anteilig zur Verfügung steht. Außerdem fallen für Sie keine Gebühren für den Datenabruf an.

Sie können auch zur Richtlinie „Nur kostenloses Kontingent“ wechseln, nachdem Sie in einem Monat Kosten für den Datenabruf verursacht haben. Die Richtlinie „Nur kostenloses Kontingent“ wirkt sich dann nur auf neue Abrufanforderungen aus, nicht auf die bereits erfolgten Anforderungen. Die zuvor entstandenen Kosten werden Ihnen in Rechnung gestellt.

Richtlinie „Max Retrieval Rate”

Sie können Ihre Datenabruf-Richtlinie auf Max. Abrufrate festlegen, um die maximale Abrufrate zu kontrollieren, indem Sie ein Datenabrufkontingent angeben, das einen Höchstwert hat. bytes-per-hour Wenn Sie die Datenabruf-Richtlinie auf Max. Abrufrate setzen, wird eine neue Abrufanforderung zurückgewiesen, wenn dadurch die Spitzenabrufrate der laufenden Jobs das in der Richtlinie festgelegte Kontingent überschreiten würde. bytes-per-hour Wenn eine Abrufanforderung abgelehnt wird, erhalten Sie eine Fehlermeldung mit dem Hinweis, dass die Anforderung aufgrund der aktuellen Richtlinie für den Datenabruf zurückgewiesen wurde.

Wenn Sie als Richtlinie für den Datenabruf die maximale Abrufrate festlegen, kann sich dies darauf auswirken, wie viel von Ihrem im Rahmen des AWS kostenlosen Kontingents zugewiesenen Betrag Sie an einem Tag nutzen können. Angenommen, Sie legen „Max Retrieval Rate” z. B. auf 1 MB pro Stunde fest. Das ist weniger als der Tarif für das AWS kostenlose Kontingent. Um sicherzustellen, dass Sie das Tageslimit für das AWS kostenlose Kontingent optimal nutzen, können Sie Ihre Versicherung zunächst auf „Nur kostenloses Kontingent“ setzen und dann, falls erforderlich, zu einem späteren Zeitpunkt zum Tarif mit maximalem Abruftarif wechseln. Weitere Informationen zur Berechnung Ihres Abrufkontingents finden Sie unter Amazon Glacier FAQs.

Keine Richtlinie für den Abruf von Daten

Wenn Sie Ihre Richtlinie für den Datenabruf auf „Kein Abruflimit“ festgelegt haben, werden alle gültigen Datenabrufanforderungen akzeptiert. Die Kosten für den Datenabruf hängen von Ihrer Nutzung ab.

Verwenden der Amazon Glacier-Konsole zum Einrichten einer Richtlinie für den Datenabruf

So erstellen Sie eine Richtlinie zum Datenabruf mithilfe der Amazon Glacier-Konsole

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Glacier-Konsole zu https://console.aws.amazon.com/glacier/Hause.
2. Wählen Sie unter Region auswählen eine AWS-Region aus dem Drop-down-Menü aus. Sie können für jede Methode eine Richtlinie zum Datenabruf konfigurieren. AWS-Region
3. Wählen Sie im Navigationsbereich auf der linken Seite Datenabruf-Einstellungen aus.
4. Wählen Sie Bearbeiten aus. Die Seite Datenabruf-Richtlinien bearbeiten wird angezeigt.
5. Wählen Sie unter Datenabruf-Richtlinien eine Richtlinie aus.

   Sie können eine der drei Richtlinien für den Datenabruf auswählen: Kein Abruflimit, Nur kostenloses Kontingent oder Eine maximale Abrufrate angeben.

   - Wenn Sie Kein Abruflimit auswählen, werden alle gültigen Datenabrufanforderungen akzeptiert.
   - Wenn Sie nur das kostenlose Kontingent wählen, werden Datenabrufanfragen, die das AWS kostenlose Kontingent überschreiten, nicht akzeptiert.
   - Wenn Sie Eine maximale Abrufrate angeben auswählen, werden Datenabrufanforderungen zurückgewiesen, wenn sie dazu führen würden, dass die maximale Abrufrate der laufenden Aufträge die von Ihnen angegebene maximale Abrufrate übersteigt. Sie müssen im Feld GB/Stunde unter Maximale Abrufrate einen Wert für Gigabyte (GB) pro Stunde angeben. Sobald Sie einen Wert in das Feld GB/Stunde eingeben, berechnet die Konsole die geschätzten Kosten für Sie.
6. Wählen Sie Änderungen speichern aus.

Verwenden der Amazon Glacier-API zum Einrichten einer Datenabruf-Richtlinie

Sie können eine Richtlinie zum Datenabruf anzeigen und einrichten, indem Sie die Amazon Glacier REST-API oder die AWS SDKs verwenden.

Verwenden der Amazon Glacier REST-API zum Einrichten einer Datenabruf-Richtlinie

Mithilfe der Amazon Glacier REST-API können Sie eine Richtlinie für den Datenabruf anzeigen und festlegen. Mit der Get Data Retrieval Policy (GET policy)-Operation können Sie eine vorhandene Richtlinie für den Datenabruf anzeigen. Zum Einrichten einer Richtlinie für den Datenabruf verwenden Sie die Set Data Retrieval Policy (PUT policy)-Operation.

Bei Verwendung der PUT-Richtlinienoperation wählen Sie den Richtlinientyp für den Datenabruf durch Festlegen des JSON-Felds `Strategy` auf den Wert `BytesPerHour`, `FreeTier` bzw. `None` aus. `BytesPerHour` entspricht der Auswahl von Eine maximale Abrufrate angeben in der Konsole, `FreeTier` entspricht der Auswahl von Nur kostenloses Kontingent und `None` entspricht der Auswahl von Kein Abruflimit.

Wenn Sie mit der Initiate Job (POST jobs)-Operation einen Datenabrufauftrag initiieren, der die in der Richtlinie für den Datenabruf festgelegte maximale Abrufrate übersteigt, wird die `Initiate Job`-Operation beendet und eine Ausnahme ausgelöst.

Verwenden von AWS SDKs So richten Sie eine Datenabruf-Richtlinie ein

AWS bietet Ihnen SDKs die Möglichkeit, Anwendungen für Amazon Glacier zu entwickeln. Diese SDKs bieten Bibliotheken, die der zugrunde liegenden REST-API zugeordnet sind, und stellen Objekte bereit, mit denen Sie auf einfache Weise Anfragen erstellen und Antworten verarbeiten können. Weitere Informationen finden Sie unter Verwenden von AWS SDKs mit Amazon Glacier.

Taggen von Amazon Glacier-Ressourcen

Ein Tag ist eine Bezeichnung, die Sie einer AWS Ressource zuweisen. Jedes Tag besteht aus einem Schlüssel und einem Wert, die Sie beide selbst definieren. Sie können die von Ihnen definierten Tags Amazon Glacier (Amazon Glacier) Tresorressourcen zuweisen. Die Verwendung von Tags ist eine einfache und dennoch leistungsstarke Möglichkeit, AWS Ressourcen zu verwalten und Daten, einschließlich Rechnungsdaten, zu organisieren.

Themen

- Grundlagen des Markierens
- Tag-Einschränkungen
- Kosten mithilfe von Tags verfolgen
- Verwalten der Zugriffskontrolle mit Markieren
- Zugehörige Abschnitte

Grundlagen des Markierens

Sie verwenden die Amazon Glacier-Konsole AWS Command Line Interface (AWS CLI) oder die Amazon Glacier-API, um die folgenden Aufgaben auszuführen:

- Hinzufügen von Tags zu einem Tresor
- Auflisten der Tags für einen Tresor
- Entfernen von Tags von einem Tresor

Informationen darüber, wie Sie Tags hinzufügen, auflisten und entfernen, finden Sie unter Kennzeichnen Ihrer Amazon Glacier-Tresore.

Sie können Tags verwenden, um Ihre Tresore zu kategorisieren. Sie können Tresore beispielsweise nach Zweck, Inhaber oder Umgebung kategorisieren. Da Sie für jeden Tag den Schlüssel und Wert definieren, können Sie eine auf benutzerdefinierte Reihe von Kategorien anlegen, die Ihren jeweiligen Anforderungen gerecht wird. Sie könnten zum Beispiel eine Reihe von Tags definieren, mit der Sie Tresore nach Inhaber und Zweck für den Tresor nachverfolgen können. Im Folgenden sind einige Beispiele für Tags aufgeführt:

- Inhaber: Name

- Zweck: Videoarchive
- Umgebung: Produktion

Tag-Einschränkungen

Grundlegende Tag-Beschränkungen sind wie folgt:

- Die maximale Anzahl an Tags pro Ressource (Tresor) beträgt 50.
- Bei Tag-Schlüsseln und -Werten muss die Groß-/Kleinschreibung beachtet werden.

Tag-Schlüssel-Beschränkungen sind wie folgt:

- Innerhalb eines Satzes von Tags für einen Tresor muss jeder Tag-Schlüssel eindeutig sein. Wenn Sie einen Tag mit einem Schlüssel hinzufügen, der bereits verwendet wird, wird das vorhandene Schlüssel-Wert-Paar durch den neuen Tag überschrieben.
- Tag-Schlüssel können nicht mit beginnen`aws:`, da dieses Präfix für die Verwendung durch reserviert ist AWS. AWS kann in Ihrem Namen Tags erstellen, die mit diesem Präfix beginnen, aber Sie können sie nicht bearbeiten oder löschen.
- Tag-Schlüssel müssen zwischen 1 und 128 Unicode-Zeichen lang sein.
- Tag-Schlüssel müssen die folgenden Zeichen enthalten: Unicode-Zeichen, Ziffern, Leerzeichen sowie die folgenden Sonderzeichen: `_ . / = + - @`.

Tag-Wert-Beschränkungen sind wie folgt:

- Tag-Werte müssen zwischen 0 und 255 Unicode-Zeichen lang sein.
- Tag-Werte können leer sein. Ansonsten müssen sie die folgenden Zeichen enthalten: Unicode-Zeichen, Ziffern, Leerzeichen und die folgenden Sonderzeichen: `_ . / = + - @`.

Kosten mithilfe von Tags verfolgen

Sie können Tags verwenden, um Ihre AWS Kosten zu kategorisieren und nachzuverfolgen. Wenn Sie Tags auf AWS Ressourcen, einschließlich Tresoren, anwenden, enthält Ihr AWS Kostenzuordnungsbericht die Nutzung und die Kosten, die nach Stichwörtern zusammengefasst sind. Sie können Tags anwenden, die geschäftliche Kategorien (wie Kostenstellen, Anwendungsnamen und Eigentümer) darstellen, um die Kosten für mehrere Services zu organisieren. Weitere

Informationen finden Sie unter Verwenden von Kostenzuordnungs-Tags für benutzerdefinierte Fakturierungsberichte im AWS Billing -Benutzerhandbuch.

Verwalten der Zugriffskontrolle mit Markieren

Sie können Tags als eine Bedingung in einer Zugriffsrichtlinienanweisung verwenden. Zum Beispiel können Sie einen Tag zur gesetzlichen Aufbewahrung einrichten und ihn als Bedingung in eine Datenaufbewahrungsrichtlinie einfügen, die bestimmt, dass „Löschen des Archivs durch jeden abgelehnt wird, wenn der Wert des Tags zur gesetzlichen Aufbewahrung auf `True` festgelegt ist.“ Sie können die Datenaufbewahrungsrichtlinie bereitstellen und den Tag zur gesetzlichen Aufbewahrung unter normalen Bedingungen auf `False` festlegen. Wenn Ihre Daten in Warteschleife gesetzt werden, um eine Untersuchung zu unterstützen, können Sie die gesetzliche Aufbewahrung einfach einschalten, indem Sie den Wert des Tags auf `True` festlegen und diese Aufbewahrung später auf ähnliche Weise wieder entfernen. Weitere Informationen finden Sie unter Zugriffssteuerung mit Tags im -IAM-Benutzerhandbuch.

Zugehörige Abschnitte

- Kennzeichnen Ihrer Amazon Glacier-Tresore

Protokollieren von Amazon Glacier-API-Aufrufen mit AWS CloudTrail

Amazon Glacier (Amazon Glacier) ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon Glacier ausgeführt wurden. CloudTrail erfasst alle API-Aufrufe für Amazon Glacier als Ereignisse, einschließlich Aufrufe von der Amazon Glacier-Konsole und von Codeaufrufen an Amazon Glacier APIs. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für Amazon Glacier. Auch wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail -Konsole in Event history (Ereignisverlauf) anzeigen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon Glacier gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

Informationen zu Amazon Glacier in CloudTrail

CloudTrail ist auf Ihrem aktiviert AWS-Konto , wenn Sie das Konto erstellen. Wenn in Amazon Glacier Aktivitäten auftreten, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem anzeigen, suchen und herunterladen AWS-Konto. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, einschließlich Ereignissen für Amazon Glacier, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole erstellen, gilt der Trail standardmäßig für alle AWS Regionen. Der Trail protokolliert Ereignisse aus allen AWS Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie unter:

- Übersicht zum Erstellen eines Trails
- CloudTrail Unterstützte Dienste und Integrationen

- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien von mehreren Konten

Alle Amazon Glacier-Aktionen werden von protokolliert CloudTrail und sind in der dokumentiertAPI-Referenz für Amazon Glacier. Beispielsweise generieren Aufrufe der List Vaults (GET vaults) AktionenCreate Vault (PUT vault),Delete Vault (DELETE vault), und Einträge in den CloudTrail Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anforderung mit Root-Benutzer- oder anderen Anmeldeinformation ausgeführt wurde.
- Ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen verbundenen Benutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter CloudTrail userIdentity-Element.

Grundlegendes zu Amazon Glacier-Protokolldateieinträgen

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion, Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die Describe Vault (GET vault) AktionenCreate Vault (PUT vault), Delete Vault (DELETE vault)List Vaults (GET vaults), und demonstriert.

```
{
    "Records": [
        {
            "awsRegion": "us-east-1",
            "eventID": "52f8c821-002e-4549-857f-8193a15246fa",
            "eventName": "CreateVault",
```

```
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "HJiLgvfXCY88QJAC6rRoexS9ThvI21Q1Nqukfly02hcUPPo",
            "requestParameters": {
                "accountId": "-",
                "vaultName": "myVaultName"
            },
            "responseElements": {
                "location": "/999999999999/vaults/myVaultName"
            },
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        },
        {
            "awsRegion": "us-east-1",
            "eventID": "cdd33060-4758-416a-b7b9-dafd3afcec90",
            "eventName": "DeleteVault",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "GGdw-VfhVfLCFwAM6iVUvMQ6-fMwSqSO9FmRd0eRSa_Fc7c",
            "requestParameters": {
                "accountId": "-",
                "vaultName": "myVaultName"
            },
            "responseElements": null,
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
```

```
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        },
        {
            "awsRegion": "us-east-1",
            "eventID": "355750b4-e8b0-46be-9676-e786b1442470",
            "eventName": "ListVaults",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "yPTs22ghTsWprFivb-2u30FAaDALIZP17t4jM_xL9QJQyVA",
            "requestParameters": {
                "accountId": "-"
            },
            "responseElements": null,
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        },
        {
            "awsRegion": "us-east-1",
            "eventID": "569e830e-b075-4444-a826-aa8b0acad6c7",
            "eventName": "DescribeVault",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "QRt1ZdFLGn0TCm784HmKafBmcB2lVaV81UU3fsOR3PtoIiM",
```

```
            "requestParameters": {
                "accountId": "-",
                "vaultName": "myVaultName"
            },
            "responseElements": null,
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        }
    ]
}
```

API-Referenz für Amazon Glacier

Amazon Glacier unterstützt eine Reihe von Vorgängen — insbesondere eine Reihe von RESTful API-Aufrufen —, die es Ihnen ermöglichen, mit dem Service zu interagieren.

Sie können jede Programmierbibliothek verwenden, die HTTP-Anfragen senden kann, um Ihre REST-Anfragen an Amazon Glacier zu senden. Beim Senden einer REST-Anfrage verlangt Amazon Glacier, dass Sie jede Anfrage authentifizieren, indem Sie die Anfrage signieren. Außerdem müssen Sie bei Upload eines Archivs die Prüfsumme der Nutzlast berechnen und in Ihrer Anfrage einschließen. Weitere Informationen finden Sie unter Signieren von Anforderungen.

Wenn ein Fehler auftritt, müssen Sie wissen, was Amazon Glacier in einer Fehlerantwort sendet, damit Sie ihn verarbeiten können. In diesem Abschnitt werden zusätzlich zur Dokumentation der REST-Operationen alle diese Informationen bereitgestellt, damit Sie REST-API-Aufrufe direkt durchführen können.

Sie können entweder die REST-API-Aufrufe direkt verwenden oder Amazon verwenden SDKs , das Wrapper-Bibliotheken bereitstellt. Diese Bibliotheken signieren jede von Ihnen gesendete Anfrage und berechnen die Prüfsumme der Nutzlast in Ihrer Anfrage. Daher SDKs vereinfacht die Verwendung von Amazon Ihre Codierungsaufgabe. Dieses Entwicklerhandbuch enthält praktische Beispiele für grundlegende Amazon Glacier-Operationen unter Verwendung AWS SDK für Java von.NET. Weitere Informationen finden Sie unter Verwenden von AWS SDKs mit Amazon Glacier.

Themen

- Häufig verwendete Anforderungsheader
- Allgemeine Antwort-Header
- Signieren von Anforderungen
- Berechnen von Prüfsummen
- Fehlermeldungen
- Tresor-Operationen
- Archivoperationen
- Operationen für mehrteiligen Upload
- Auftragsoperationen
- In Auftragsoperationen verwendete Datentypen

- Datenabrufoperationen

Häufig verwendete Anforderungsheader

REST-Anfragen von Amazon Glacier (Amazon Glacier) enthalten Header, die grundlegende Informationen über die Anfrage enthalten. In der folgenden Tabelle werden Header beschrieben, die von allen Amazon Glacier REST-Anfragen verwendet werden können.

Header-Name	Beschreibung	Erforderlich
`Authorization`	Der Header, der zum Signieren von Anforderungen erforderlich ist. Amazon Glacier benötigt Signature Version 4. Weitere Informationen finden Sie unter Signieren von Anforderungen. Typ: Zeichenfolge	Ja
`Content-Length`	Die Länge des Anforderungstextes (ohne Header). Typ: Zeichenfolge Bedingung: Nur für die Upload Archive (POST archive)-API erforderlich.	Bedingt
`Date`	Das Datum, an dem die im `Authorization` -Header enthaltene Signatur erstellt werden kann. Wenn der `Date`-Header zum Signieren verwendet werden soll, muss das Format den Grundsätzen von ISO 8601 entsprechen In diesem Fall wird der `x-amz-date` -Header nicht benötigt. Wenn `x-amz-date` vorhanden ist, hat dieser Wert immer Vorrang vor dem Wert des `Date`-Headers. Wenn der Datum-Header nicht zum Signieren verwendet wird, kann er in einem der vollständigen Datumsformate angegeben werden, die durch RFC 2616, Abschnitt 3.3, festgelegt sind. Das folgende Datum/die folgende Uhrzeit `Wed, 10 Feb 2017`	Bedingt

Header-Name	Beschreibung	Erforderlich
	`12:00:00 GMT` ist beispielsweise ein gültiger date/time Header für die Verwendung mit Amazon Glacier. Wenn der `Date`-Header zum Signieren verwendet wird, muss er den Grundsätzen von ISO 8601 entsprechen und im Format `YYYYMMDD'T'HHMMSS'Z'` angegeben werden. Typ: Zeichenfolge Bedingung: Wenn `Date` zwar angegeben ist, aber nicht das ISO 8601-Grundformat aufweist, muss auch der `x-amz-date` -Header vorhanden sein. Wenn `Date` entsprechend den Grundsätzen von ISO 8601 angegeben ist, reicht diese Angabe zum Signieren von Anforderungen aus und der `x-amz-date` -Header ist nicht erforderlich. Weitere Informationen finden Sie unter Umgang mit Datumswerten in Signature Version 4 im Amazon Web Services-Glossar.	
`Host`	Dieser Header kennzeichnet den Service-Endpunkt, an den Sie Ihre Anforderungen senden. Der Wert muss die Form "`glacier.`*`region`*`.amazonaws.com` " haben, wobei er durch eine AWS Regionsbezeichnung wie `us-west-2` ersetzt *`region`* wird. Typ: Zeichenfolge	Ja

Header-Name	Beschreibung	Erforderlich
`x-amz-content-sha256`	Die berechnete SHA256 Prüfsumme einer gesamten Payload, die entweder mit oder hochgeladen wurde. Upload Archive (POST archive) Upload Part (PUT uploadID) Dieser Header stimmt nicht mit dem `x-amz-sha256-tree-hash` -Header überein, obwohl für einige kleine Nutzlasten unter Umständen dieselben Werte verwendet werden. Wenn `x-amz-content-sha256` erforderlich ist, muss sowohl `x-amz-content-sha256` als auch `x-amz-sha256-tree-hash` angegeben werden. Typ: Zeichenfolge Bedingung: Erforderlich für Streaming-API, Upload Archive (POST archive) und Upload Part (PUT uploadID).	Bedingt

Header-Name	Beschreibung	Erforderlich
`x-amz-date`	Das Datum, an dem die Signatur im Autorisierungs-Header erstellt wurde. Das Datum muss den Grundsätzen von ISO 8601 entsprechen und im Format `YYYYMMDD'T'HHMMSS'Z'` angegeben werden. Das Folgende date/time `20170210T120000Z` gilt beispielsweise `x-amz-date` für die Verwendung mit Amazon Glacier. Typ: Zeichenfolge Bedingung: `x-amz-date` ist bei allen Anforderungen optional. Damit kann das Datum überschrieben werden, das zum Signieren von Anforderungen verwendet wird. Wenn der `Date`-Header den Grundsätzen von ISO 8601 entspricht, ist `x-amz-date` nicht erforderlich. Wenn `x-amz-date` vorhanden ist, hat dieser Wert immer Vorrang vor dem Wert des `Date`-Headers. Weitere Informationen finden Sie unter Umgang mit Datumswerten in Signature Version 4 im Amazon Web Services-Glossar.	Bedingt
`x-amz-glacier-version`	Die zu verwendende Amazon Glacier-API-Version. Die aktuelle Version ist `2012-06-01` . Typ: Zeichenfolge	Ja

Header-Name	Beschreibung	Erforderlich
`x-amz-sha256-tree-hash`	Die berechnete SHA256 Tree-Hash-Prüfsumme für ein hochgeladenes Archiv (Upload Archive (POST archive)) oder einen Archivteil (). Upload Part (PUT uploadID) Weitere Informationen zum Berechnen dieser Prüfsumme finden Sie unter Berechnen von Prüfsummen. Typ: Zeichenfolge Standard: keiner Bedingung: Erforderlich für Upload Archive (POST archive) und Upload Part (PUT uploadID).	Bedingt

Allgemeine Antwort-Header

In der folgenden Tabelle werden Antwort-Header beschrieben, die für die meisten API-Antworten gelten.

Name	Beschreibung
`Content-Length`	Die Länge des Antworttexts in Bytes. Typ: Zeichenfolge
`Date`	Datum und Uhrzeit der Antwort von Amazon Glacier (Amazon Glacier), `Wed, 10 Feb 2017 12:00:00 GMT` z. B. Das Datumsformat muss einem der vollständigen Datumsformate folgen, die in RFC 2616, Abschnitt 3.3 angegeben sind. Beachten Sie, dass ein zurückgegebenes `Date` leicht von anderen Daten abweichen kann. So stimmt das von einem Upload Archive (POST archive) zurückgegebene Datum möglicherweise nicht mit dem für das Archiv in einer Bestandsliste für den Tresor angegebenen Datum überein. Typ: Zeichenfolge

Name	Beschreibung
`x-amzn-RequestId`	Ein von Amazon Glacier erstellter Wert, der Ihre Anfrage eindeutig identifiziert. Falls Sie ein Problem mit Amazon Glacier haben, AWS können Sie diesen Wert verwenden, um das Problem zu beheben. Es ist empfehlenswert, diese Werte zu protokollieren. Typ: Zeichenfolge
`x-amz-sha256-tree-hash`	Die SHA256 Tree-Hash-Prüfsumme des Archivs oder des Inventartextteils. Weitere Informationen zum Berechnen dieser Prüfsumme finden Sie unter Berechnen von Prüfsummen. Typ: Zeichenfolge

Signieren von Anforderungen

Amazon Glacier verlangt, dass Sie jede Anfrage, die Sie senden, authentifizieren, indem Sie die Anfrage signieren. Zum Signieren einer Anforderung berechnen Sie eine digitale Signatur mit einer kryptografischen Hash-Funktion. Ein kryptografischer Hash ist eine Funktion, die auf Grundlage der Eingabe einen einzigartigen Hash-Wert zurückgibt. Die Eingabe in die Hash-Funktion besteht aus dem Text Ihrer Anforderung und Ihrem geheimen Zugriffsschlüssel. Die Hash-Funktion gibt einen Hash-Wert zurück, den Sie in die Anforderung als Ihre Signatur einfügen. Die Signatur ist Teil des Headers `Authorization` in der Anforderung.

Nach Erhalt Ihrer Anfrage berechnet Amazon Glacier die Signatur mit derselben Hash-Funktion und Eingabe neu, mit der Sie die Anfrage signiert haben. Wenn die resultierende Signatur mit der Signatur in der Anfrage übereinstimmt, verarbeitet Amazon Glacier die Anfrage. Andernfalls wird die Anforderung abgelehnt.

Amazon Glacier unterstützt die Authentifizierung mit AWS Signature Version 4. Der Prozess zum Berechnen einer Signatur lässt sich in drei Aufgaben untergliedern:

- Aufgabe 1: Erstellen einer kanonischen Anforderung

  Ordnen Sie Ihre HTTP-Anforderung in einem kanonischen Format neu an. Die Verwendung eines kanonischen Formulars ist erforderlich, da Amazon Glacier dasselbe kanonische Formular

verwendet, wenn es eine Signatur neu berechnet, um sie mit der von Ihnen gesendeten zu vergleichen.

- Aufgabe 2: Erstellen einer zu signierenden Zeichenfolge

  Erstellen Sie eine Zeichenfolge, die Sie als einen der Eingabewerte für die kryptografische Hash-Funktion nutzen. Die als zu signierende Zeichenfolge bezeichnete Zeichenfolge ist eine Kombination aus dem Namen des Hash-Algorithmus, dem Anforderungsdatum, einer Zeichenfolge mit dem Umfang der Anmeldeinformationen und der kanonischen Anforderung aus der vorherigen Aufgabe. Die Zeichenfolge für den Gültigkeitsbereich der Anmeldeinformationen selbst ist eine Verkettung von Datums-, Regions- und Serviceinformationen. AWS

- Aufgabe 3: Erstellen einer Signatur

  Erstellen Sie eine Signatur für Ihre Anforderung. Verwenden Sie dazu eine kryptografische Hash-Funktion, die zwei Eingabezeichenfolgen akzeptiert: die zu signierende Zeichenfolge und einen abgeleiteten Schlüssel. Der abgeleitete Schlüssel wird berechnet, indem Sie mit Ihrem geheimen Zugriffsschlüssel beginnen und anhand der Zeichenfolge für den Gültigkeitsbereich der Anmeldeinformationen eine Reihe von Hash-basierten Nachrichtenauthentifizierungscodes erstellen (). HMACs Beachten Sie, dass es sich bei der in diesem Signierungsschritt verwendeten Hash-Funktion nicht um den Tree-Hash-Algorithmus handelt, der in Amazon Glacier APIs zum Hochladen von Daten verwendet wird.

Themen

- Signatur-Berechnungsbeispiel
- Berechnen von Signaturen für Streaming-Operationen

Signatur-Berechnungsbeispiel

Das folgende Beispiel macht Sie damit vertraut, wie Sie eine Signatur für Create Vault (PUT vault) erstellen. Das Beispiel kann als Referenz verwendet werden, um Ihre Signaturberechnungsmethode zu überprüfen. Weitere Informationen finden Sie unter Signieren von AWS API-Anfragen im IAM-Benutzerhandbuch.

In diesem Beispiel wird Folgendes angenommen:

- Der Zeitstempel der Anforderung ist `Fri, 25 May 2012 00:24:53 GMT`.
- Der Endpunkt ist die Region „USA Ost (Nord-Virginia)“ (`us-east-1`).

Die allgemeine Anforderungssyntax (einschließlich JSON-Text) ist:

```
PUT /-/vaults/examplevault HTTP/1.1
Host: glacier.us-east-1.amazonaws.com
Date: Fri, 25 May 2012 00:24:53 GMT
Authorization: SignatureToBeCalculated
x-amz-glacier-version: 2012-06-01
```

Die kanonische Form der für Aufgabe 1: Erstellen einer kanonischen Anforderung berechneten Anforderung ist:

```
PUT
/-/vaults/examplevault

host:glacier.us-east-1.amazonaws.com
x-amz-date:20120525T002453Z
x-amz-glacier-version:2012-06-01

host;x-amz-date;x-amz-glacier-version
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Die letzte Zeile der kanonischen Anforderungen ist der Hash des Anforderungstextes. Beachten Sie auch die leere dritte Zeile in der kanonischen Anforderung. Der Grund dafür ist, dass es keine Abfrageparameter für diese API gibt.

Die zu signierende Zeichenfolge für Aufgabe 2: Erstellen einer zu signierenden Zeichenfolge ist:

```
AWS4-HMAC-SHA256
20120525T002453Z
20120525/us-east-1/glacier/aws4_request
5f1da1a2d0feb614dd03d71e87928b8e449ac87614479332aced3a701f916743
```

Die erste Zeile der zu signierenden Zeichenfolge ist der Name des Algorithmus, die zweite Zeile der Zeitstempel, die dritte Zeile der Geltungsbereich der Anmeldeinformationen und die letzte Zeile ein Hash der kanonischen Anforderung aus Aufgabe 1: Erstellen einer kanonischen Anforderung. In dem Geltungsbereich für die Anmeldeinformationen ist auch u.a. der Name des zu verwenden Service angegeben, `glacier`.

Für Aufgabe 3: Erstellen einer Signatur kann der abgeleitete Schlüssel wie folgt dargestellt werden:

```
derived key = HMAC(HMAC(HMAC(HMAC("AWS4" + YourSecretAccessKey,"20120525"),"us-
east-1"),"glacier"),"aws4_request")
```

Wenn der geheime Zugriffsschlüssel `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY` verwendet wird, lautet die berechnete Signatur:

```
3ce5b2f2fffac9262b4da9256f8d086b4aaf42eba5f111c21681a65a127b7c2a
```

Der letzte Schritt besteht im Erstellen des `Authorization`-Headers. Für den Demo-Zugriffsschlüssel `AKIAIOSFODNN7EXAMPLE` (mit hinzugefügten Zeilenumbrüchen zur besseren Lesbarkeit) lautet der Header:

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20120525/us-east-1/
glacier/aws4_request,
SignedHeaders=host;x-amz-date;x-amz-glacier-version,
Signature=3ce5b2f2fffac9262b4da9256f8d086b4aaf42eba5f111c21681a65a127b7c2a
```

Berechnen von Signaturen für Streaming-Operationen

Upload Archive (POST archive) und Upload Part (PUT uploadID) sind Streaming-Operationen, für die Sie beim Signieren und Senden der Anforderung einen zusätzlichen Header `x-amz-content-sha256` angeben müssen. Die Signierschritte für die Streaming-Operationen sind mit den Schritten für andere Operationen identisch. Es muss lediglich ein Streaming-Header hinzugefügt werden.

Die Berechnung des Streaming-Headers `x-amz-content-sha256` basiert auf dem SHA256 Hash des gesamten Inhalts (Payload), der hochgeladen werden soll. Beachten Sie, dass sich diese Berechnung vom SHA256 Tree-Hash (Berechnen von Prüfsummen) unterscheidet. Abgesehen von trivialen Fällen unterscheidet sich der SHA-256-Hashwert der Nutzdaten vom SHA256 Baumhash der Nutzdaten.

Wenn die Nutzdaten als Byte-Array angegeben sind, können Sie den folgenden Java-Codeausschnitt verwenden, um den Hash zu berechnen. SHA256

```
public static byte[] computePayloadSHA256Hash2(byte[] payload) throws
 NoSuchAlgorithmException, IOException {
    BufferedInputStream bis =
       new BufferedInputStream(new ByteArrayInputStream(payload));
    MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
    byte[] buffer = new byte[4096];
    int bytesRead = -1;
    while ( (bytesRead = bis.read(buffer, 0, buffer.length)) != -1 ) {
        messageDigest.update(buffer, 0, bytesRead);
    }
    return messageDigest.digest();
}
```

In ähnlicher Weise können Sie in C# den SHA256 Hash der Nutzlastdaten berechnen, wie im folgenden Codeausschnitt gezeigt.

```
public static byte[] CalculateSHA256Hash(byte[] payload)
{
    SHA256 sha256 = System.Security.Cryptography.SHA256.Create();
    byte[] hash = sha256.ComputeHash(payload);

    return hash;
}
```

Signatur-Berechnungsbeispiel für die Streaming-API

Das folgende Beispiel führt Sie durch die Einzelheiten der Erstellung einer Signatur für Upload Archive (POST archive) einen der beiden Streaming-Dienste APIs in Amazon Glacier. In diesem Beispiel wird Folgendes angenommen:

- Der Zeitstempel der Anforderung ist `Mon, 07 May 2012 00:00:00 GMT`.
- Der Endpunkt ist die Region „USA Ost (Nord-Virginia)“ („us-east-1“).
- Die Inhaltsnutzlast ist die Zeichenfolge „Willkommen bei Amazon Glacier“.

Die allgemeine Anforderungssyntax (einschließlich JSON-Text) ist unten im Beispiel zu sehen. Beachten Sie, dass der Header `x-amz-content-sha256` eingeschlossen ist. In diesem vereinfachten Beispiel haben `x-amz-sha256-tree-hash` und `x-amz-content-sha256`

denselben Wert. Beim Hochladen von Archiven, die größer als 1 MB sind, ist dies jedoch nicht der Fall.

```
POST /-/vaults/examplevault HTTP/1.1
Host: glacier.us-east-1.amazonaws.com
Date: Mon, 07 May 2012 00:00:00 GMT
x-amz-archive-description: my archive
x-amz-sha256-tree-hash: SHA256 tree hash
x-amz-content-sha256: SHA256 payload hash
Authorization: SignatureToBeCalculated
x-amz-glacier-version: 2012-06-01
```

Die kanonische Form der Anforderung, die für Aufgabe 1: Erstellen einer kanonischen Anforderung berechnet wird, ist unten zu sehen. Beachten Sie, dass der Streaming-Header `x-amz-content-sha256` mit seinem Wert eingeschlossen ist. Das bedeutet, dass Sie zuerst die Payload lesen und den SHA256 Hash berechnen und dann die Signatur berechnen müssen.

```
POST
/-/vaults/examplevault

host:glacier.us-east-1.amazonaws.com
x-amz-content-sha256:726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
x-amz-date:20120507T000000Z
x-amz-glacier-version:2012-06-01

host;x-amz-content-sha256;x-amz-date;x-amz-glacier-version
726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
```

Der Rest der Signaturberechnung erfolgt entsprechend den unter Signatur-Berechnungsbeispiel beschriebenen Schritten. Der `Authorization`-Header mit dem geheimen Zugriffschlüssel `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY` und dem Zugriffschlüssel `AKIAIOSFODNN7EXAMPLE` (mit hinzugefügten Zeilenumbrüchen zur besseren Lesbarkeit) ist unten dargestellt:

```
Authorization=AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20120507/us-east-1/glacier/aws4_request,
SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-amz-glacier-version,
Signature=b092397439375d59119072764a1e9a144677c43d9906fd98a5742c57a2855de6
```

Berechnen von Prüfsummen

Bei einem Archiv-Upload müssen die Header `x-amz-sha256-tree-hash` und `x-amz-content-sha256` enthalten sein. Der `x-amz-sha256-tree-hash`-Header ist eine Prüfsumme der Nutzlast in Ihrem Anfragetext. In diesem Thema wird die Berechnung des `x-amz-sha256-tree-hash`-Headers beschrieben. Der `x-amz-content-sha256`-Header ist ein Hash der gesamten Nutzlast und ist für die Autorisierung erforderlich. Weitere Informationen finden Sie unter Signatur-Berechnungsbeispiel für die Streaming-API.

Die Nutzlast Ihrer Anfrage kann Folgendes sein:

- Gesamtes Archiv: Wenn Sie ein Archiv in einer einzelnen Anforderung mit der API „Upload Archive“ hochladen, wird das gesamte Archiv im Anforderungstext gesendet. In diesem Fall muss die Prüfsumme des gesamten Archivs enthalten sein.
- Archivteile: Wenn Sie ein Archiv mit der API „Multipart Upload“ in Teilen hochladen, senden Sie nur einen Teil des Archivs im Anforderungstext. In diesem Fall ist die Prüfsumme des Archivteils enthalten. Nachdem Sie alle Teile hochgeladen haben, senden Sie eine „Complete Multipart Upload“-Anfrage, in der die Prüfsumme des gesamten Archivs enthalten sein muss.

Die Prüfsumme für die Nutzlast ist ein SHA-256-Struktur-Hash. Dies wird als Struktur-Hash bezeichnet, da bei der Berechnung der Prüfsumme die Struktur eines SHA-256-Hash-Werts berechnet wird. Der Hash-Wert am Stamm ist die Prüfsumme des gesamten Archivs.

> **Note**
>
> In diesem Abschnitt wird eine Möglichkeit zur Berechnung des SHA-256-Struktur-Hashs beschrieben. Allerdings können Sie einen beliebigen Vorgang verwenden, bei dem das gleiche Ergebnis erzielt wird.

Der SHA-256-Struktur-Hash wird wie folgt berechnet:

1. Berechnen Sie den SHA-256-Hash für jeden 1 MB großen Teil der Nutzlastdaten. Der letzte Datenanteil kann kleiner als 1 MB sein. Wenn Sie beispielsweise ein Archiv mit 3,2 MB hochladen,

berechnen Sie die SHA-256-Hash-Werte für jeden der ersten drei 1 MB großen Datenstücke und berechnen dann den SHA-256-Hash der verbleibenden 0,2 MB Daten. Diese Hash-Werte formen die Blattknoten der Baumstruktur.

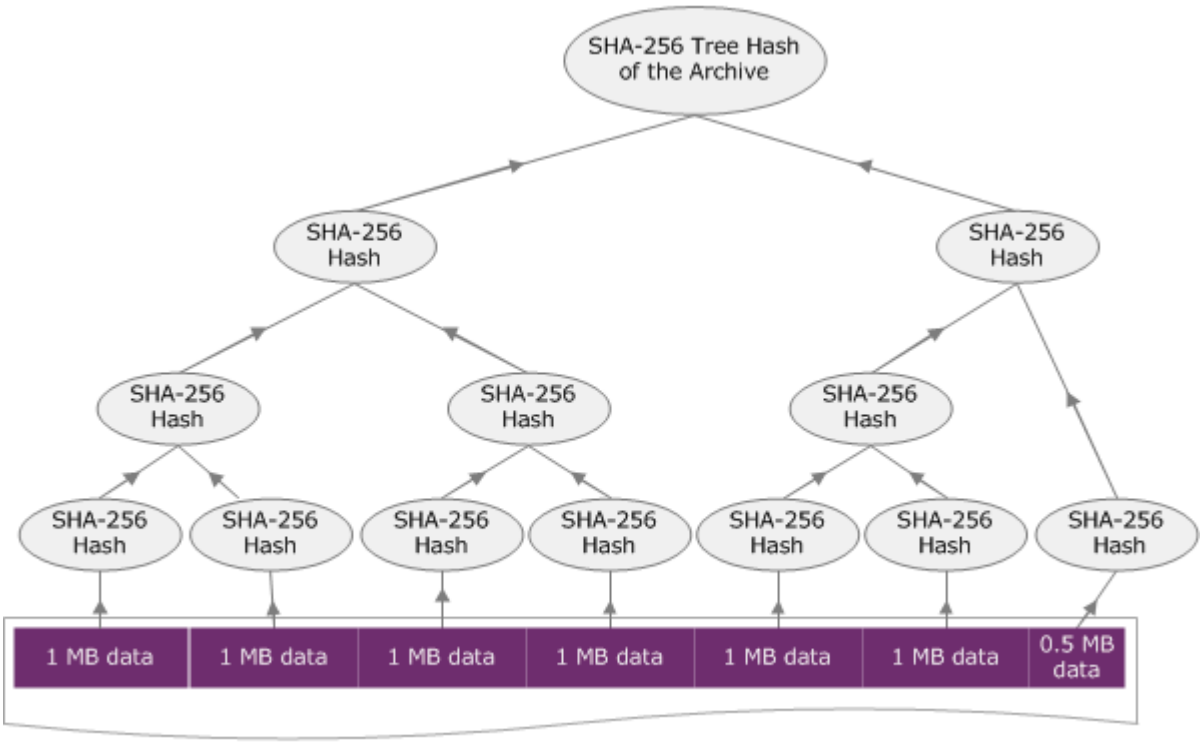
2. Erstellen Sie die nächste Ebene der Baumstruktur.
   a. Verketten Sie die Hash-Werte von zwei aufeinanderfolgenden untergeordneten Knoten und berechnen Sie den SHA-256-Wert der verketteten Hash-Werte. Durch diese Verkettung und Erstellung des SHA-256-Hashs wird ein übergeordneten Knoten für die beiden untergeordneten Knoten erstellt.
   b. Wenn nur ein untergeordneter Knoten übrig bleibt, stufen Sie diesen Hash-Wert auf die nächste Ebene in der Baumstruktur hoch.
3. Wiederholen Sie Schritt 2, bis die sich daraus ergebende Struktur einen Stamm hat. Der Stamm der Baumstruktur stellt einen Hash des gesamten Archivs bereit und ein Stamm der entsprechenden Nebenstruktur stellt den Hash des Teils eines mehrteiligen Uploads bereit.

Themen

- Struktur-Hash-Beispiel 1: Hochladen eines Archivs in einer einzelnen Anfrage
- Struktur-Hash-Beispiel 2: Hochladen eines Archivs mit einem mehrteiligen Upload
- Berechnen des Struktur-Hashs einer Datei
- Erhalten von Prüfsummen bei Daten-Download

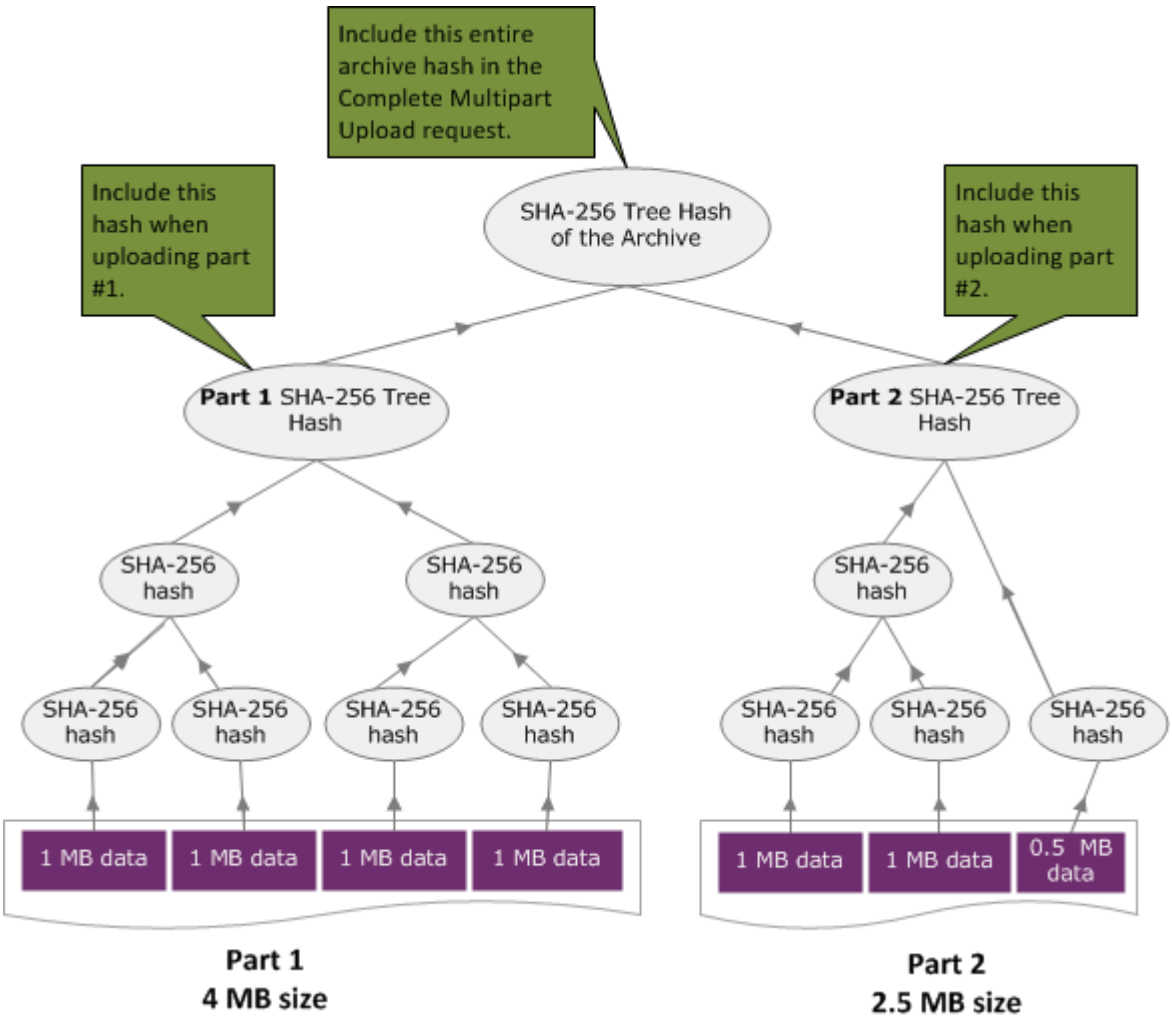
Struktur-Hash-Beispiel 1: Hochladen eines Archivs in einer einzelnen Anfrage

Wenn Sie ein Archiv in einer einzelnen Anfrage mit der „Upload Archive“-API hochladen (Informationen dazu unter Upload Archive (POST archive)), ist das gesamte Archiv in der Anfragenutzlast enthalten. Dem entsprechend muss der Struktur-Hash des gesamten Archivs im `x-amz-sha256-tree-hash`-Anfrage-Header enthalten sein. Angenommen, Sie möchten ein Archiv mit 6,5 MB hochladen. Das folgende Diagramm verdeutlicht den Vorgang zur Erstellung eines SHA-256-Hashs für das Archiv. Sie lesen das Archiv und berechnen den SHA-256-Hash für jeden 1 MB große Teil. Sie berechnen den Hash für die 0,5 MB der verbleiben Daten und erstellen die Struktur wie in den folgenden Schritten beschrieben.



Struktur-Hash-Beispiel 2: Hochladen eines Archivs mit einem mehrteiligen Upload

Der Vorgang zur Berechnung des Struktur-Hashs beim Upload eines Archivs mittels mehrteiligem Upload entspricht dem Archiv-Upload in einer einzelnen Anfrage. Der einzige Unterschied besteht darin, dass bei einem mehrteiligen Upload in jeder Anfrage (mit der Upload Part (PUT uploadID)-API) nur ein Teil des Archivs hochgeladen wird, und Sie daher die Prüfsumme nur dieses Teils im `x-amz-sha256-tree-hash`-Anfrage-Header bereitstellen. Nachdem alle Teile hochgeladen wurden, müssen Sie allerdings die „Complete Multipart Upload“-Anfrage senden (Informationen unter Complete Multipart Upload (POST uploadID)), wobei der Struktur-Hash des gesamten Archivs im `x-amz-sha256-tree-hash`-Anfrage-Header enthalten ist.



Berechnen des Struktur-Hashs einer Datei

Die hier gezeigten Algorithmen wurden zu Demonstrationszwecken ausgewählt. Sie können den Code nach Bedarf für Ihr Implementierungsszenario optimieren. Wenn Sie ein Amazon SDK verwenden, um gegen Amazon Glacier (Amazon Glacier) zu programmieren, wird die Baum-Hash-Berechnung für Sie durchgeführt und Sie müssen nur die Dateireferenz angeben.

Example 1: Java-Beispiel

Das folgende Beispiel zeigt, wie der SHA256 Baumhash einer Datei mithilfe von Java berechnet wird. Sie können dieses Beispiel entweder ausführen, indem Sie den Speicherort einer Datei als ein Argument angeben, oder Sie können die `TreeHashExample.computeSHA256TreeHash`-Methode direkt aus Ihrem Code verwenden.

```
import java.io.File;
import java.io.FileInputStream;
```

```
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class TreeHashExample {

static final int ONE_MB = 1024 * 1024;

    /**
     * Compute the Hex representation of the SHA-256 tree hash for the specified
     * File
     *
     * @param args
     *            args[0]: a file to compute a SHA-256 tree hash for
     */
    public static void main(String[] args) {

        if (args.length < 1) {
            System.err.println("Missing required filename argument");
            System.exit(-1);
        }

        File inputFile = new File(args[0]);
        try {

            byte[] treeHash = computeSHA256TreeHash(inputFile);
            System.out.printf("SHA-256 Tree Hash = %s\n", toHex(treeHash));

        } catch (IOException ioe) {
            System.err.format("Exception when reading from file %s: %s", inputFile,
                    ioe.getMessage());
            System.exit(-1);

        } catch (NoSuchAlgorithmException nsae) {
            System.err.format("Cannot locate MessageDigest algorithm for SHA-256: %s",
                    nsae.getMessage());
            System.exit(-1);
        }
    }

    /**
     * Computes the SHA-256 tree hash for the given file
     *
     * @param inputFile
```

```
     *            a File to compute the SHA-256 tree hash for
     * @return a byte[] containing the SHA-256 tree hash
     * @throws IOException
     *             Thrown if there's an issue reading the input file
     * @throws NoSuchAlgorithmException
     */
    public static byte[] computeSHA256TreeHash(File inputFile) throws IOException,
            NoSuchAlgorithmException {

        byte[][] chunkSHA256Hashes = getChunkSHA256Hashes(inputFile);
        return computeSHA256TreeHash(chunkSHA256Hashes);
    }

    /**
     * Computes a SHA256 checksum for each 1 MB chunk of the input file. This
     * includes the checksum for the last chunk even if it is smaller than 1 MB.
     *
     * @param file
     *            A file to compute checksums on
     * @return a byte[][] containing the checksums of each 1 MB chunk
     * @throws IOException
     *             Thrown if there's an IOException when reading the file
     * @throws NoSuchAlgorithmException
     *             Thrown if SHA-256 MessageDigest can't be found
     */
    public static byte[][] getChunkSHA256Hashes(File file) throws IOException,
            NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");

        long numChunks = file.length() / ONE_MB;
        if (file.length() % ONE_MB > 0) {
            numChunks++;
        }

        if (numChunks == 0) {
            return new byte[][] { md.digest() };
        }

        byte[][] chunkSHA256Hashes = new byte[(int) numChunks][];
        FileInputStream fileStream = null;

        try {
            fileStream = new FileInputStream(file);
```

```
            byte[] buff = new byte[ONE_MB];

            int bytesRead;
            int idx = 0;
            int offset = 0;

            while ((bytesRead = fileStream.read(buff, offset, ONE_MB)) > 0) {
                md.reset();
                md.update(buff, 0, bytesRead);
                chunkSHA256Hashes[idx++] = md.digest();
                offset += bytesRead;
            }

            return chunkSHA256Hashes;

        } finally {
            if (fileStream != null) {
                try {
                    fileStream.close();
                } catch (IOException ioe) {
                    System.err.printf("Exception while closing %s.\n %s",
file.getName(),
                            ioe.getMessage());
                }
            }
        }
    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1 MB chunk
     * checksums.
     *
     * This method uses a pair of arrays to iteratively compute the tree hash
     * level by level. Each iteration takes two adjacent elements from the
     * previous level source array, computes the SHA-256 hash on their
     * concatenated value and places the result in the next level's destination
     * array. At the end of an iteration, the destination array becomes the
     * source array for the next level.
     *
     * @param chunkSHA256Hashes
     *            An array of SHA-256 checksums
     * @return A byte[] containing the SHA-256 tree hash for the input chunks
     * @throws NoSuchAlgorithmException
     *             Thrown if SHA-256 MessageDigest can't be found
```

```
     */
    public static byte[] computeSHA256TreeHash(byte[][] chunkSHA256Hashes)
            throws NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");

        byte[][] prevLvlHashes = chunkSHA256Hashes;

        while (prevLvlHashes.length > 1) {

            int len = prevLvlHashes.length / 2;
            if (prevLvlHashes.length % 2 != 0) {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];

            int j = 0;
            for (int i = 0; i < prevLvlHashes.length; i = i + 2, j++) {

                // If there are at least two elements remaining
                if (prevLvlHashes.length - i > 1) {

                    // Calculate a digest of the concatenated nodes
                    md.reset();
                    md.update(prevLvlHashes[i]);
                    md.update(prevLvlHashes[i + 1]);
                    currLvlHashes[j] = md.digest();

                } else { // Take care of remaining odd chunk
                    currLvlHashes[j] = prevLvlHashes[i];
                }
            }

            prevLvlHashes = currLvlHashes;
        }

        return prevLvlHashes[0];
    }

    /**
     * Returns the hexadecimal representation of the input byte array
     *
     * @param data
```

```
     *            a byte[] to convert to Hex characters
     * @return A String containing Hex characters
     */
    public static String toHex(byte[] data) {
        StringBuilder sb = new StringBuilder(data.length * 2);

        for (int i = 0; i < data.length; i++) {
            String hex = Integer.toHexString(data[i] & 0xFF);

            if (hex.length() == 1) {
                // Append leading zero.
                sb.append("0");
            }
            sb.append(hex);
        }
        return sb.toString().toLowerCase();
    }
}
```

Example 2: C# .NET-Beispiel

Das folgende Beispiel zeigt, wie der SHA256 Baumhash einer Datei berechnet wird. Sie können dieses Beispiel ausführen, indem Sie den Speicherort der Datei als Argument angeben.

```
using System;
using System.IO;

using System.Security.Cryptography;

namespace ExampleTreeHash
{
    class Program
    {
        static int ONE_MB = 1024 * 1024;

        /**
        * Compute the Hex representation of the SHA-256 tree hash for the
        * specified file
        *
        * @param args
        *            args[0]: a file to compute a SHA-256 tree hash for
        */
        public static void Main(string[] args)
```

```
        {
            if (args.Length < 1)
            {
                Console.WriteLine("Missing required filename argument");
                Environment.Exit(-1);
            }
            FileStream inputFile = File.Open(args[0], FileMode.Open, FileAccess.Read);
            try
            {
                byte[] treeHash = ComputeSHA256TreeHash(inputFile);
                Console.WriteLine("SHA-256 Tree Hash = {0}",
BitConverter.ToString(treeHash).Replace("-", "").ToLower());
                Console.ReadLine();
                Environment.Exit(-1);
            }
            catch (IOException ioe)
            {
                Console.WriteLine("Exception when reading from file {0}: {1}",
                    inputFile, ioe.Message);
                Console.ReadLine();
                Environment.Exit(-1);
            }
            catch (Exception e)
            {
                Console.WriteLine("Cannot locate MessageDigest algorithm for SHA-256:
{0}",
                    e.Message);
                Console.WriteLine(e.GetType());
                Console.ReadLine();
                Environment.Exit(-1);
            }
            Console.ReadLine();
        }


        /**
         * Computes the SHA-256 tree hash for the given file
         *
         * @param inputFile
         *            A file to compute the SHA-256 tree hash for
         * @return a byte[] containing the SHA-256 tree hash
         */
        public static byte[] ComputeSHA256TreeHash(FileStream inputFile)
        {
```

```
            byte[][] chunkSHA256Hashes = GetChunkSHA256Hashes(inputFile);
            return ComputeSHA256TreeHash(chunkSHA256Hashes);
        }


        /**
         * Computes a SHA256 checksum for each 1 MB chunk of the input file. This
         * includes the checksum for the last chunk even if it is smaller than 1 MB.
         *
         * @param file
         *            A file to compute checksums on
         * @return a byte[][] containing the checksums of each 1MB chunk
         */
        public static byte[][] GetChunkSHA256Hashes(FileStream file)
        {
            long numChunks = file.Length / ONE_MB;
            if (file.Length % ONE_MB > 0)
            {
                numChunks++;
            }

            if (numChunks == 0)
            {
                return new byte[][] { CalculateSHA256Hash(null, 0) };
            }
            byte[][] chunkSHA256Hashes = new byte[(int)numChunks][];

            try
            {
                byte[] buff = new byte[ONE_MB];

                int bytesRead;
                int idx = 0;

                while ((bytesRead = file.Read(buff, 0, ONE_MB)) > 0)
                {
                    chunkSHA256Hashes[idx++] = CalculateSHA256Hash(buff, bytesRead);
                }
                return chunkSHA256Hashes;
            }
            finally
            {
                if (file != null)
                {
```

```
                try
                {
                    file.Close();
                }
                catch (IOException ioe)
                {
                    throw ioe;
                }
            }
        }

    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1MB chunk
     * checksums.
     *
     * This method uses a pair of arrays to iteratively compute the tree hash
     * level by level. Each iteration takes two adjacent elements from the
     * previous level source array, computes the SHA-256 hash on their
     * concatenated value and places the result in the next level's destination
     * array. At the end of an iteration, the destination array becomes the
     * source array for the next level.
     *
     * @param chunkSHA256Hashes
     *            An array of SHA-256 checksums
     * @return A byte[] containing the SHA-256 tree hash for the input chunks
     */
    public static byte[] ComputeSHA256TreeHash(byte[][] chunkSHA256Hashes)
    {
        byte[][] prevLvlHashes = chunkSHA256Hashes;
        while (prevLvlHashes.GetLength(0) > 1)
        {

            int len = prevLvlHashes.GetLength(0) / 2;
            if (prevLvlHashes.GetLength(0) % 2 != 0)
            {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];

            int j = 0;
            for (int i = 0; i < prevLvlHashes.GetLength(0); i = i + 2, j++)
```

```
                {

                    // If there are at least two elements remaining
                    if (prevLvlHashes.GetLength(0) - i > 1)
                    {

                        // Calculate a digest of the concatenated nodes
                        byte[] firstPart = prevLvlHashes[i];
                        byte[] secondPart = prevLvlHashes[i + 1];
                        byte[] concatenation = new byte[firstPart.Length +
 secondPart.Length];
                        System.Buffer.BlockCopy(firstPart, 0, concatenation, 0,
 firstPart.Length);
                        System.Buffer.BlockCopy(secondPart, 0, concatenation,
 firstPart.Length, secondPart.Length);

                        currLvlHashes[j] = CalculateSHA256Hash(concatenation,
 concatenation.Length);

                    }
                    else
                    { // Take care of remaining odd chunk
                        currLvlHashes[j] = prevLvlHashes[i];
                    }
                }

                prevLvlHashes = currLvlHashes;
            }

            return prevLvlHashes[0];
        }

        public static byte[] CalculateSHA256Hash(byte[] inputBytes, int count)
        {
            SHA256 sha256 = System.Security.Cryptography.SHA256.Create();
            byte[] hash = sha256.ComputeHash(inputBytes, 0, count);
            return hash;
        }
    }
}
```

Erhalten von Prüfsummen bei Daten-Download

Beim Abrufen eines Archivs mit der „Initiate Job“-API (Informationen unter Initiate Job (POST jobs)) können Sie optional einen abzurufenden Bereich des Archivs angeben. Beim Abrufen eines Archivs mit der „Get Job Output“-API (Informationen unter Get Job Output (GET output)) können Sie optional einen abzurufenden Bereich des Archivs angeben. Es ist wichtig, dass Sie zwei Merkmale dieser Bereiche verstehen, wenn Sie Ihre Archivdaten abrufen und herunterladen. Der abzurufende Bereich muss durch Megabytewerte auf das Archiv beziehen. Sowohl der abzurufende als auch der herunterzuladende Bereich muss sich durch Struktur-Hash-Werte zuordnen lassen, um die Werte der Prüfsumme zu erhalten, wenn Sie Ihre Daten herunterladen. Die beiden Typen der Bereichszuordnung werden wie folgt definiert:

- Auf Megabyte ausgerichtet — Ein Bereich [StartByte, EndBytes] ist auf Megabyte (1024*1024) ausgerichtet, wenn er durch 1 MB teilbar StartBytesist und EndBytesplus 1 durch 1 MB teilbar ist oder dem Ende des angegebenen Archivs entspricht (Archivbytegröße minus 1). Ein in der „Initiate Job“-API verwendeter Bereich, sofern angegeben, muss durch Megabytewerte zugeordnet sein.
- Tree-Hash Aligned — Ein Bereich [StartBytes, EndBytes] ist ein Baumhash, der in Bezug auf ein Archiv ausgerichtet ist, genau dann, wenn die Wurzel des über den Bereich erstellten Baum-Hashes einem Knoten im Baumhash des gesamten Archivs entspricht. Sowohl der abzurufende als auch der herunterzuladende Bereich muss durch Struktur-Hash-Werte zugeordnet sein, um die Werte der Prüfsumme für Ihre heruntergeladenen Daten zu erhalten. Ein Beispiel für Bereiche und deren Beziehungen zum Archiv-Struktur-Hash finden Sie unter Struktur-Hash-Beispiel: Abrufen eines durch Struktur-Hash-Werte zugeordneten Archivbereichs.

  Beachten Sie, dass ein durch Struktur-Hash-Werte zugeordneter Bereich zudem durch Megabyte-Werte zugeordnet ist. Ein durch Megabyte-Werte zugeordneter Bereich ist allerdings nicht notwendigerweise durch Struktur-Hash-Werte zugeordnet.

Mit den folgenden Fällen wird beschreiben, wann Sie beim Download Ihrer Archivdaten einen Prüfsummenwert erhalten:

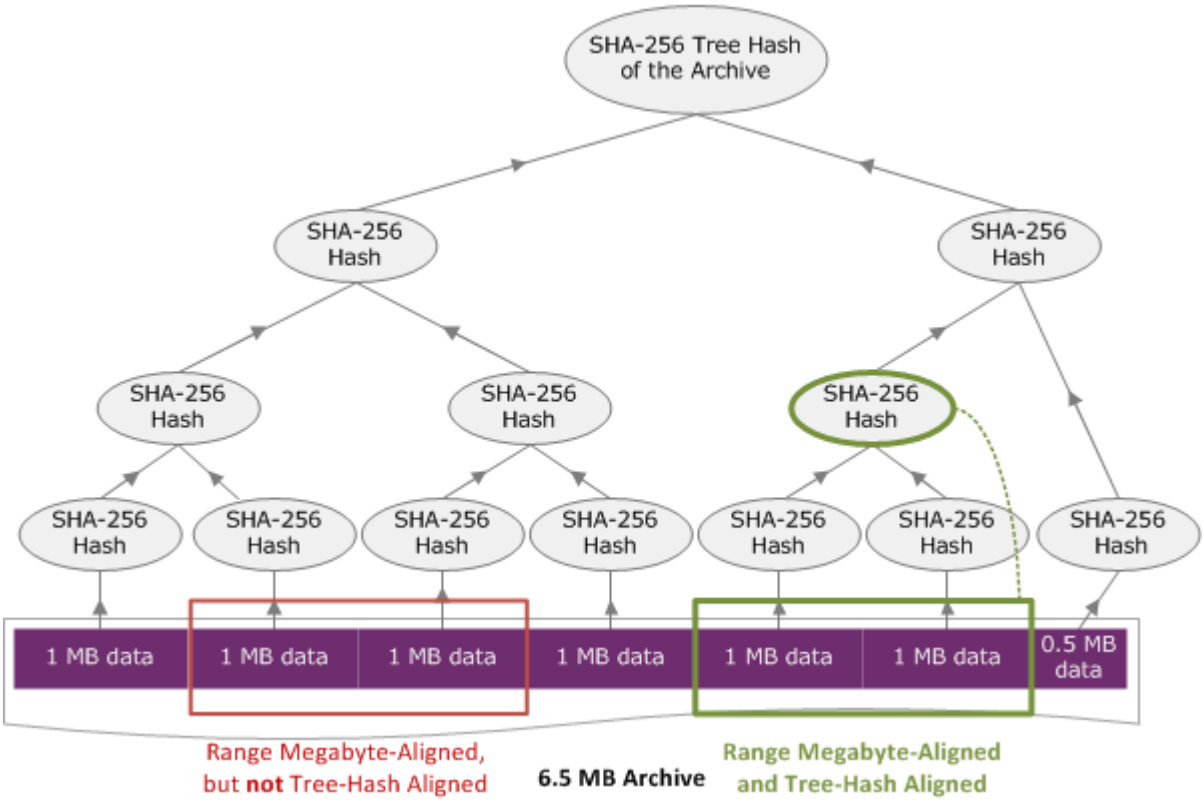
- Wenn Sie in der „Initiate Job“-Anfrage keinen abzurufenden Bereich angeben und das gesamte Archiv in der „Get Job“-Anfrage herunterladen.
- Wenn Sie in der „Initiate Job“-Anfrage keinen abzurufenden Bereich angeben und einen durch Struktur-Hash-Werte zugeordneten Bereich zum Download in der „Get Job“-Anfrage angeben.

- Wenn Sie in der „Initiate Job“-Anfrage keinen durch Struktur-Hash-Werte zugeordneten Bereich angeben und den gesamten Bereich in der „Get Job“-Anfrage herunterladen.
- Wenn Sie in der „Initiate Job“-Anfrage einen durch Struktur-Hash-Werte zugeordneten Bereich angeben und einen durch Struktur-Hash-Werte zugeordneten Bereich zum Download in der „Get Job“-Anfrage angeben.

Wenn Sie einen abzurufenden Bereich in der „Initiate Job“-Anfrage angeben, der nicht auf ein Struktur-Hash bezogen ist, dann können Sie Ihre Archivdaten dennoch abrufen, aber es werden keine Prüfsummenwerte zurückgegeben, wenn Sie die Daten in der „Get Job“-Anfrage herunterladen.

Struktur-Hash-Beispiel: Abrufen eines durch Struktur-Hash-Werte zugeordneten Archivbereichs

Angenommen, es ist ein 6,5 MB großes Archiv in Ihrem Tresor vorhanden und Sie möchten 2 MB des Archivs abrufen. Wie Sie den 2-MB-Bereich in der „Initiate Job“-Anfrage angeben, bestimmt, ob Sie beim Download der Daten Prüfsummenwerte für die Daten erhalten. Im folgenden Diagramm werden zwei 2 MB große Bereiche für 6,5 MB große Archive dargestellt, die heruntergeladen werden können. Beide Bereiche sind durch Megabyte-Werte zugeordnet, aber nur eins ist durch Struktur-Hash-Werte zugeordnet.



Durch Struktur-Hash-Werte zugeordnet Bereichsangabe

In diesem Abschnitt wird genau angegeben, was einen durch Struktur-Hash-Werte zugeordneten Bereich ausmacht. Durch Struktur-Hash-Werte zugeordnete Bereiche sind wichtig, wenn Sie einen Teil eines Archivs herunterladen und Sie den abzurufenden Datenbereich und den herunterzuladenden Bereich aus den abgerufenen Daten angeben. Wenn diese beiden Bereiche durch Struktur-Hash-Werte zugeordnet sind, erhalten Sie beim Download der Daten Prüfsummendaten.

Ein Bereich [A, B] gilt nur dann hinsichtlich eines Archiv als durch Struktur-Hash-Werte zugeordnet, wenn ein neuer Struktur-Hash über [A, B] des Stamm des für den Bereich erstellten Struktur-Hashs einem Knoten in dem Struktur-Hash des gesamten Archivs entspricht. Dies wird im Diagramm in Struktur-Hash-Beispiel: Abrufen eines durch Struktur-Hash-Werte zugeordneten Archivbereichs verdeutlicht. In diesem Abschnitt werden die Angaben für Struktur-Hash-Zuordnung bereitgestellt.

Beachten Sie [P, Q] als die Bereichsabfrage für ein Archiv von N Megabyte (MB). Wobei P sowie Q Mehrfache eines MB sind. Beachten Sie, dass der tatsächliche eingeschlossene Bereich [P MB, Q MB - 1 Byte] ist. Zur Vereinfachung zeigen wir dies allerdings als [P, Q]. Demnach ergibt sich Folgendes:

- Wenn P eine ungerade Zahl ist, gibt es nur einen möglichen Bereich, der dem Struktur-Hash zugeordnet ist, nämlich [P, P + 1 MB).
- Wenn P eine gerade Zahl und k die der Maximalwert ist, wobei P als 2k * X geschrieben werden kann, dann sind höchstens k dem Struktur-Hash-Werte zugeordnete Bereiche möglich, die mit P beginnen. X ist eine Ganzzahl größer 0. Die durch Struktur-Hash-Werte zugeordneten Bereiche fallen in folgende Kategorien:
  - Für jedes i, wobei (0 <= i <= k) ist und $P + 2^i < N$ ist, dann ist $[P, Q + 2^i]$ ein durch Struktur-Hash-Werte zugeordneter Bereich.
  - P = 0 ist ein besonderer Fall, bei dem A = 2[lgN]*0 ist.

Fehlermeldungen

Im Falle eines Fehlers gibt die API eine der folgenden Ausnahmen zurück:

Code	Description	HTTP-Statuscode	Typ
`AccessDeniedException`	Wird zurückgegeben, wenn versucht wurde, auf eine Ressource zuzugreifen, die durch eine AWS Identity and Access Management (IAM-) Richtlinie nicht zulässig ist, oder wenn die falsche AWS-Konto ID in der Anforderungs-URI verwendet wurde. Weitere Informationen finden Sie unter Identity and Access Management für Amazon Glacier.	`403 Forbidden`	Client
`BadRequest`	Wird zurückgegeben, wenn die Anfrage nicht verarbeitet werden kann.	`400 Bad Request`	Client
`ExpiredTokenException`	Wird zurückgegeben, wenn das in der Anfrage verwendete Sicherheits-Token abgelaufen ist.	`403 Forbidden`	Client
`InsufficientCapacityException`	Wird zurückgegeben, wenn nicht ausreichende Kapazität vorhanden ist, um die beschleunigte Anfrage zu verarbeiten. Dieser Fehler gilt nur für beschleunigte Abrufe und nicht für Standard- oder Massenabrufe.	`503 Service Unavailable`	Server
`InvalidParameterValueException`	Wird zurückgegeben, wenn ein Parameter der Anfrage falsch angegeben wurde.	`400 Bad Request`	Client

Code	Description	HTTP-Statuscode	Typ
`InvalidSignatureException`	Wird zurückgegeben, wenn die Anfragesignatur ungültig ist.	`403 Forbidden`	Client
`LimitExceededException`	Wird zurückgegeben, wenn die Anfrage dazu führt, dass eines der folgenden Limits überschritten wird: ein Tresorlimit, ein Tags-Limit oder das Limit der bereitgestellten Kapazität.	`400 Bad Request`	Client
`MissingAuthenticationTokenException`	Wird zurückgegeben, wenn keine Authentifizierungsdaten für die Anfrage gefunden werden.	`400 Bad Request`	Client
`MissingParameterValueException`	Wird zurückgegeben, wenn ein erforderlicher Header oder Parameter in der Anfrage fehlt.	`400 Bad Request`	Client
`PolicyEnforcedException`	Wird zurückgegeben, wenn ein Abrufauftrag das Limit der Abrufrate der aktuellen Datenrichtlinie überschreitet. Weitere Informationen zu Datenabrufrichtlinien finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf.	`400 Bad Request`	Client
`ResourceNotFoundException`	Wird zurückgegeben, wenn die angegebene Ressource, wie z. B. ein Tresor, Upload-ID oder Auftrags-ID nicht vorhanden ist.	`404 Not Found`	Client

Code	Description	HTTP-Statuscode	Typ
`RequestTimeoutException`	Wird zurückgegeben, wenn beim Hochladen eines Archivs ein Timeout bei Amazon Glacier (Amazon Glacier) beim Empfang des Uploads auftritt.	`408 Request Timeout`	Client
`SerializationException`	Wird zurückgegeben, wenn der Text der Anfrage ungültig ist. Wenn eine JSON-Nutzlast eingeschlossen ist, prüfen Sie, ob sie ein gültiges Format hat.	`400 Bad Request`	Client
`ServiceUnavailableException`	Wird zurückgegeben, wenn der Service die Anfrage nicht abschließen kann.	`500 Internal Server Error`	Server
`ThrottlingException`	Wird zurückgegeben, wenn Sie die Anzahl Ihrer Anfragen an Amazon Glacier reduzieren müssen.	`400 Bad Request`	Client
`UnrecognizedClientException`	Wird zurückgegeben, wenn die Zugriffsschlüssel-ID oder der Sicherheitstoken ungültig ist.	`400 Bad Request`	Client

Verschiedene Amazon Glacier APIs geben dieselbe Ausnahme zurück, jedoch mit unterschiedlichen Ausnahmemeldungen, um Ihnen bei der Behebung des jeweiligen aufgetretenen Fehlers zu helfen.

Amazon Glacier gibt Fehlerinformationen im Antworttext zurück. Die folgenden Beispiele zeigen einige der Antworten auf Fehler.

Beispiel 1: Beschreiben einer Auftragsanfrage mit einer Auftrags-ID, die nicht existiert

Angenommen, Sie senden eine Describe Job (GET JobID)-Anfrage für einen Auftrags, der nicht existiert. Das heißt, Sie geben eine Auftrags-ID an, die nicht existiert.

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVEXAMPLEbadJobID HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Als Antwort gibt Amazon Glacier die folgende Fehlerantwort zurück.

```
HTTP/1.1 404 Not Found
x-amzn-RequestId: AAABaZ9N92Iiyv4N7sru3ABEpSQkuFtmH3NP6aAC51ixfjg
Content-Type: application/json
Content-Length: 185
Date: Wed, 10 Feb 2017 12:00:00 GMT
{
  "code": "ResourceNotFoundException",
  "message": "The job ID was not found: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVEXAMPLEbadJobID",
  "type": "Client"
  }
```

Wobei Folgendes gilt:

Code

Eine der allgemeinen Ausnahmen.

Typ: Zeichenfolge

Fehlermeldung

Eine generische Beschreibung der Fehlerbedingung, die für die API spezifisch ist, die den Fehler zurückgibt.

Typ: Zeichenfolge

Typ

Die Quelle des Fehlers. Das Feld kann einen der folgenden Werte haben: `Client`, `Server` oder `Unknown`

Typ: Zeichenfolge.

Beachten Sie Folgendes in der vorherigen Antwort:

- Für die Fehlerantwort gibt Amazon Glacier die Statuscodewerte `4xx` und zurück`5xx`. In diesem Beispiel ist der Statuscode `404 Not Found`.
- Der Wert des `Content-Type`-Headers `application/json` zeigt JSON im Text an
- Das JSON im Text liefert die Information zum Fehler.

Angenommen, Sie geben in der vorherigen Anfrage anstatt einer falschen Auftrags-ID einen Tresor an, der nicht existiert. Die Antwort gibt eine andere Nachricht zurück.

```
HTTP/1.1 404 Not Found
x-amzn-RequestId: AAABBeC9Zw0rp_5D0L8VfB3FA_WlTupqTKAUehMcPhdgni0
Content-Type: application/json
Content-Length: 154
Date: Wed, 10 Feb 2017 12:00:00 GMT
{
  "code": "ResourceNotFoundException",
  "message": "Vault not found for ARN: arn:aws:glacier:us-west-2:012345678901:vaults/
examplevault",
  "type": "Client"
}
```

Beispiel 2: Anfrage zum Auflisten von Aufträgen mit einem ungültigen Wert für den Anfrageparameter

In diesem Beispiel senden Sie eine Aufträge auflisten (GET jobs)-Anfrage zum Abrufen von Tresoraufträgen mit einem bestimmten `statuscode` und Sie geben einen falschen `statuscode`-Wert `finished` an, anstatt der zulässigen Werte `InProgress`, `Succeeded` oder `Failed`.

```
GET /-/vaults/examplevault/jobs?statuscode=finished HTTP/1.1
```

```
Host: glacier.us-west-2.amazonaws.com
Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Amazon Glacier gibt das `InvalidParameterValueException` mit einer entsprechenden Nachricht zurück.

```
HTTP/1.1 400 Bad Request
x-amzn-RequestId: AAABaZ9N92Iiyv4N7sru3ABEpSQkuFtmH3NP6aAC51ixfjg
Content-Type: application/json
Content-Length: 141
Date: Wed, 10 Feb 2017 12:00:00 GMT
{
  "code": "InvalidParameterValueException",
  "message": "The job status code is not valid: finished",
  "type: "Client"
}
```

Tresor-Operationen

Im Folgenden sind die in Amazon Glacier verfügbaren Tresoroperationen aufgeführt.

Themen

- Abort Vault Lock (DELETE lock-policy)
- Add Tags To Vault (POST tags add)
- Create Vault (PUT vault)
- Complete Vault Lock (POST lockId)
- Delete Vault (DELETE vault)
- Delete Vault Access Policy (DELETE access-policy)
- Delete Vault Notifications (DELETE notification-configuration)
- Describe Vault (GET vault)
- Get Vault Access Policy (GET access-policy)
- Get Vault Lock (GET lock-policy)

- Get Vault Notifications (GET notification-configuration)
- Initiate Vault Lock (POST lock-policy)
- List Tags For Vault (GET tags)
- List Vaults (GET vaults)
- Remove Tags From Vault (POST tags remove)
- Set Vault Access Policy (PUT access-policy)
- Set Vault Notification Configuration (PUT notification-configuration)

Abort Vault Lock (DELETE lock-policy)

Beschreibung

Mit dieser Operation wird der Prozess zum Verriegeln eines Tresors beendet, wenn die Tresorsperre nicht den Status `Locked` aufweist. Wenn diese Operation angefordert wird, während die Tresorsperre den Status `Locked` aufweist, gibt die Operation einen `AccessDeniedException`-Fehler zurück. Durch das Beenden des Prozesses zum Verriegeln des Tresors wird die Tresor-Verriegelungsrichtlinie aus dem angegebenen Tresor entfernt.

Eine Tresorverriegelung wird in den Status `InProgress` versetzt, indem Initiate Vault Lock (POST lock-policy) aufgerufen wird. Eine Tresorverriegelung wird in den Status `Locked` versetzt, indem Complete Vault Lock (POST lockId) aufgerufen wird. Sie können den Status einer Tresorsperre abrufen, indem Sie Get Vault Lock (GET lock-policy) aufrufen. Weitere Informationen über den Tresorverriegelungsprozess finden Sie unter Amazon Glacier Tresorschloss. Weitere Informationen zu Tresorverriegelungs-Richtlinien finden Sie unter Tresorverriegelungs-Richtlinien.

Dieser Vorgang ist idempotent. Sie können diesen Vorgang mehrmals aufrufen, wenn der Tresor den Status `InProgress` aufweist oder wenn keine Richtlinie mit dem Tresor verknüpft ist.

Anforderungen

Um die Tresor-Sperrrichtlinie zu löschen, senden Sie eine HTTP-`DELETE`-Anforderung an den URI der Unterressource `lock-policy` des Tresors.

Syntax

```
DELETE /AccountId/vaults/vaultName/lock-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
```

```
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Wenn die Richtlinie erfolgreich gelöscht wurde, gibt Amazon Glacier eine `HTTP 204 No Content` Antwort zurück.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie der Prozess zum Verriegeln eines Tresors beendet wird.

Beispielanfrage

In diesem Beispiel wird eine `DELETE`-Anfrage an die Unterressource `lock-policy` des Tresors namens **`examplevault`** gesendet.

```
DELETE /-/vaults/examplevault/lock-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
x-amz-glacier-version: 2012-06-01
```

Beispielantwort

Wenn die Richtlinie erfolgreich gelöscht wurde, gibt Amazon Glacier eine `HTTP 204 No Content` Antwort zurück, wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Complete Vault Lock (POST lockId)

- Get Vault Lock (GET lock-policy)

- Initiate Vault Lock (POST lock-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Add Tags To Vault (POST tags add)

Mit dieser Operation werden einem Tresor die angegebenen Tags hinzugefügt. Jedes Tag besteht aus einem Schlüssel und einem Wert. Jeder Tresor kann bis zu 50 Tags aufweisen. Wenn Ihre Anforderung eine Überschreitung des Tag-Grenzwerts für den Tresors zur Folge hätte, löst der Vorgang den Fehler `LimitExceededException` aus.

Wenn ein Tag bereits im Tresor unter dem angegebenen Schlüssel vorhanden ist, wird der vorhandene Schlüsselwert überschrieben. Weitere Informationen zu Tags erhalten Sie unter Taggen von Amazon Glacier-Ressourcen.

Anforderungssyntax

Um einem Tresor Tags hinzuzufügen, senden Sie eine HTTP-POST-Anforderung an den URI der Tags, wie im folgenden Syntaxbeispiel gezeigt.

```
POST /AccountId/vaults/vaultName/tags?operation=add HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
```

```
   "Tags":
      {
         "string": "string",
         "string": "string"
      }
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Name	Beschreibung	Erforderlich
`operation=add`	Ein einziger Abfragezeichenfolgeparameter `operation` mit dem Wert `add` zur Unterscheidung von Remove Tags From Vault (POST tags remove).	Ja

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der Anforderungstext enthält die folgenden JSON-Felder.

Tags

Die Tags, die dem Tresor hinzugefügt werden sollen. Jedes Tag besteht aus einem Schlüssel und einem Wert. Der Wert kann auch eine leere Zeichenfolge sein.

Typ: Zeichenfolge-zu-Zeichenfolge-Zuweisung

Längenbeschränkungen: Minimale Länge von 1. Die maximale Länge beträgt 10 Zeichen.

Erforderlich: Ja

Antworten

Wenn die Anfrage erfolgreich ist, gibt der Service die HTTP-Antwort `204 No Content` zurück.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Das folgende Beispiel zeigt die HTTP-POST-Anforderung mit den Tags, die dem Tresor hinzugefügt werden sollen.

```
POST /-/vaults/examplevault/tags?operation=add HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01

{
  "Tags":
    {
       "examplekey1": "examplevalue1",
       "examplekey2": "examplevalue2"
    }
}
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier a zurück, `HTTP 204 No Content` wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

Zugehörige Abschnitte

- List Tags For Vault (GET tags)
- Remove Tags From Vault (POST tags remove)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Create Vault (PUT vault)

Beschreibung

Mit dieser Operation wird ein neuer Tresor mit dem angegebenen Namen erstellt. Der Name des Tresors muss innerhalb einer AWS Region für einen eindeutig sein AWS-Konto. Sie können bis zu 1.000 Tresore pro Konto erstellen. Informationen zum Erstellen weiterer Tresore finden Sie auf der Amazon Glacier-Produktdetailseite.

Sie müssen bei der Benennung eines Tresors die folgenden Richtlinien beachten.

- Namen können zwischen 1 und 255 Zeichen lang sein.
- Gültige Zeichen sind a-z, A-Z, 0–9, „_“ (Unterstrich), „-“ (Bindestrich) und „.“ (Punkt).

Dieser Vorgang ist idempotent, Sie können dieselbe Anfrage mehrmals senden und hat keine weiteren Auswirkungen, nachdem Amazon Glacier (Amazon Glacier) den angegebenen Tresor zum ersten Mal erstellt hat.

Anforderungen

Syntax

Zum Erstellen eines Tresors senden Sie eine HTTP PUT-Anforderung an den URI des zu erstellenden Tresors.

```
PUT /AccountId/vaults/VaultName HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein

einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der Anforderungstext für diese Operation muss leer sein (0 Bytes).

Antworten

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Beschreibung
`Location`	Der relative URI-Pad des erstellten Tresors. Typ: Zeichenfolge

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Im folgenden Beispiel wird eine HTTP PUT-Anfrage gesendet, um einen Tresor namens `examplevault` zu erstellen.

```
PUT /-/vaults/examplevault HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Content-Length: 0
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Amazon Glacier erstellt den Tresor und gibt den relativen URI-Pfad des Tresors im `Location` Header zurück. Die Konto-ID wird im `Location`-Header immer angegeben. Dies ist unabhängig von der Angabe der Konto-ID oder eines Bindestrichs („-“) in der Anfrage.

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Location: /111122223333/vaults/examplevault
```

Zugehörige Abschnitte

- List Vaults (GET vaults)
- Delete Vault (DELETE vault)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Complete Vault Lock (POST lockId)

Beschreibung

Diese Operation schließt den Tresorverriegelungsprozess ab, indem sie die Tresorverriegelung vom Zustand `InProgress` in den Zustand `Locked` überführt, wodurch die Tresorverriegelungsrichtlinie unveränderbar wird. Eine Tresorverriegelung wird in den Status `InProgress` versetzt, indem Initiate Vault Lock (POST lock-policy) aufgerufen wird. Sie können den Status der Tresorverriegelung abrufen, indem Sie Get Vault Lock (GET lock-policy) aufrufen. Weitere Informationen über den Tresorverriegelungsprozess finden Sie unter Amazon Glacier Tresorschloss.

Dieser Vorgang ist idempotent. Diese Anfrage ist immer erfolgreich, wenn die Tresorverriegelung im Status `Locked` ist und die bereitgestellte Vault-Lock-ID der Vault-Lock-ID entspricht, die ursprünglich für die Verriegelung des Tresors verwendet wurde.

Wenn eine ungültige Vault-Lock-ID in der Anfrage übergeben wird, wenn die Tresorverriegelung im Status `Locked` ist, gibt die Operation einen `AccessDeniedException`-Fehler zurück. Wenn eine ungültige Vault-Lock-ID in der Anfrage übergeben wird, wenn die Tresorverriegelung im Status `InProgress` ist, gibt die Operation einen `InvalidParameter`-Fehler zurück.

Anforderungen

Um den Tresorverriegelungsprozess abzuschließen, senden Sie eine HTTP-POST-Anfrage an den URI der `lock-policy`-Unterressource des Tresors mit einer gültigen Vault-Lock-ID.

Syntax

```
POST /AccountId/vaults/vaultName/lock-policy/lockId HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Der `lockId`-Wert ist die von einer Initiate Vault Lock (POST lock-policy)-Anfrage erhaltene Vault-Lock-ID.

Anforderungsparameter

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Wenn die Anfrage erfolgreich ist, gibt der Service die HTTP-Antwort `204 No Content` zurück.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Im folgenden Beispiel wird eine HTTP-POST-Anfrage mit der Vault-Lock-ID gesendet, um den Tresorverriegelungsprozess abzuschließen.

```
POST /-/vaults/examplevault/lock-policy/AE863rKkWZU53SLW5be4DUcW HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier (Amazon Glacier) eine `HTTP 204 No Content` Antwort zurück, wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

Zugehörige Abschnitte

- Abort Vault Lock (DELETE lock-policy)
- Get Vault Lock (GET lock-policy)

- Initiate Vault Lock (POST lock-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Delete Vault (DELETE vault)

Beschreibung

Diese Operation löscht einen Tresor. Amazon Glacier (Amazon Glacier) löscht einen Tresor nur, wenn sich im Tresor keine Archive gemäß der letzten Inventarisierung befinden und seit der letzten Inventur keine Schreibvorgänge in den Tresor stattgefunden haben. Wenn eine dieser Bedingungen nicht erfüllt ist, schlägt das Löschen des Tresors fehl (d. h., der Tresor wird nicht entfernt) und Amazon Glacier gibt einen Fehler zurück.

Sie können den Describe Vault (GET vault) Vorgang verwenden, der Tresorinformationen bereitstellt, einschließlich der Anzahl der Archive im Tresor. Die Informationen basieren jedoch auf dem zuletzt von Amazon Glacier generierten Tresorbestand.

Dieser Vorgang ist idempotent.

> **Note**
>
> Wenn Sie einen Tresor löschen, wird die Tresorzugriffsrichtlinie, die dem Tresor angefügt ist, ebenfalls gelöscht. Weitere Informationen zu Tresorzugriffsrichtlinien finden Sie unter Richtlinien für den Tresorzugriff.

Anforderungen

Um einen Tresor zu löschen, senden Sie eine `DELETE`-Anfrage an den Ressourcen-URI des Tresors.

Syntax

```
DELETE /AccountId/vaults/VaultName HTTP/1.1
```

```
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Im folgenden Beispiel wird ein Tresor namens `examplevault` gelöscht. Die Beispielanfrage ist eine `DELETE`-Anfrage an den URI der zu löschenden Ressource (Tresor).

```
DELETE /-/vaults/examplevault HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

Zugehörige Abschnitte

- Create Vault (PUT vault)
- List Vaults (GET vaults)
- Initiate Job (POST jobs)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Delete Vault Access Policy (DELETE access-policy)

Description

Diese Operation löscht die dem angegebenen Tresor zugeordnete Zugriffsrichtlinie. Der Vorgang ist letztendlich konsistent, d. h. es kann einige Zeit dauern, bis Amazon Glacier (Amazon Glacier) die Zugriffsrichtlinie vollständig entfernt hat, und Sie können die Auswirkungen der Richtlinie möglicherweise noch für kurze Zeit sehen, nachdem Sie die Löschanforderung gesendet haben.

Dieser Vorgang ist idempotent. Sie können das Löschen mehrmals aufrufen, auch wenn dem Tresor keine Richtlinie zugeordnet ist. Weitere Informationen zu Tresorzugriffsrichtlinien finden Sie unter Richtlinien für den Tresorzugriff.

Anforderungen

Um die aktuelle Tresorzugriffsrichtlinie zu löschen, senden Sie eine HTTP-`DELETE`-Anfrage an den URI der Unterressource `access-policy` des Tresors.

Syntax

```
DELETE /AccountId/vaults/vaultName/access-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen

zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Als Reaktion gibt Amazon Glacier `204 No Content` zurück, wenn die Richtlinie erfolgreich gelöscht wurde.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie eine Tresorzugriffsrichtlinie gelöscht wird.

Beispielanforderung

In diesem Beispiel wird eine `DELETE`-Anfrage an die Unterressource `access-policy` des Tresors namens **`examplevault`** gesendet.

```
DELETE /-/vaults/examplevault/access-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
x-amz-glacier-version: 2012-06-01
```

Beispielantwort

Wenn die Richtlinie erfolgreich gelöscht wurde, gibt Amazon Glacier als Antwort a zurück, `204 No Content` wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Get Vault Access Policy (GET access-policy)
- Set Vault Access Policy (PUT access-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Delete Vault Notifications (DELETE notification-configuration)

Description

Diese Operation löscht die für einen Tresor eingestellte Benachrichtigungskonfiguration Set Vault Notification Configuration (PUT notification-configuration). Der Vorgang ist letztendlich konsistent, d. h. es kann einige Zeit dauern, bis Amazon Glacier (Amazon Glacier) die Benachrichtigungen vollständig deaktiviert hat, und Sie erhalten möglicherweise noch einige Benachrichtigungen für kurze Zeit, nachdem Sie die Löschanforderung gesendet haben.

Anforderungen

Senden Sie eine `DELETE`-Anfrage an die Unterressource `notification-configuration` des Tresors, um die Benachrichtigungskonfiguration eines Tresors zu löschen.

Syntax

```
DELETE /AccountId/vaults/VaultName/notification-configuration HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie die Benachrichtigungskonfiguration für einen Tresor entfernt wird.

Beispielanforderung

In diesem Beispiel wird eine `DELETE`-Anfrage an die Unterressource `notification-configuration` des Tresors namens `examplevault` gesendet.

```
DELETE /111122223333/vaults/examplevault/notification-configuration HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Get Vault Notifications (GET notification-configuration)
- Set Vault Notification Configuration (PUT notification-configuration)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Describe Vault (GET vault)

Beschreibung

Diese Operation gibt Informationen über einen Tresor zurück, einschließlich Tresor-ARN (Amazon-Ressourcenname), Datum der Erstellung, Anzahl der Archive im Tresor und Gesamtgröße aller Archive im Tresor. Die Anzahl der Archive und ihre Gesamtgröße entsprechen dem zuletzt von Amazon Glacier (Amazon Glacier) generierten Tresorinventar (siheArbeiten mit Tresoren in Amazon Glacier). Amazon Glacier generiert etwa täglich Tresorbestände. Das bedeutet, dass wenn

Sie ein Archiv zum Tresor hinzufügen oder aus ihm entfernen und dann sofort eine Anfrage zur Tresorbeschreibung senden, die Änderungen möglicherweise in der Antwort nicht reflektiert werden.

Anforderungen

Senden Sie eine `GET`-Anfrage an den URI der spezifischen Tresorressource, um Informationen über einen Tresor zu erhalten.

Syntax

```
GET /AccountId/vaults/VaultName HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
  "CreationDate" : String,
  "LastInventoryDate" : String,
  "NumberOfArchives" : Number,
  "SizeInBytes" : Number,
  "VaultARN" : String,
  "VaultName" : String
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

CreationDate

Das UTC-Datum, zu dem der Tresor erstellt wurde.

Typ: Eine Zeichenkettendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

LastInventoryDate

Das UTC-Datum, an dem Amazon Glacier die letzte Tresorinventur abgeschlossen hat. Informationen zum Initiieren einer Inventur für einen Tresor finden Sie unter Initiate Job (POST jobs).

Typ: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

NumberOfArchives

Die Anzahl der Archive im Tresor gemäß der letzten Tresorinventur. Dieses Feld wird Null zurückgeben, wenn noch keine Inventur auf dem Tresor ausgeführt wurde, zum Beispiel, wenn Sie den Tresor gerade erst erstellt haben.

Typ: Zahl

SizeInBytes

Die Gesamtgröße in Bytes der Archive im Tresor, einschließlich jedes Overheads pro Archiv, am Datum der letzten Inventur. Dieses Feld wird `null` zurückgeben, wenn noch keine Inventur auf dem Tresor ausgeführt wurde, zum Beispiel, wenn Sie den Tresor gerade erst erstellt haben.

Typ: Zahl

VaultARN

Der Amazon-Ressourcenname (ARN) des Tresors.

Typ: Zeichenfolge

VaultName

Der Tresorname, der zum Zeitpunkt der Erstellung angegeben wurde. Der Tresorname ist auch in der ARN des Tresors enthalten.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Das folgende Beispiel zeigt, wie Sie Informationen über den Tresor namens `examplevault` erhalten.

```
GET /-/vaults/examplevault HTTP/1.1
```

```
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: 260

{
  "CreationDate" : "2012-02-20T17:01:45.198Z",
  "LastInventoryDate" : "2012-03-20T17:03:43.221Z",
  "NumberOfArchives" : 192,
  "SizeInBytes" : 78088912,
  "VaultARN" : "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault",
  "VaultName" : "examplevault"
}
```

Zugehörige Abschnitte

- Create Vault (PUT vault)
- List Vaults (GET vaults)
- Delete Vault (DELETE vault)
- Initiate Job (POST jobs)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Get Vault Access Policy (GET access-policy)

Description

Durch diesen Vorgang wird die im Tresor festgelegte `access-policy`-Unterressource abgerufen. Weitere Informationen zum Festlegen dieser Unterressource finden Sie unter Set Vault Access Policy (PUT access-policy). Wenn auf dem Tresor keine Zugriffsrichtlinie eingerichtet ist, gibt die Operation einen `404 Not found`-Fehler zurück. Weitere Informationen zu Tresorzugriffsrichtlinien finden Sie unter Richtlinien für den Tresorzugriff.

Anforderungen

Um die aktuelle Tresorzugriffsrichtlinie zurückzugeben, senden Sie eine HTTP-`GET`-Anfrage an den URI der Unterressource `access-policy` des Tresors.

Syntax

```
GET /AccountId/vaults/vaultName/access-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Als Antwort gibt Amazon Glacier (Amazon Glacier) die Tresorzugriffsrichtlinie im JSON-Format im Hauptteil der Antwort zurück.

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length

{
  "Policy": "string"
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

Richtlinie

Die Tresorzugriffsrichtlinie als eine JSON-Zeichenfolge, die "\" als Escape-Zeichen verwendet.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie eine Tresorzugriffsrichtlinie abgerufen wird.

Beispielanforderung

In diesem Beispiel wird eine GET-Anforderung an den URI der `access-policy`-Unterressource eines Tresors gesendet.

```
GET /-/vaults/examplevault/access-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier die Tresorzugriffsrichtlinie als JSON-Zeichenfolge im Hauptteil der Antwort zurück. Die zurückgegebene JSON-Zeichenfolge verwendet „\“ als Escape-Zeichen, wie in den Beispielen Set Vault Access Policy (PUT access-policy) gezeigt. Das folgende Beispiel zeigt die zurückgegebene JSON-Zeichenfolge zur besseren Lesbarkeit allerdings ohne als Escape-Zeichen.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: length

{
  "Policy": "
    {
      "Version": "2012-10-17",
      "Statement": [
        {
```

```
          "Sid": "allow-time-based-deletes",
          "Principal": {
            "AWS": "999999999999"
          },
          "Effect": "Allow",
          "Action": "glacier:Delete*",
          "Resource": [
            "arn:aws:glacier:us-west-2:999999999999:vaults/examplevault"
          ],
          "Condition": {
            "DateGreaterThan": {
              "aws:CurrentTime": "2018-12-31T00:00:00Z"
            }
          }
        }
      ]
    }
  "
}
```

Zugehörige Abschnitte

- Delete Vault Access Policy (DELETE access-policy)
- Set Vault Access Policy (PUT access-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Get Vault Lock (GET lock-policy)

Description

Mit dieser Operation werden die folgenden Attribute aus der `lock-policy`-Unterressource abgerufen, die für den angegeben Tresor festgelegt ist:

- Die für den Tresor festgelegte Tresorverriegelungsrichtlinie.
- Der Status der Tresorverriegelung. Er lautet entweder `InProgess` oder `Locked`.
- Bei Ablauf der Vault-Lock-ID. Die Vault-Lock-ID wird verwendet, um den Prozess zur Tresorverriegelung abzuschließen.
- Wenn die Tresorverriegelung initiiert und in den `InProgress`-Status versetzt wurde.

Eine Tresorverriegelung wird in den Status `InProgress` versetzt, indem Initiate Vault Lock (POST lock-policy) aufgerufen wird. Eine Tresorverriegelung wird in den Status `Locked` versetzt, indem Complete Vault Lock (POST lockId) aufgerufen wird. Sie können den Prozess zur Tresorverriegelung beenden, indem Sie Abort Vault Lock (DELETE lock-policy) aufrufen. Weitere Informationen über den Tresorverriegelungsprozess finden Sie unter Amazon Glacier Tresorschloss.

Wenn auf dem Tresor keine Tresorverriegelungsrichtlinie eingerichtet ist, gibt die Operation einen `404 Not found`-Fehler zurück. Weitere Informationen zu Tresorverriegelungs-Richtlinien finden Sie unter Tresorverriegelungs-Richtlinien.

Anforderungen

Sie geben die aktuelle Tresorverriegelungsrichtlinie und andere Attribute zurück, indem Sie eine HTTP-`GET`-Anfrage an den URI der `lock-policy`-Unterressource des Tresors senden. Orientieren Sie sich dabei an dem folgenden Syntaxbeispiel.

Syntax

```
GET /AccountId/vaults/vaultName/lock-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Als Antwort gibt Amazon Glacier (Amazon Glacier) die Tresorzugriffsrichtlinie im JSON-Format im Hauptteil der Antwort zurück.

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length

{
  "Policy": "string",
  "State": "string",
  "ExpirationDate": "string",
  "CreationDate":"string"
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

Richtlinie

Die Tresorverriegelungsrichtlinie als eine JSON-Zeichenfolge, bei der "\" als Escape-Zeichen verwendet wird.

Typ: Zeichenfolge

Status

Der Status der Tresorverriegelung.

Typ: Zeichenfolge

Zulässige Werte: `InProgress|Locked`

ExpirationDate

Datum und Uhrzeit im UTC-Format für den Ablauf der Vault-Lock-ID. Dieser Wert kann `null` lauten, wenn die Tresorverriegelung den Status `Locked` angenommen hat.

Typ: Eine Zeichenkettendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

CreationDate

Datum und Uhrzeit im UTC-Format für den Zeitpunkt, an dem die Tresorverriegelung den Status `InProgress` angenommen hat.

Typ: Eine Zeichenkettendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie eine Tresorverriegelungsrichtlinie abgerufen wird.

Beispielanforderung

In diesem Beispiel wird eine `GET`-Anforderung an den URI der `lock-policy`-Unterressource eines Tresors gesendet.

```
GET /-/vaults/examplevault/lock-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier die Tresorzugriffsrichtlinie als JSON-Zeichenfolge im Hauptteil der Antwort zurück. Die zurückgegebene JSON-Zeichenfolge, bei der "\" als Escape-Zeichen verwendet wird, wie in der Beispielanforderung Initiate Vault Lock (POST lock-policy) verdeutlicht. Das folgende Beispiel zeigt die zurückgegebene JSON-Zeichenfolge zur besseren Lesbarkeit allerdings ohne als Escape-Zeichen.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: length

{
  "Policy": "
    {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "Define-vault-lock",
          "Principal": {
            "AWS": "arn:aws:iam::999999999999:root"
          },
          "Effect": "Deny",
          "Action": "glacier:DeleteArchive",
          "Resource": [
            "arn:aws:glacier:us-west-2:999999999999:vaults/examplevault"
          ],
          "Condition": {
            "NumericLessThanEquals": {
              "glacier:ArchiveAgeInDays": "365"
            }
          }
```

```
        }
      ]
    }
  ",
  "State": "InProgress",
  "ExpirationDate": "exampledate",
  "CreationDate": "exampledate"
}
```

Zugehörige Abschnitte

- Abort Vault Lock (DELETE lock-policy)
- Complete Vault Lock (POST lockId)
- Initiate Vault Lock (POST lock-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Get Vault Notifications (GET notification-configuration)

Description

Diese Operation ruft die `notification-configuration`-Unterressource ab, die auf dem Tresor eingerichtet ist, (siehe Set Vault Notification Configuration (PUT notification-configuration). Wenn die Benachrichtigungskonfiguration für einen Tresor nicht eingerichtet ist, gibt die Operation einen `404 Not Found`-Fehler zurück. Weitere Informationen zu Tresorbenachrichtigungen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier.

Anforderungen

Um Informationen zur Benachrichtigungskonfiguration abzurufen, senden Sie eine GET-Anfrage an den URI der `notification-configuration`-Unterressource des Tresors.

Syntax

```
GET /AccountId/vaults/VaultName/notification-configuration HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
```

```
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length
{
  "Events": [
    String,
    ...
  ],
  "SNSTopic": String
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

--Ereignisse

Eine Liste von einem oder mehreren Ereignissen, für die Amazon Glacier (Amazon Glacier) eine Benachrichtigung an das angegebene Amazon SNS SNS-Thema sendet. Informationen über Tresorereignisse, für die Sie Tresore so konfigurieren können, dass Benachrichtigungen veröffentlicht werden, finden Sie unter Set Vault Notification Configuration (PUT notification-configuration).

Typ: Array

SNSTopic

Der Amazon-Ressourcenname (ARN) des Amazon Simple Notification Service (Amazon SNS)-Themas. Weitere Informationen finden Sie unter Erste Schritte mit Amazon SNS im Handbuch „Erste Schritte“ für den Amazon Simple Notification Service.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie die Benachrichtigungskonfiguration für einen Tresor abgerufen wird.

Beispielanforderung

In diesem Beispiel wird eine `GET`-Anfrage an die `notification-configuration`-Unterressource eines Tresors gesendet.

```
GET /-/vaults/examplevault/notification-configuration HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Falls die Anforderung erfolgreich beantwortet wird, wird das Dokument zur Konfiguration der Prüfprotokollierung im Antworttext im JSON-Format angezeigt. In diesem Beispiel zeigt die Konfiguration, dass Benachrichtigungen für zwei Ereignisse (`ArchiveRetrievalCompleted` und `InventoryRetrievalCompleted`) an das Amazon-SNS-Thema `arn:aws:sns:us-west-2:012345678901:mytopic` gesendet werden.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 150

{
  "Events": [
    "ArchiveRetrievalCompleted",
    "InventoryRetrievalCompleted"
  ],
  "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic"
```

```
}
```

Zugehörige Abschnitte

- Delete Vault Notifications (DELETE notification-configuration)
- Set Vault Notification Configuration (PUT notification-configuration)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Initiate Vault Lock (POST lock-policy)

Beschreibung

Diese Operation initiiert den Prozess zur Tresorverriegelung, indem Folgendes geschieht:

- Installieren einer Richtlinie zur Tresorverriegelung für den angegebenen Tresor.
- Einstellen des Verriegelungsstatus der Tresorverriegelung auf `InProgress`.
- Rückgabe einer Vault-Lock-ID, die verwendet wird, um den Prozess zur Tresorverriegelung abzuschließen.

Sie können pro Tresor eine Richtlinie zur Tresorverriegelung mit einer Größe von bis zu 20 KB festlegen. Weitere Informationen zu Tresorverriegelungs-Richtlinien finden Sie unter Tresorverriegelungs-Richtlinien.

Sie müssen den Prozess zur Tresorverriegelung innerhalb von 24 Stunden, nachdem die Tresorverriegelung in den Status `InProgress` eintritt, abschließen. Nach diesen 24 Stunden läuft die Vault-Lock-ID ab, der Tresor verlässt automatisch den Status `InProgress` und die Richtlinie zur Tresorverriegelung wird vom Tresor entfernt. Sie rufen Complete Vault Lock (POST lockId) auf, um den Prozess zur Tresorverriegelung abzuschließen, indem Sie den Status der Tresorverriegelung auf `Locked` einstellen.

> **Note**
>
> Wenn eine Tresorverriegelung im Status `Locked` ist, können Sie keine neue Tresorverriegelung für den Tresor initiieren.

Sie können den Prozess zur Tresorverriegelung beenden, indem Sie Abort Vault Lock (DELETE lock-policy) aufrufen. Sie können den Status der Tresorverriegelung abrufen, indem Sie Get Vault Lock (GET lock-policy) aufrufen. Weitere Informationen über den Tresorverriegelungsprozess finden Sie unter Amazon Glacier Tresorschloss.

Wenn diese Operation aufgerufen wird, wenn die Tresorverriegelung im Status `InProgress` ist, gibt die Operation einen `AccessDeniedException`-Fehler zurück. Wenn die Tresorverriegelung im Status `InProgress` ist, müssen Sie Abort Vault Lock (DELETE lock-policy) aufrufen, bevor Sie eine neue Tresorverriegelungs-Richtlinie initiieren können.

Anforderungen

Um den Prozess zur Tresorverriegelung zu initiieren, senden Sie eine HTTP-`POST`-Anforderung an den URI der `lock-policy`-Unterressource des Tresors. Orientieren Sie sich dabei an dem folgenden Syntaxbeispiel.

Syntax

```
POST /AccountId/vaults/vaultName/lock-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
  "Policy": "string"
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der

Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der Anforderungstext enthält die folgenden JSON-Felder.

Richtlinie

Die Tresorverriegelungsrichtlinie als eine JSON-Zeichenfolge, bei der "\" als Escape-Zeichen verwendet wird.

Typ: Zeichenfolge

Erforderlich: Ja

Antworten

Amazon Glacier (Amazon Glacier) gibt eine `HTTP 201 Created` Antwort zurück, wenn die Richtlinie akzeptiert wird.

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-lock-id: lockId
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Beschreibung
`x-amz-lock-id`	Die Vault-Lock-ID, die verwendet wird, um den Prozess zur Tresorverriegelung abzuschließen. Typ: Zeichenfolge

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Das folgende Beispiel sendet eine HTTP-PUT-Anfrage an den URI der `lock-policy`-Unterressource des Tresors. In der JSON-Zeichenfolge `Policy` wird „\“ als Escape-Zeichen verwendet.

```
PUT /-/vaults/examplevault/lock-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01
```

```
{"Policy":"{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Sid\":\"Define-vault-
lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam::999999999999:root
\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-
west-2:999999999999:vaults/examplevault\",\"Condition\":{\"NumericLessThanEquals\":
{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}"}
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier eine `HTTP 201 Created` Antwort zurück, wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
x-amz-lock-id: AE863rKkWZU53SLW5be4DUcW
```

Zugehörige Abschnitte

- Abort Vault Lock (DELETE lock-policy)
- Complete Vault Lock (POST lockId)
- Get Vault Lock (GET lock-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

List Tags For Vault (GET tags)

Diese Operation listet alle Tags auf, die einem Tresor angefügt sind. Die Operation gibt eine leere Zuweisung zurück, wenn keine Tags vorhanden sind. Weitere Informationen zu Tags erhalten Sie unter Taggen von Amazon Glacier-Ressourcen.

Anforderungssyntax

Um die Tags für einen Tresor aufzulisten, senden Sie eine HTTP-GET-Anfrage an den URI der Tags, wie im folgenden Syntaxbeispiel gezeigt.

```
GET /AccountId/vaults/vaultName/tags HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Wenn die Operation erfolgreich ist, sendet der Service eine HTTP-`200 OK`-Antwort zurück.

Antwortsyntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length
{
   "Tags":
      {
         "string" : "string",
         "string" : "string"
      }
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

Tags

Die dem Tresor angefügten Tags. Jedes Tag besteht aus einem Schlüssel und einem Wert.

Typ: Zeichenfolge-zu-Zeichenfolge-Zuweisung

Erforderlich: Ja

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispiel: Auflisten von Tags für einen Tresor

Das folgende Beispiel listet die Tags für einen Tresor auf.

Beispielanfrage

In diesem Beispiel wird eine GET-Anfrage gesendet, um eine Liste von Tags von dem angegebenen Tresor abzurufen.

```
GET /-/vaults/examplevault/tags HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier (Amazon Glacier) a `HTTP 200 OK` mit einer Liste von Tags für den Tresor zurück, wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: length

{
   "Tags",
      {
         "examplekey1": "examplevalue1",
         "examplekey2": "examplevalue2"
      }
}
```

Zugehörige Abschnitte

- Add Tags To Vault (POST tags add)
- Remove Tags From Vault (POST tags remove)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

List Vaults (GET vaults)

Beschreibung

Diese Operation listet alle Tresore auf, die das Konto des aufrufenden Benutzers besitzt. Die zurückgegebene Liste ist ASCII-sortiert nach Tresorname.

Standardmäßig gibt diese Operation bis zu 10 Elemente pro Anfrage zurück. Wenn mehr aufzulistende Tresore vorhanden sind, enthält das Feld `marker` im Antworttext den Amazon-Ressourcennamen (ARN) des Tresors, bei dem die Liste mit einer neuen List Vaults-Anfrage fortzusetzen ist, andernfalls hat das Feld `marker` den Wert `null`. In Ihrer nächsten List Vaults-Anfrage setzen Sie den `marker` Parameter auf den Wert, den Amazon Glacier (Amazon Glacier) in den Antworten auf Ihre vorherige List Vaults-Anfrage zurückgegeben hat. Sie können die in der Antwort zurückgegebene Anzahl der Tresore auch begrenzen, indem Sie in der Anfrage den Parameter `limit` angeben.

Anforderungen

Um eine Liste der Tresore abzurufen, senden Sie eine `GET`-Anfrage an die Ressource Tresore.

Syntax

```
GET /AccountId/vaults HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der

Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Diese Operation verwendet die folgenden Anforderungsparameter.

Name	Beschreibung	Erforderlich
`limit`	Die maximale Anzahl der zurückzugebenden Tresore. Das Standardlimit ist 10. Die Anzahl der zurückgegebenen Tresore kann kleiner sein als das festgelegte Limit, aber die Anzahl der zurückgegebenen Tresore überschreitet das Limit nie. Typ: Zeichenfolge Beschränkungen: Mindestganzzahlwert 1. Höchstganzzahlwert 10.	Nein
`marker`	Eine zur Paginierung verwendete Zeichenfolge. `marker` gibt den ARN des Tresors an, nach dem die Auflistung der Tresore beginnen soll. (Der von `marker` angegebene Tresor ist in der zurückgegebenen Liste nicht enthalten.) Rufen Sie den Wert für `marker` aus einer vorherigen List Vaults-Antwort ab. Sie müssen `marker` nur dann einschließen, wenn Sie die Paginierung der Ergebnisse fortsetzen, die in einer vorherigen List Vaults-Anfrage gestartet wurde. Die Angabe eines leeren Werts ("") für die Markierung gibt eine Liste von Tresoren zurück, die mit dem ersten Tresor beginnt. Typ: Zeichenfolge	Nein

Name	Beschreibung	Erforderlich
	Einschränkungen: Keine	

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
  "Marker": String
  "VaultList": [
   {
    "CreationDate": String,
    "LastInventoryDate": String,
    "NumberOfArchives": Number,
    "SizeInBytes": Number,
    "VaultARN": String,
    "VaultName": String
   },
   ...
  ]
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

CreationDate

Das Erstellungsdatum des Tresors, in UTC (Coordinated Universal Time).

Typ: Zeichenfolge. Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, zum Beispiel `2013-03-20T17:03:43.221Z`.

LastInventoryDate

Das Datum der letzten Tresorinventur, in UTC (Coordinated Universal Time). Dieses Feld kann Null sein, wenn noch keine Inventur auf dem Tresor ausgeführt wurde, zum Beispiel, wenn Sie den Tresor gerade erst erstellt haben. Informationen zum Initiieren einer Inventur für einen Tresor finden Sie unter Initiate Job (POST jobs).

Typ: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

Marker

Der `vaultARN`, der angibt, wo die Paginierung der Ergebnisse fortgesetzt werden soll. Mithilfe des `marker` in einer anderen List Vaults-Anfrage erhalten Sie mehr Tresore in der Liste. Wenn keine anderen Tresore vorhanden sind, ist dieser Wert `null`.

Typ: Zeichenfolge

NumberOfArchives

Die Anzahl der Archive im Tresor am Datum der letzten Inventur.

Typ: Zahl

SizeInBytes

Die Gesamtgröße, in Bytes, aller Archive im Tresor, einschließlich jedes Overheads pro Archiv, am Datum der letzten Inventur.

Typ: Zahl

VaultARN

Der Amazon-Ressourcenname (ARN) des Tresors.

Typ: Zeichenfolge

VaultList

Ein Array von Objekten, von dem jedes Objekt eine Beschreibung eines Tresors bereitstellt.

Typ: Array

VaultName

Der Name des Tresors.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispiel: Alle Tresore auflisten

Das folgende Beispiel listet Tresore auf. Da die Parameter `marker` und `limit` in der Anfrage nicht angegeben sind, werden bis zu 10 Tresore zurückgegeben.

Beispielanfrage

```
GET /-/vaults HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Der `Marker` ist `null` und zeigt damit an, dass keine aufzulistenden Tresore mehr vorhanden sind.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: 497

{
  "Marker": null,
  "VaultList": [
   {
    "CreationDate": "2012-03-16T22:22:47.214Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 2,
    "SizeInBytes": 12334,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault1",
    "VaultName": "examplevault1"
   },
   {
    "CreationDate": "2012-03-19T22:06:51.218Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 0,
    "SizeInBytes": 0,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault2",
    "VaultName": "examplevault2"
   },
   {
    "CreationDate": "2012-03-19T22:06:51.218Z",
    "LastInventoryDate": "2012-03-25T12:14:31.121Z",
    "NumberOfArchives": 0,
    "SizeInBytes": 0,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault3",
    "VaultName": "examplevault3"
   }
  ]
}
```

Beispiel: Teilweise Liste von Tresoren

Das folgende Beispiel gibt zwei Tresore zurück, beginnend mit dem vom `marker` angegebenen Tresor.

Beispielanfrage

```
GET /-/vaults?limit=2&marker=arn:aws:glacier:us-west-2:012345678901:vaults/
examplevault1 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Zwei Tresore werden in der Liste zurückgegeben. Der `Marker` enthält den ARN des Tresors, mit dem die Paginierung in einer weiteren List Vaults-Anfrage fortgesetzt wird.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: 497

{
  "Marker": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault3",
  "VaultList": [
   {
    "CreationDate": "2012-03-16T22:22:47.214Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 2,
    "SizeInBytes": 12334,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault1",
    "VaultName": "examplevault1"
   },
   {
    "CreationDate": "2012-03-19T22:06:51.218Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 0,
    "SizeInBytes": 0,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault2",
    "VaultName": "examplevault2"
   }
  ]
}
```

Zugehörige Abschnitte

- Create Vault (PUT vault)
- Delete Vault (DELETE vault)
- Initiate Job (POST jobs)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Remove Tags From Vault (POST tags remove)

Mit dieser Operation werden Tags aus der Gruppe der Tags entfernt, die einem Tresor angefügt sind. Weitere Informationen zu Tags erhalten Sie unter Taggen von Amazon Glacier-Ressourcen.

Dieser Vorgang ist idempotent. Der Vorgang wird erfolgreich ausgeführt, auch wenn dem Tresor keine Tags angefügt wurden.

Anforderungssyntax

Um Tags aus einem Tresor zu entfernen, senden Sie eine HTTP-POST-Anforderung an den URI der Tags, wie im folgenden Syntaxbeispiel gezeigt.

```
POST /AccountId/vaults/vaultName/tags?operation=remove HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
{
   "TagKeys": [
      "string",
      "string"
   ]
```

```
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Name	Beschreibung	Erforderlich
`operation=remove`	Ein einziger Abfragezeichenfolgeparameter `operation` mit dem Wert `remove` zur Unterscheidung von Add Tags To Vault (POST tags add).	Ja

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der Anforderungstext enthält die folgenden JSON-Felder.

TagKeys

Eine Liste der Tag-Schlüssel. Alle entsprechenden Tags werden aus dem Tresor entfernt.

Typ: Zeichenfolge-Array

Längenbeschränkung: Mindestens 1 Element in der Liste. Maximal 10 Elemente in der Liste.

Erforderlich: Ja

Antworten

Wenn die Aktion erfolgreich ist, gibt der Dienst die HTTP-Antwort `204 No Content` mit leerem HTTP-Textinhalt zurück.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Das folgende Beispiel zeigt die HTTP-POST-Anforderung zum Entfernen der angegebenen Tags.

```
POST /-/vaults/examplevault/tags?operation=remove HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
```

```
x-amz-glacier-version: 2012-06-01

{
   "TagsKeys": [
      "examplekey1",
      "examplekey2"
   ]
}
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier (Amazon Glacier) a zurück, `HTTP 204 No Content` wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

Zugehörige Abschnitte

- Add Tags To Vault (POST tags add)
- List Tags For Vault (GET tags)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Set Vault Access Policy (PUT access-policy)

Description

Mit diesem Vorgang wird eine Zugriffsichtlinie für einen Tresor festgelegt, wodurch gleichzeitig eine bereits vorhandene Richtlinie überschrieben wird. Senden Sie eine PUT-Anforderung an die `access-policy`-Unterressource des Tresors, um eine Tresor-Zugriffsrichtlinie zu konfigurieren. Sie können

pro Tresor eine Zugriffsrichtlinie mit einer Größe von bis zu 20 KB festlegen. Weitere Informationen zu Tresorzugriffsrichtlinien finden Sie unter Richtlinien für den Tresorzugriff.

Anforderungen

Syntax

Sie legen eine Tresor-Zugriffsrichtlinie fest, indem Sie eine HTTP-PUT-Anforderung an den URI der `access-policy`-Unterressource des Tresors senden. Orientieren Sie sich dabei an dem folgenden Syntaxbeispiel.

```
PUT /AccountId/vaults/vaultName/access-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
  "Policy": "string"
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der Anforderungstext enthält die folgenden JSON-Felder.

Richtlinie

Die Tresorzugriffsrichtlinie als eine JSON-Zeichenfolge, die "\" als Escape-Zeichen verwendet.

Typ: Zeichenfolge

Erforderlich: Ja

Antworten

Als Antwort gibt Amazon Glacier zurück, `204 No Content` wenn die Richtlinie akzeptiert wird.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanforderung

Das folgende Beispiel sendet eine HTTP-PUT-Anfrage an den URI der `access-policy`-Unterressource des Tresors. In der JSON-Zeichenfolge `Policy` wird „\“ als Escape-Zeichen verwendet.

```
PUT /-/vaults/examplevault/access-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01

{"Policy":"{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Sid\":
\"Define-owner-access-rights\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":
\"arn:aws:iam::999999999999:root\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":
\"arn:aws:glacier:us-west-2:999999999999:vaults/examplevault\"}]}"}
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier (Amazon Glacier) a zurück, `HTTP 204 No Content` wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

Zugehörige Abschnitte

- Delete Vault Access Policy (DELETE access-policy)
- Get Vault Access Policy (GET access-policy)

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Set Vault Notification Configuration (PUT notification-configuration)

Description

Das Abrufen eines Archivs und eines Tresorbestands sind asynchrone Vorgänge in Amazon Glacier (Amazon Glacier), für die Sie zuerst einen Job initiieren und warten müssen, bis der Job abgeschlossen ist, bevor Sie die Job-Ausgabe herunterladen können. Sie können einen Tresor so konfigurieren, dass er eine Mitteilung an ein Amazon Simple Notification Service (Amazon SNS)-Thema sendet, wenn diese Aufträge abgeschlossen sind. Sie können mithilfe dieser Operation Benachrichtigungen für den Tresor konfigurieren. Weitere Informationen finden Sie unter Konfiguration von Tresor-Benachrichtigungen in Amazon Glacier.

Senden Sie eine PUT-Anforderung an die Unterressource `notification-configuration` des Tresors, um Tresorbenachrichtigungen zu konfigurieren. Eine Benachrichtigungskonfiguration ist spezifisch für einen Tresor. Daher wird sie auch als eine Unterressource des Tresors bezeichnet. Die Anfrage sollte ein JSON-Dokument enthalten, das ein Amazon Simple Notification Service (Amazon SNS) -Thema und die Ereignisse enthält, für die Amazon Glacier Benachrichtigungen zu diesem Thema senden soll.

Sie können einen Tresor so konfigurieren, dass er eine Benachrichtigung für die folgenden Tresorereignisse veröffentlicht:

- **`ArchiveRetrievalCompleted`**: Dieses Ereignis tritt ein, wenn ein Auftrag, der für einen Archivabruf initiiert wurde, abgeschlossen ist (Initiate Job (POST jobs)). Der Status des abgeschlossenen Auftrags kann `Succeeded` oder `Failed` sein. Die Benachrichtigung, die an das SNS-Thema gesendet wird, ist die gleiche Ausgabe, wie sie von Describe Job (GET JobID) zurückgegeben wurde.
- **`InventoryRetrievalCompleted`**: Dieses Ereignis tritt ein, wenn ein Auftrag, der für einen Bestandsabruf initiiert wurde, abgeschlossen ist (Initiate Job (POST jobs)). Der Status des abgeschlossenen Auftrags kann `Succeeded` oder `Failed` sein. Die Benachrichtigung, die an

das SNS-Thema gesendet wird, ist die gleiche Ausgabe, wie sie von Describe Job (GET JobID) zurückgegeben wurde.

Amazon-SNS-Themen müssen die Berechtigung für den Tresor erteilen, damit Benachrichtigungen für das Thema veröffentlicht werden dürfen.

Anforderungen

Senden Sie eine PUT-Anfrage an den URI der Unterressource `notification-configuration` des Tresors, um die Benachrichtigungskonfiguration auf Ihrem Tresor einzurichten. Geben Sie die Konfiguration im Antworttext an. Die Konfiguration umfasst den Namen des Amazon-SNS-Themas und ein Array von Ereignissen, die Benachrichtigungen für jedes Thema auslösen.

Syntax

```
PUT /AccountId/vaults/VaultName/notification-configuration HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01

{
   "SNSTopic": String,
   "Events":[String, ...]
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der JSON im Anfragentext enthält die folgenden Felder.

--Ereignisse

Eine Reihe von einem oder mehreren Ereignissen, für die Amazon Glacier eine Benachrichtigung senden soll.

Gültige Werte: `ArchiveRetrievalCompleted` | `InventoryRetrievalCompleted`

Erforderlich: Ja

Typ: Array

SNSTopic

Der ARN des Amazon-SNS-Themas. Weitere Informationen finden Sie unter Erste Schritte mit Amazon SNS im Handbuch „Erste Schritte“ für Amazon Simple Notification Service.

Erforderlich: Ja

Typ: Zeichenfolge

Antworten

Als Antwort gibt Amazon Glacier (Amazon Glacier) zurück`204 No Content`, ob die Benachrichtigungskonfiguration akzeptiert wurde.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt, wie eine Tresorbenachrichtigung konfiguriert wird.

Beispielanforderung

Die folgende Anforderung richtet die `examplevault`-Benachrichtigungskonfiguration so ein, dass Benachrichtigungen für zwei Ereignisse (`ArchiveRetrievalCompleted` und `InventoryRetrievalCompleted`) an das Amazon-SNS-Thema `arn:aws:sns:us-west-2:012345678901:mytopic` gesendet werden.

```
PUT /-/vaults/examplevault/notification-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"],
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic"
}
```

Beispielantwort

Eine erfolgreiche Antwort gibt ein `204 No Content` zurück.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Get Vault Notifications (GET notification-configuration)
- Delete Vault Notifications (DELETE notification-configuration)
- Identity and Access Management für Amazon Glacier

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einem der sprachspezifischen Amazon SDKs finden Sie im Folgenden:

- AWS Command Line Interface

Archivoperationen

Im Folgenden sind die Archivierungsvorgänge aufgeführt, die in Amazon Glacier verwendet werden können.

Themen

- Delete Archive (DELETE archive)
- Upload Archive (POST archive)

Delete Archive (DELETE archive)

Beschreibung

Mit dieser Operation wird ein Archiv aus einem Tresor gelöscht. Sie können zu einem gegebenen Zeitpunkt jeweils nur ein Archiv aus einem Tresor löschen. Zum Löschen des Archivs müssen Sie dessen Archiv-ID in der Löschanforderung angeben. Sie können die Archiv-ID ermitteln, indem Sie den Tresorbestand für den Tresor, in dem sich das Archiv befindet, herunterladen. Weitere Informationen zum Herunterladen des Tresorbestands finden Sie unter Ein Tresor-Inventar in Amazon Glacier herunterladen.

Nach der Löschung eines Archivs kann noch eine erfolgreiche Anforderung zur Initiierung eines Auftrags zum Abrufen des gelöschten Archivs möglich sein, der betreffende Auftrag schlägt jedoch fehl.

Wenn Sie ein Archiv löschen und währenddessen noch ein Abruf des Archivs über die Archiv-ID bearbeitet wird, kann der Abruf den folgenden Szenarien entsprechend fehlschlagen oder gelingen:

- Wenn der Archivabruf-Job die Daten aktiv für den Download vorbereitet, während Amazon Glacier (Amazon Glacier) die Anfrage zum Löschen von Archiven empfängt, schlägt der Archivierungsabruf möglicherweise fehl.
- Wenn der Archivabruf-Job das Archiv erfolgreich für den Download vorbereitet hat, wenn Amazon Glacier die Anfrage zum Löschen des Archivs empfängt, können Sie die Ausgabe herunterladen.

Weitere Informationen zum Archivabruf finden Sie unter Ein Archiv in Amazon Glacier herunterladen.

Dieser Vorgang ist idempotent. Der Versuch, ein bereits gelöschtes Archiv zu löschen, erzeugt keinen Fehler.

Anforderungen

Zum Löschen eines Archivs senden Sie eine `DELETE`-Anforderung an den URI der Archivressource.

Syntax

```
DELETE /AccountId/vaults/VaultName/archives/ArchiveID HTTP/1.1
Host: glacier.Region.amazonaws.com
x-amz-Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Im folgenden Beispiel wird gezeigt, wie ein Archiv aus dem Tresor namens `examplevault` gelöscht wird.

Beispielanfrage

Die ID des zu löschenden Archivs wird als untergeordnete Ressource von `archives` angegeben.

```
DELETE /-/vaults/examplevault/archives/NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Wenn die Anfrage erfolgreich ist, antwortet Amazon Glacier mit `204 No Content` dem Hinweis, dass das Archiv gelöscht wurde.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Archive (POST archive)
- Identity and Access Management für Amazon Glacier

Upload Archive (POST archive)

Beschreibung

Mit dieser Operation wird ein Archiv einem Tresor hinzugefügt. Erfolgreich hochgeladene Daten werden dauerhaft gespeichert. Als Antwort gibt Amazon Glacier (Amazon Glacier) die Archiv-ID im `x-amz-archive-id` Header der Antwort zurück. Sie sollten die zurückgegeben Archiv-ID speichern, damit Sie später auf das Archiv zugreifen können.

Sie müssen einen SHA256 Struktur-Hash der Daten angeben, die Sie hochladen. Hinweise zur Berechnung eines SHA256 Baumhashs finden Sie unterBerechnen von Prüfsummen.

> **Note**
>
> Der SHA256 Baumhash ist nur für die Aktion Archiv hochladen (POST-Archiv) erforderlich, wenn die API verwendet wird. Er ist nicht erforderlich, wenn Sie den verwenden AWS CLI.

Beim Hochladen eines Archivs können Sie optional eine Archivbeschreibung mit bis zu 1 024 druckbaren ASCII-Zeichen angeben. Amazon Glacier gibt die Archivbeschreibung zurück, wenn Sie entweder das Archiv abrufen oder das Tresorinventar abrufen. Amazon Glacier interpretiert die Beschreibung in keiner Weise. Eine Archivbeschreibung muss nicht eindeutig sein. Die Beschreibung bietet keine Möglichkeit, die Archivliste abzurufen oder zu sortieren.

Mit Ausnahme der optionalen Archivbeschreibung unterstützt Amazon Glacier keine zusätzlichen Metadaten für die Archive. Die Archiv-ID ist eine opake Sequenz von Zeichen, die keine Rückschlüsse auf das Archiv erlaubt. Es bietet sich darum an, clientseitig Metadaten für die Archive zu erstellen. Weitere Informationen finden Sie unter Arbeiten mit Archiven in Amazon Glacier.

Archive sind unveränderlich. Nachdem ein Archiv hochgeladen wurde, kann weder das Archiv noch seine Beschreibung bearbeitet werden.

Anforderungen

Verwenden Sie zum Hochladen eines Archivs die HTTP-POST-Methode und richten Sie Anforderung auf die `archives`-Unterressource des Tresors aus, in dem Sie das Archiv speichern möchten. Die Anfrage muss die Größe der Archiv-Payload und die Prüfsumme (SHA256 Tree-Hash) enthalten und kann optional eine Beschreibung des Archivs enthalten.

Syntax

```
POST /AccountId/vaults/VaultName/archives
Host: glacier.Region.amazonaws.com
x-amz-glacier-version: 2012-06-01
Date: Date
Authorization: SignatureValue
x-amz-archive-description: Description
x-amz-sha256-tree-hash: SHA256 tree hash
x-amz-content-sha256: SHA256 linear hash
Content-Length: Length
```

```
<Request body.>
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Bei dieser Implementierung der Operation werden keine Anforderungsparameter verwendet.

Anforderungsheader

Bei diesem Vorgang werden zusätzlich zu den Anforderungs-Headern, die allen Vorgängen gemeinsam sind, die folgenden Anforderungs-Header verwendet. Weitere Informationen über allgemeine Anforderungsheader finden Sie unter Häufig verwendete Anforderungsheader.

Name	Beschreibung	Erforderlich
`Content-Length`	Die Größe des Objekts in Bytes. Weitere Informationen finden Sie unter http://www.w3. org/Protocols/rfc2616/rfc2616-sec14.html #sec14 .13. Typ: Zahl Standard: keiner Einschränkungen: Keine	Ja
`x-amz-archive-description`	Die optionale Beschreibung des Archivs, das Sie hochladen . Es kann sich dabei um eine Beschreibung in Textform oder	Nein

Name	Beschreibung	Erforderlich
	eine Kennzeichnung Ihrer Wahl handeln. Die Beschreibung muss sich nicht eindeutig von den Beschreibungen für andere Archive unterscheiden. Wenn Sie die Inventarliste eines Tresors abrufen (siehe Initiate Job (POST jobs)), enthält die Antwort diese Beschreibung für jedes zurückgegebene Archiv. Typ: Zeichenfolge Standard: keiner Einschränkungen: Die Beschreibung darf höchstens 1 024 Zeichen umfassen. Zulässig sind 7-Bit-ASCII-Zeichen ohne Steuercodes, also die ASCII-Dezimalwerte 32 bis 126 bzw. die ASCII-Hexadezimalwerte 0x20 bis 0x7E.	
`x-amz-content-sha256`	Die SHA256 Prüfsumme (ein linearer Hash) der Nutzlast. Dieser Wert stimmt nicht mit dem im `x-amz-sha256-tree-hash` -Header angegebenen Wert überein. Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	Ja

Name	Beschreibung	Erforderlich
`x-amz-sha256-tree-hash`	Die vom Benutzer berechnete Prüfsumme, der SHA256 Baumhash, der Nutzlast. Hinweise zur Berechnung des SHA256 Baumhashs finden Sie unter. Berechnen von Prüfsummen Wenn Amazon Glacier eine andere Prüfsumme der Nutzlast berechnet, lehnt Amazon die Anfrage ab. Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	Ja

Anforderungstext

Der Anfragetext enthält die hochzuladenden Daten.

Antworten

Als Reaktion darauf speichert Amazon Glacier das Archiv dauerhaft und gibt einen URI-Pfad zur Archiv-ID zurück.

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-sha256-tree-hash: ChecksumComputedByAmazonGlacier
Location: Location
x-amz-archive-id: ArchiveId
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Beschreibung
`Location`	Der relative URI-Pfad der neu hinzugefügten Archivressource. Typ: Zeichenfolge
`x-amz-archive-id`	Die ID des Archivs. Dieser Wert wird ebenfalls als Teil des `Location`-Headers angegeben. Typ: Zeichenfolge
`x-amz-sha256-tree-hash`	Die von Amazon Glacier berechnete Prüfsumme des Archivs. Typ: Zeichenfolge

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Das folgende Beispiel zeigt eine Anforderung zum Hochladen eines Archivs.

```
POST /-/vaults/examplevault/archives HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-sha256-tree-hash:
 beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60
x-amz-content-sha256: 7f2fe580edb35154041fa3d4b41dd6d3adaef0c85d2ff6309f1d4b520eeecda3
Content-Length: 2097152
x-amz-glacier-version: 2012-06-01
```

```
Authorization: Authorization=AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-
amz-glacier-
version,Signature=16b9a9e220a37e32f2e7be196b4ebb87120ca7974038210199ac5982e792cace

<Request body (2097152 bytes).>
```

Beispielantwort

Die nachfolgende erfolgreiche Antwort enthält einen `Location` Header, in dem Sie die ID abrufen können, die Amazon Glacier dem Archiv zugewiesen hat.

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
x-amz-sha256-tree-hash:
 beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60
Location: /111122223333/vaults/examplevault/archives/
NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
x-amz-archive-id: NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
```

Zugehörige Abschnitte

- Arbeiten mit Archiven in Amazon Glacier
- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Delete Archive (DELETE archive)
- Identity and Access Management für Amazon Glacier

Operationen für mehrteiligen Upload

Im Folgenden sind die mehrteiligen Upload-Operationen aufgeführt, die für die Verwendung in Amazon Glacier verfügbar sind.

Themen

- Abort Multipart Upload (DELETE uploadID)
- Complete Multipart Upload (POST uploadID)

- Mehrteiligen Upload initiieren (POST multipart uploads)
- List Parts (GET uploadID)
- List Multipart Uploads (GET multipart-uploads)
- Upload Part (PUT uploadID)

Abort Multipart Upload (DELETE uploadID)

Description

Mit diesem Befehl wird ein durch die Upload-ID angegebener mehrteiliger Upload abgebrochen.

Wenn die Anforderung zum Abbruch des mehrteiligen Uploads erfolgreich ist, können Sie die Upload-ID nicht mehr zum Hochladen weiterer Teile oder für andere Operationen verwenden. Der Versuch, einen abgeschlossenen mehrteiligen Upload zu beenden, schlägt fehl. Es ist allerdings für kurze Zeit möglich, einen bereits beendeten Upload-Vorgang zu beenden.

Dieser Vorgang ist idempotent.

Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

Anforderungen

Um einen mehrteiligen Upload zu beenden, senden Sie die HTTP-Anforderung `DELETE` an den URI der untergeordneten Ressource `multipart-uploads` des Tresors und geben Sie die ID des betreffenden Teils des mehrteiligen Uploads im URI an.

Syntax

```
DELETE /AccountId/vaults/VaultName/multipart-uploads/uploadID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In

diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiel

Beispielanforderung

Im folgenden Beispiel wird eine DELETE-Anforderung an den URI einer Ressource mit der ID des mehrteiligen Uploads gesendet.

```
DELETE /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE  HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Part (PUT uploadID)
- Complete Multipart Upload (POST uploadID)
- List Multipart Uploads (GET multipart-uploads)
- List Parts (GET uploadID)
- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Identity and Access Management für Amazon Glacier

Complete Multipart Upload (POST uploadID)

Beschreibung

Sie rufen diesen mehrteiligen Upload-Vorgang auf, um Amazon Glacier (Amazon Glacier) darüber zu informieren, dass alle Archivteile hochgeladen wurden und Amazon Glacier das Archiv nun aus den hochgeladenen Teilen zusammenstellen kann.

Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

Nach dem Zusammenstellen und Speichern des Archivs im Tresor gibt Amazon Glacier die Archiv-ID der neu erstellten Archivressource zurück. Nachdem Sie ein Archiv hochgeladen haben, sollten Sie die zurückgegebene Archiv-ID speichern, um das Archiv an einem späteren Zeitpunkt abrufen zu können.

In der Anfrage müssen Sie den berechneten SHA256 Baumhash des gesamten Archivs angeben, das Sie hochgeladen haben. Hinweise zur Berechnung eines SHA256 Baumhashs finden Sie unterBerechnen von Prüfsummen. Auf der Serverseite erstellt Amazon Glacier auch den SHA256 Tree-Hash des zusammengestellten Archivs. Wenn die Werte übereinstimmen, speichert Amazon Glacier das Archiv im Tresor. Andernfalls wird ein Fehler zurückgegeben und der Vorgang schlägt fehl. Die List Parts (GET uploadID)-Operation gibt eine Liste der hochgeladenen Teile für einen bestimmten mehrteiligen Upload zurück. Sie enthält die Prüfsummeninformationen für jeden hochgeladenen Teil, die zum Debuggen eines Problems mit einer falschen Prüfsumme genutzt werden können.

Darüber hinaus sucht Amazon Glacier auch nach fehlenden Inhaltsbereichen. Beim Upload von Teilen geben Sie Bereichswerte an, die bestimmen, wie die Teile in der endgültigen Zusammenstellung des Archivs zusammenpassen. Beim Zusammenstellen des endgültigen Archivs sucht Amazon Glacier nach fehlenden Inhaltsbereichen und ob Inhaltsbereiche fehlen, gibt Amazon Glacier einen Fehler zurück und der Vorgang Complete Multipart Upload schlägt fehl.

Die „Complete Multipart Upload“-Operation ist idempotent. Wenn Sie die Operation innerhalb von kurzer Zeit nach der ersten erfolgreichen „Complete Multipart Upload“-Operation erneut aufrufen, ist diese Operation erfolgreich und gibt die gleiche Archiv-ID zurück. Dies ist bei Netzwerkproblemen oder einem 500-Serverfehler nützlich, da Sie die „Complete Multipart Upload“-Anforderung in diesem Fall wiederholen können und die gleiche Archiv-ID erhalten, ohne doppelte Archive erstellen zu müssen. Beachten Sie jedoch, dass Sie die „List Parts“-Operation nach Abschluss des mehrteiligen

Uploads nicht aufrufen können, und der mehrteilige Upload nicht in der „List Multipart Uploads“-Antwort angezeigt wird. Dies ist auch dann der Fall, wenn ein idempotenter Abschluss möglich ist.

Anforderungen

Um einen mehrteiligen Upload abzuschließen, senden Sie eine HTTP-POST-Anforderung an die URI der Upload-ID, die Amazon Glacier als Antwort auf Ihre Anfrage zum Initiate Multipart Upload erstellt hat. Dies ist der gleiche URI, den Sie beim Uploads der Teile verwendet haben. Zusätzlich zu den häufig erforderlichen Headern müssen Sie das Ergebnis des SHA256 Tree-Hashs des gesamten Archivs und die Gesamtgröße des Archivs in Byte angeben.

Syntax

```
POST /AccountId/vaults/VaultName/multipart-uploads/uploadID
Host: glacier.Region.amazonaws.com
Date: date
Authorization: SignatureValue
x-amz-sha256-tree-hash: SHA256 tree hash of the archive
x-amz-archive-size: ArchiveSize in bytes
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Bei diesem Vorgang werden zusätzlich zu den Anforderungs-Headern, die allen Vorgängen gemeinsam sind, die folgenden Anforderungs-Header verwendet. Weitere Informationen über allgemeine Anforderungsheader finden Sie unter Häufig verwendete Anforderungsheader.

Name	Beschreibung	Erforderlich
`x-amz-archive-size`	Die Gesamtgröße des gesamten Archivs in Byte. Dieser Wert sollte die Summe aller Größen der einzelnen Teile sein, die Sie hochgeladen haben. Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	Ja
`x-amz-sha256-tree-hash`	Der SHA256 Baumhash des gesamten Archivs. Es ist der Baumhash des SHA256 Baumhashs der einzelnen Teile. Wenn der Wert, den Sie in der Anfrage angeben, nicht mit dem von Amazon Glacier berechneten SHA256 Baumhash des fertig zusammengestellten Archivs übereinstimmt, gibt Amazon Glacier einen Fehler zurück und die Anfrage schlägt fehl. Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	Ja

Anfrage-Elemente

Bei dieser Operation werden keine Anfrageelemente verwendet.

Antworten

Amazon Glacier (Amazon Glacier) erstellt einen SHA256 Baumhash des gesamten Archivs. Wenn der Wert mit dem SHA256 Tree-Hash des gesamten Archivs übereinstimmt, das Sie in der Anfrage angegeben haben, fügt Amazon Glacier das Archiv dem Tresor hinzu. Als Antwort wird der HTTP-`Location`-Header mit dem URL-Pfad der neu erstellten Archivressource zurückgegeben. Wenn die Archivgröße oder SHA256 die Größe, die Sie in der Anfrage gesendet haben, nicht übereinstimmt, gibt Amazon Glacier einen Fehler zurück und der Upload bleibt unvollständig. Es ist

möglich, die „Complete Multipart Upload“-Operation später mit korrekten Werten zu wiederholen. Dann können Sie erfolgreich ein Archiv erstellen. Wenn ein mehrteiliger Upload nicht abgeschlossen wird, fordert Amazon Glacier irgendwann die Upload-ID zurück.

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
x-amz-archive-id: ArchiveId
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Beschreibung
`Location`	Der relative URI-Pfad des neu erstellen Archivs. Diese URL enthält die Archiv-ID, die von Amazon Glacier generiert wird. Typ: Zeichenfolge
`x-amz-archive-id`	Die ID des Archivs. Dieser Wert wird ebenfalls als Teil des `Location`-Headers angegeben. Typ: Zeichenfolge

Antwortfelder

Dieser Vorgang gibt keinen Antworttext zurück.

Beispiel

Beispielanfrage

In diesem Beispiel wird eine HTTP POST-Anfrage an den URI gesendet, der von einer „Initiate Multipart Upload“-Anfrage zurückgegeben wurde. Die Anfrage gibt sowohl den SHA256 Tree-Hash des gesamten Archivs als auch die Gesamtgröße des Archivs an.

```
POST /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
z-amz-Date: 20170210T120000Z
x-amz-sha256-tree-hash:1ffc0f54dd5fdd66b62da70d25edacd0
x-amz-archive-size:8388608
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Die folgende Beispielantwort zeigt, dass Amazon Glacier erfolgreich ein Archiv aus den von Ihnen hochgeladenen Teilen erstellt hat. Die Antwort umfasst die Archiv-ID mit vollständigem Pfad.

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/archives/
NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
x-amz-archive-id: NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
```

Sie können jetzt HTTP-Anfragen an den URI der neu hinzugefügten Ressource/des Archivs senden. Sie können beispielsweise eine GET-Anfrage senden, um das Archiv abzurufen.

Zugehörige Abschnitte

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Part (PUT uploadID)

- Abort Multipart Upload (DELETE uploadID)
- List Multipart Uploads (GET multipart-uploads)
- List Parts (GET uploadID)
- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Delete Archive (DELETE archive)
- Identity and Access Management für Amazon Glacier

Mehrteiligen Upload initiieren (POST multipart uploads)

Description

Dieser Vorgang initiiert einen mehrteiligen Upload (siehe Hochladen von großen Archiven in Teilen (Multipart Upload)). Amazon Glacier (Amazon Glacier) erstellt eine mehrteilige Upload-Ressource und gibt ihre ID in der Antwort zurück. Sie verwenden diese Upload-ID bei allen darauf folgenden mehrteiligen Upload-Vorgängen.

Wenn Sie den mehrteiligen Upload initiieren, geben Sie die Größe der einzelnen Teile in Bytes an. Die Größe der Teile muss ein Mebibyte (MiB) (1024 Kibibyte [KiB]) multipliziert mit einer Potenz von 2 sein, z. B. 1048576 (1 MiB), 2097152 (2 MiB), 4194304 (4 MiB), 8388608 (8 MiB) usw. Der kleinste zulässige Wert für die Größe ist 1 MiB, der größte 4 GiB (Gibibytes).

Mit Ausnahme des letzten Teils muss jeder Teil, den Sie mit dieser Upload-ID hochladen, dieselbe Größe aufweisen. Der letzte Teil kann dieselbe Größe haben oder kleiner sein. Beispiel: Angenommen, Sie möchten eine Datei mit 16,2 MiB hochladen. Wenn Sie einen mehrteiligen Upload mit 4 MiB großen Teilen initiieren, laden Sie vier Teile mit jeweils 4 MiB und einen Teil mit 0,2 MiB hoch.

> **Note**
>
> Sie müssen die Größe des Archivs nicht kennen, wenn Sie einen mehrteiligen Upload starten, da Sie bei Amazon Glacier nicht die Gesamtgröße des Archivs angeben müssen.

Nachdem Sie den mehrteiligen Upload abgeschlossen haben, entfernt Amazon Glacier die mehrteilige Upload-Ressource, auf die die ID verweist. Amazon Glacier entfernt auch die mehrteilige Upload-Ressource, wenn Sie den mehrteiligen Upload abbrechen, oder sie kann entfernt werden,

wenn für einen Zeitraum von 24 Stunden keine Aktivität stattfindet. Die ID ist unter Umständen nach 24 Stunden noch vorhanden. Dieses Verhalten kann jedoch von Anwendungen nicht vorausgesetzt werden.

Anforderungen

Senden Sie zum Initiieren eines mehrteiligen Uploads eine HTTP-POST-Anforderung an den URI der `multipart-uploads`-Unterressource des Tresors, in dem Sie das Archiv speichern möchten. Die Anforderung muss die Größe der Teile enthalten und kann optional auch eine Beschreibung des Archivs beinhalten.

Syntax

```
POST /AccountId/vaults/VaultName/multipart-uploads
Host: glacier.us-west-2.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
x-amz-archive-description: ArchiveDescription
x-amz-part-size: PartSize
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Bei diesem Vorgang werden zusätzlich zu den Anforderungs-Headern, die allen Vorgängen gemeinsam sind, die folgenden Anforderungs-Header verwendet. Weitere Informationen über allgemeine Anforderungsheader finden Sie unter Häufig verwendete Anforderungsheader.

Name	Description	Erforderlich
`x-amz-part-size`	Die Größe jedes Teils mit Ausnahme des letzten Teils, in Bytes. Der letzte Teil kann kleiner als die anderen Teile sein. Typ: Zeichenfolge Standard: keiner Einschränkungen: Die Größe der Teile muss ein Mebibyte (MiB) (1024 KiB) multipliziert mit einer Potenz von 2 sein, z. B. 1048576 (1 MiB), 2097152 (2 MiB), 4194304 (4 MiB), 8388608 (8 MiB) usw. Der kleinste zulässige Wert für die Größe ist 1 MiB, der größte 4 GiB (4096 MiB).	Ja
`x-amz-archive-description`	Die Beschreibung des Archivs, das Sie in Teilen hochladen. Es kann sich dabei um eine Beschreibung in Textform oder eine eindeutige Kennzeichnung Ihrer Wahl handeln. Wenn Sie eine Inventarliste eines Tresors abrufen (siehe Initiate Job (POST jobs)), enthält die Antwort diese Beschreibung für jedes zurückgegebene Archiv. Führende Leerzeichen in Archivbeschreibungen werden entfernt. Typ: Zeichenfolge Standard: keiner Einschränkungen: Die Beschreibung darf höchstens 1 024 Bytes umfassen. Zulässig sind 7-Bit-ASCII-Zeichen ohne Steuercodes,	Nein

Name	Description	Erforderlich
	also die ASCII-Dezimalwerte 32 bis 126 bzw. die ASCII-Hexadezimalwerte 0x20 bis 0x7E.	

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

In der Antwort erstellt Amazon Glacier eine mehrteilige Upload-Ressource, die durch eine ID identifiziert wird, und gibt den relativen URI-Pfad der mehrteiligen Upload-ID zurück.

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
x-amz-multipart-upload-id: multiPartUploadId
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Description
`Location`	Der relative URI-Pfad der mehrteiligen Upload-ID, die Amazon Glacier erstellt hat. Über den URI-Pfad können Sie Ihre Anforderungen zum Hochladen von Teilen genau ausrichten und den mehrteiligen Upload abschließen. Typ: Zeichenfolge
`x-amz-multipart-upload-id`	Die ID des mehrteiligen Uploads. Dieser Wert wird ebenfalls als Teil des `Location`-Headers angegeben.

Name	Description
	Typ: Zeichenfolge

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiel

Beispielanforderung

Im folgenden Beispiel wird ein mehrteiliger Upload durch Senden einer HTTP-POST-Anforderung an den URI der `multipart-uploads`-Unterressource eines Tresors mit der Bezeichnung `examplevault` initiiert. Die Anforderung schließt Header ein, die die Größe der Teile auf 4 MiB (4 194 304 Bytes) festlegen und die optionale Archivbeschreibung enthalten.

```
POST /-/vaults/examplevault/multipart-uploads
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-archive-description: MyArchive-101
x-amz-part-size: 4194304
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Amazon Glacier erstellt eine mehrteilige Upload-Ressource und fügt sie der `multipart-uploads` Unterressource des Tresors hinzu. Der `Location`-Antwort-Header schließt den relativen URI-Pfad zur ID für den mehrteiligen Upload ein.

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
```

```
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE
x-amz-multipart-upload-id:
 OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE
```

Weitere Informationen über das Hochladen einzelner Teile finden Sie unter Upload Part (PUT uploadID).

Zugehörige Abschnitte

- Upload Part (PUT uploadID)
- Complete Multipart Upload (POST uploadID)
- Abort Multipart Upload (DELETE uploadID)
- List Multipart Uploads (GET multipart-uploads)
- List Parts (GET uploadID)
- Delete Archive (DELETE archive)
- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Identity and Access Management für Amazon Glacier

List Parts (GET uploadID)

Description

Diese mehrteilige Upload-Operation listet die Teile eines Archivs auf, die in einem spezifischen mehrteiligen Upload, der durch eine Upload-ID identifiziert wird, hochgeladen wurde. Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

Sie können diese Anfrage jederzeit stellen, während ein mehrteiliger Upload in Bearbeitung ist, bevor Sie den mehrteiligen Upload abschließen. Amazon Glacier gibt die Teileliste zurück, sortiert nach dem Bereich, den Sie beim Hochladen jedes Teils angegeben haben. Wenn Sie nach Abschluss des mehrteiligen Uploads eine Anfrage zum Auflisten von Ersatzteilen senden, gibt Amazon Glacier (Amazon Glacier) einen Fehler zurück.

Die List Parts-Operation unterstützt Paginierung. Sie sollten immer das Feld `Marker` im Antworttext auf eine Markierung überprüfen, bei der die Liste fortgesetzt werden soll. Wenn keine Elemente mehr vorhanden sind, ist der Wert des Felds `marker` gleich `null`. Wenn der nicht Null `marker` ist, haben Sie zum Abrufen des nächsten Teilesatzes eine weitere List Parts-Anfrage gesendet, wobei der Anforderungsparameter `marker` auf den Markierungswert gesetzt wurde, den Amazon Glacier als Antwort auf Ihre vorherige List Parts-Anfrage zurückgegeben hat.

Sie können die in der Antwort zurückgegebene Anzahl der Teile auch begrenzen, indem Sie in der Anfrage den Parameter `limit` angeben.

Anforderungen

Syntax

Um die Liste der Teile in einem mehrteiligen Upload, der in Bearbeitung ist, aufzulisten, senden Sie eine `GET`-Anfrage an den URI der Ressource mit der ID des mehrteiligen Uploads. Die ID des mehrteiligen Uploads wird zurückgegeben, wenn Sie einen mehrteiligen Upload initiieren (Mehrteiligen Upload initiieren (POST multipart uploads)). Sie können optional die Parameter `marker` und `limit` angeben.

```
GET /AccountId/vaults/VaultName/multipart-uploads/uploadID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Name	Description	Erforderlich
`limit`	Die maximale Anzahl der zurückzugebenden Teile. Das Standardlimit ist 50. Die Anzahl der zurückgegebenen Teile kann kleiner sein als das festgelegte Limit, aber die Anzahl der zurückgegebenen Teile überschreitet das Limit nie. Typ: Zeichenfolge Beschränkungen: Mindestganzzahlwert 1. Höchstganzzahlwert `50`.	Nein
`marker`	Eine zur Paginierung verwendete opake Zeichenfolge. `marker` gibt den Teil an, mit dem die Auflistung von Teilen beginnen soll. Rufen Sie den `marker`-Wert von der Antwort einer vorherigen List Parts-Antwort ab. Sie müssen den `marker` nur dann einschließen, wenn Sie die Paginierung der Ergebnisse fortsetzen, die in einer vorherigen List Parts-Anfrage gestartet wurde. Typ: Zeichenfolge Einschränkungen: Keine	Nein

Anforderungsheader

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
    "ArchiveDescription" : String,
    "CreationDate" : String,
    "Marker": String,
    "MultipartUploadId" : String,
    "PartSizeInBytes" : Number,
    "Parts" :
    [ {
      "RangeInBytes" : String,
      "SHA256TreeHash" : String
      },
      ...
     ],
    "VaultARN" : String
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

ArchiveDescription

Die Beschreibung des Archivs, die in der Initiate Multipart Upload-Anfrage angegeben wurde. Der Wert dieses Felds ist `null`, wenn in der Initiate Multipart Upload-Operation keine Archivbeschreibung angegeben wurde.

Typ: Zeichenfolge

CreationDate

Die UTC-Zeit, zu der der mehrteilige Upload initiiert wurde.

Typ: Zeichenfolge. Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, zum Beispiel `2013-03-20T17:03:43.221Z`.

Marker

Eine opake Zeichenfolge, die angibt, wo die Paginierung der Ergebnisse fortgesetzt werden soll. Mithilfe des `marker` in einer anderen List Parts-Anfrage erhalten Sie mehr Aufträge in der Liste. Wenn keine anderen Teile vorhanden sind, ist dieser Wert `null`.

Typ: Zeichenfolge

MultipartUploadId

Die ID des Uploads, zu dem die Teile gehören.

Typ: Zeichenfolge

PartSizeInBytes

Die Größe des Teils in Bytes. Dies ist derselbe Wert wie der, den Sie in der Initiate Multipart Upload-Anfrage angegeben haben.

Typ: Zahl

Komponenten

Eine Liste der Teilegrößen des mehrteiligen Uploads. Jedes Objekt im Array enthält ein `RangeBytes` und `sha256-tree-hash` Name-Wert-Paar.

Typ: Array

RangeInBytes

Der Byte-Bereich eines Teils, einschließlich des oberen Werts des Bereichs.

Typ: Zeichenfolge

SHA256TreeHash

Der SHA256 Baum-Hash-Wert, den Amazon Glacier für das Teil berechnet hat. Dieses Feld hat niemals den Wert `null`.

Typ: Zeichenfolge

VaultARN

Der ARN (Amazon-Ressourcenname) des Tresors, für den der mehrteilige Upload initiiert wurde.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispiel: Auflisten von Teilen eines mehrteiligen Uploads

Im folgenden Beispiel werden alle Teile eines Uploads aufgelistet. In dem Beispiel wird eine HTTP-GET-Anfrage an den URI der spezifischen ID des mehrteiligen Uploads gesendet, der in Bearbeitung ist, und es werden bis zu 1 000 Teile zurückgegeben.

Beispielanforderung

```
GET /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

In der Antwort gibt Amazon Glacier eine Liste der hochgeladenen Teile zurück, die der angegebenen mehrteiligen Upload-ID zugeordnet sind. In diesem Beispiel gibt es nur zwei Teile. Der Wert des zurückgegebenen `Marker`-Felds ist `null`. Dadurch wird angezeigt, dass keine Teile des mehrteiligen Uploads mehr vorhanden sind.

```
HTTP/1.1 200 OK
```

```
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 412

{
    "ArchiveDescription" : "archive description",
    "CreationDate" : "2012-03-20T17:03:43.221Z",
    "Marker": null,
    "MultipartUploadId" :
 "OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE",
    "PartSizeInBytes" : 4194304,
    "Parts" :
    [ {
      "RangeInBytes" : "0-4194303",
      "SHA256TreeHash" : "01d34dabf7be316472c93b1ef80721f5d4"
      },
      {
      "RangeInBytes" : "4194304-8388607",
      "SHA256TreeHash" : "0195875365afda349fc21c84c099987164"
      }],
    "VaultARN" : "arn:aws:glacier:us-west-2:012345678901:vaults/demo1-vault"
}
```

Beispiel: Auflisten von Teilen eines mehrteiligen Uploads (Angeben der Parameter Markierung und Limitanfrage)

Das folgende Beispiel zeigt, wie Sie Paginierung verwenden können, um eine begrenzte Anzahl von Ergebnissen zu erhalten. In dem Beispiel wird eine HTTP-`GET`-Anfrage an den URI der spezifischen ID eines mehrteiligen Uploads gesendet, der in Bearbeitung ist, um ein Teil zurückzugeben. Ein Start-`marker`-Parameter gibt an, bei welchem Teil die Teileliste startet. Sie können den `marker`-Wert von der Antwort einer vorherigen Anfrage für eine Teileliste abrufen. Darüber hinaus ist der `limit`-Parameter in diesem Beispiel auf 1 eingestellt und es wird ein Teil zurückgegeben. Beachten Sie, dass das `Marker`-Feld nicht `null` ist, was bedeutet, das noch mindestens ein Teil abzurufen ist.

Beispielanforderung

```
GET /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE?marker=1001&limit=1 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
```

```
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

In der Antwort gibt Amazon Glacier eine Liste der hochgeladenen Teile zurück, die mit der angegebenen ID für den mehrteiligen Upload in Bearbeitung sind, verknüpft sind.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: text/json
Content-Length: 412

{
    "ArchiveDescription" : "archive description 1",
    "CreationDate" : "2012-03-20T17:03:43.221Z",
    "Marker": "MfgsKHVjbQ6EldVl72bn3_n5h2TaGZQUO-Qb3B9j3TITf7WajQ",
    "MultipartUploadId" :
 "OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE",
    "PartSizeInBytes" : 4194304,
    "Parts" :
    [ {
      "RangeInBytes" : "4194304-8388607",
      "SHA256TreeHash" : "01d34dabf7be316472c93b1ef80721f5d4"
      }],
    "VaultARN" : "arn:aws:glacier:us-west-2:012345678901:vaults/demo1-vault"
}
```

Zugehörige Abschnitte

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Part (PUT uploadID)
- Complete Multipart Upload (POST uploadID)
- Abort Multipart Upload (DELETE uploadID)
- List Multipart Uploads (GET multipart-uploads)

- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Identity and Access Management für Amazon Glacier

List Multipart Uploads (GET multipart-uploads)

Description

Diese mehrteilige Upload-Operation listet mehrteilige Uploads auf, die für den angegebenen Tresor in Bearbeitung sind. Ein mehrteiliger Upload in Bearbeitung ist ein mehrteiliger Upload, der von einer Mehrteiligen Upload initiieren (POST multipart uploads)-Anforderung initiiert, aber noch nicht abgeschlossen oder beendet wurde. Die in der List Multipart Upload-Antwort zurückgegebene Liste hat keine garantierte Reihenfolge.

Die List Multipart Uploads-Operation unterstützt Paginierung. Standardmäßig gibt diese Operation bis zu 50 mehrteilige Uploads in der Antwort zurück. Sie sollten immer das Feld `marker` im Antworttext auf eine Markierung überprüfen, bei der die Liste fortgesetzt werden soll. Wenn keine Elemente mehr vorhanden sind, ist der Wert des Felds `marker` gleich `null`.

Wenn der nicht Null `marker` ist, haben Sie zum Abrufen des nächsten Satzes mehrteiliger Uploads eine weitere Anfrage „Mehrteilige Uploads auflisten“ gesendet, wobei der Anforderungsparameter auf den Markierungswert gesetzt wurde, den Amazon Glacier (Amazon Glacier) als Antwort auf Ihre vorherige `marker` Anfrage „Mehrteilige Uploads auflisten“ zurückgegeben hat.

Beachten Sie den Unterschied zwischen dieser Operation und der Operation List Parts (GET uploadID). Die List Multipart Uploads-Operation listet alle mehrteiligen Uploads für einen Tresor auf. Die List Parts-Operation gibt Teile eines bestimmten mehrteiligen Uploads zurück, die von einer Upload-ID identifiziert werden.

Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

Anforderungen

Syntax

Um mehrteilige Uploads aufzulisten, senden Sie eine `GET`-Anfrage an den URI der `multipart-uploads`-Unterressource des Tresors. Sie können optional die Parameter `marker` und `limit` angeben.

```
GET /AccountId/vaults/VaultName/multipart-uploads HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Name	Description	Erforderlich
`limit`	Gibt die Höchstanzahl von Uploads an, die im Antworttext zurückgegeben werden. Falls nicht angegeben, gibt die List Uploads-Operation bis zu 50 Uploads zurück. Typ: Zeichenfolge Beschränkungen: Mindestganzzahlwert 1. Höchstganzzahlwert `50`.	Nein
`marker`	Eine zur Paginierung verwendete opake Zeichenfolge. `marker` gibt den Upload an, mit dem die Auflistung von Uploads beginnen soll. Rufen Sie den Wert für `marker` aus einer vorherigen List Uploads-Antwort ab. Sie müssen `marker` nur dann einschließen, wenn Sie die Paginierung der Ergebnisse fortsetzen, die in einer vorherigen List Uploads-Anfrage gestartet wurde.	Nein

Name	Description	Erforderlich
	Typ: Zeichenfolge Einschränkungen: Keine	

Anforderungsheader

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
  "Marker": String,
  "UploadsList" : [
    {
      "ArchiveDescription": String,
      "CreationDate": String,
      "MultipartUploadId": String,
      "PartSizeInBytes": Number,
      "VaultARN": String
    },
   ...
  ]
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

ArchiveDescription

Die Beschreibung des Archivs, die in der Initiate Multipart Upload-Anfrage angegeben wurde. Der Wert dieses Felds ist `null`, wenn in der Initiate Multipart Upload-Operation keine Archivbeschreibung angegeben wurde.

Typ: Zeichenfolge

CreationDate

Die UTC-Zeit, zu der der mehrteilige Upload initiiert wurde.

Typ: Zeichenfolge. Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, zum Beispiel `2013-03-20T17:03:43.221Z`.

Marker

Eine opake Zeichenfolge, die angibt, wo die Paginierung der Ergebnisse fortgesetzt werden soll. Mithilfe des `marker` in einer anderen List Multipart Uploads-Anfrage erhalten Sie mehr Uploads in der Liste. Wenn keine anderen Uploads vorhanden sind, ist dieser Wert `null`.

Typ: Zeichenfolge

PartSizeInBytes

Die in der Anfrage Mehrteiligen Upload initiieren (POST multipart uploads) angegebene Teilgröße. Dies ist die Größe aller Teile im Upload außer dem letzten Teil, der kleiner sein kann als diese Größe.

Typ: Zahl

MultipartUploadId

Die ID des mehrteiligen Uploads.

Typ: Zeichenfolge

UploadsList

Eine Liste von Metadaten über mehrteilige Upload-Objekte. Jedes Element in der Liste enthält einen Satz von Namen-Wert-Paaren für den entsprechenden Upload, einschließlich `ArchiveDescription`, `CreationDate`, `MultipartUploadId`, `PartSizeInBytes` und `VaultARN`.

Typ: Array

VaultARN

Der ARN (Amazon-Ressourcenname) des Tresors, der das Archiv enthält.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispiel: Auflisten aller mehrteiligen Uploads

Das folgende Beispiel listet alle mehrteiligen Uploads auf, die für den Tresor in Bearbeitung sind. Das Beispiel zeigt eine HTTP-`GET`-Anfrage an den URI der `multipart-uploads`-Unterressource eines bestimmten Tresors. Weil die Parameter `marker` und `limit` in der Anfrage nicht angegeben sind, werden bis zu 1 000 mehrteilige Uploads in Bearbeitung zurückgegeben.

Beispielanforderung

```
GET /-/vaults/examplevault/multipart-uploads HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

In der Antwort gibt Amazon Glacier eine Liste aller laufenden mehrteiligen Uploads für den angegebenen Tresor zurück. Das Feld `marker` hat den Wert `null`, wodurch angezeigt wird, dass keine weiteren aufzulistenden Uploads vorhanden sind.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 1054

{
  "Marker": null,
  "UploadsList": [
    {
      "ArchiveDescription": "archive 1",
      "CreationDate": "2012-03-19T23:20:59.130Z",
      "MultipartUploadId":
 "xsQdFIRsfJr20CW2AbZBKpRZAFTZSJIMtL2hYf8mvp8dM0m4RUzlaqoEye6g3h3ecqB_zqwB7zLDMeSWhwo65re4C4Ev"
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "ArchiveDescription": "archive 2",
      "CreationDate": "2012-04-01T15:00:00.000Z",
      "MultipartUploadId": "nPyGOnyFcx67qqX7E-0tSGiRi88hHMOwOxR-
_jNyM6RjVMFfV29lFqZ3rNsSaWBugg6OP92pRtufeHdQH7ClIpSF6uJc",
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "ArchiveDescription": "archive 3",
      "CreationDate": "2012-03-20T17:03:43.221Z",
      "MultipartUploadId": "qt-RBst_7yO8gVIonIBsAxr2t-db0pE4s8MNeGjKjGdNpuU-
cdSAcqG62guwV9r5jh5mLyFPzFEitTpNE7iQfHiu1XoV",
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    }
  ]
}
```

Beispiel: Teilweise Liste mehrteiliger Uploads

Das folgende Beispiel zeigt, wie Sie Paginierung verwenden können, um eine begrenzte Anzahl von Ergebnissen zu erhalten. Das Beispiel zeigt eine HTTP-`GET`-Anfrage an den URI der `multipart-uploads`-Unterressource für einen bestimmten Tresor. In diesem Beispiel ist der Parameter `limit` auf 1 festgelegt, was bedeutet, dass nur ein Upload in der Liste zurückgegeben wird, und der Parameter `marker` zeigt die ID des mehrteiligen Uploads an, mit dem die zurückgegebene Liste beginnt.

Beispielanforderung

```
GET /-/vaults/examplevault/multipart-uploads?
limit=1&marker=xsQdFIRsfJr20CW2AbZBKpRZAFTZSJIMtL2hYf8mvp8dM0m4RUzlaqoEye6g3h3ecqB_zqwB7zLDMeSW
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

In der Antwort gibt Amazon Glacier (Amazon Glacier) eine Liste mit nicht mehr als zwei laufenden mehrteiligen Uploads für den angegebenen Tresor zurück, wobei bei der angegebenen Markierung begonnen wird und zwei Ergebnisse zurückgegeben werden.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 470

{
  "Marker": "qt-RBst_7yO8gVIonIBsAxr2t-db0pE4s8MNeGjKjGdNpuU-
cdSAcqG62guwV9r5jh5mLyFPzFEitTpNE7iQfHiu1XoV",
  "UploadsList" : [
    {
      "ArchiveDescription": "archive 2",
      "CreationDate": "2012-04-01T15:00:00.000Z",
      "MultipartUploadId": "nPyGOnyFcx67qqX7E-0tSGiRi88hHMOwOxR-
_jNyM6RjVMFfV29lFqZ3rNsSaWBugg6OP92pRtufeHdQH7ClIpSF6uJc",
```

```
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    }
  ]
}
```

Zugehörige Abschnitte

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Part (PUT uploadID)
- Complete Multipart Upload (POST uploadID)
- Abort Multipart Upload (DELETE uploadID)
- List Parts (GET uploadID)
- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Identity and Access Management für Amazon Glacier

Upload Part (PUT uploadID)

Beschreibung

Diese mehrteilige Upload-Operation lädt einen Teil eines Archivs hoch. Sie können Teile eines Archivs in beliebiger Reihenfolge hochladen, weil Sie in Ihrer Upload Part-Anfrage den Bytebereich in dem zusammengesetzten Archiv angeben, der in diesem Teil hochgeladen wird. Sie können diese Teile auch parallel hochladen. Sie können bis zu 10 000 Teile für einen mehrteiligen Upload hochladen.

Weitere Informationen über mehrteilige Uploads finden Sie unter Hochladen von großen Archiven in Teilen (Multipart Upload).

Amazon Glacier (Amazon Glacier) lehnt Ihre Anfrage zum Hochladen von Ersatzteilen ab, wenn eine der folgenden Bedingungen zutrifft:

- SHA256 Baum-Hash stimmt nicht überein — Um sicherzustellen, dass die Teildaten bei der Übertragung nicht beschädigt werden, berechnen Sie einen SHA256 Baumhash des Teils und nehmen ihn in Ihre Anfrage auf. Nach Erhalt der Bauteildaten berechnet Amazon Glacier

auch einen SHA256 Baumhash. Wenn die beiden Hash-Werte einander nicht entsprechen, schlägt die Operation fehl. Informationen zur Berechnung eines SHA256 Baumhashs finden Sie unterBerechnen von Prüfsummen.

- SHA256 Linearer Hash stimmt nicht überein — Für die Autorisierung ist dies erforderlich. Sie berechnen einen SHA256 linearen Hash der gesamten hochgeladenen Nutzlast und nehmen ihn in Ihre Anfrage auf. Hinweise zur Berechnung eines SHA256 linearen Hashs finden Sie unterBerechnen von Prüfsummen.
- Größe der Teile stimmt nicht überein: Die Größe jedes Teils mit Ausnahme des letzten muss der Größe entsprechen, die in der entsprechenden Mehrteiligen Upload initiieren (POST multipart uploads)-Anforderung angegeben ist. Die Größe des letzten Teils muss dieselbe Größe wie die angegebene Größe haben oder kleiner sein.

  > **Note**
  >
  > Wenn Sie einen Teil hochladen, dessen Größe kleiner ist als die Teilegröße, die Sie in Ihrer anfänglichen Anfrage zum mehrteiligen Upload angegeben haben, und dieser Teil nicht der letzte Teil ist, dann wird die Anfrage zum Hochladen des Teils erfolgreich sein. Die folgende Complete Multipart Upload-Anfrage jedoch wird fehlschlagen.

- Bereich stimmt nicht überein: Der Bytebereichswert in der Anforderung stimmt nicht mit der in der entsprechenden Initialisierungsanforderung angegebenen Teilgröße überein. Wenn Sie beispielsweise eine Teilgröße von 4 194 304 Byte (4 MB) angeben, sind 0 bis 4 194 303 Byte (4 MB-1) und 4 194 304 (4 MB) bis 8 388 607 (8 MB-1) gültige Teilbereiche. Wenn Sie jedoch einen Bereichswert von 2 MB bis 6 MB einrichten, stimmt der Bereich nicht mit der Teilegröße überein und der Upload schlägt fehl.

Dieser Vorgang ist idempotent. Wenn Sie denselben Teil mehrere Male hochladen, überschreiben die Daten in der jüngeren Anfrage die vorher hochgeladenen Daten.

Anforderungen

Die HTTP-PUT-Anforderung wird an den URI der Upload-ID gesendet, die von der „Initiate Multipart Upload“-Anforderung zurückgegeben wurde. Amazon Glacier verwendet die Upload-ID, um Teil-Uploads einem bestimmten mehrteiligen Upload zuzuordnen. Die Anfrage muss einen SHA256 Baumhash der Teildaten (`x-amz-SHA256-tree-hash`Header), einen SHA256 linearen Hash der gesamten Nutzlast (`x-amz-content-sha256`Header), den Bytebereich (`Content-Range`Header) und die Länge des Teils in Byte (`Content-Length`Header) enthalten.

Syntax

```
PUT /AccountId/vaults/VaultName/multipart-uploads/uploadID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Range: ContentRange
Content-Length: PayloadSize
Content-Type: application/octet-stream
x-amz-sha256-tree-hash: Checksum of the part
x-amz-content-sha256: Checksum of the entire payload
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Bei diesem Vorgang werden zusätzlich zu den Anforderungs-Headern, die allen Vorgängen gemeinsam sind, die folgenden Anforderungs-Header verwendet. Weitere Informationen über allgemeine Anforderungsheader finden Sie unter Häufig verwendete Anforderungsheader.

Name	Beschreibung	Erforderlich
`Content-Length`	Identifiziert die Länge des Teils in Bytes. Typ: Zeichenfolge Standard: keiner	Nein

Name	Beschreibung	Erforderlich
	Einschränkungen: Keine	
`Content-Range`	Identifiziert den Bytebereich im zusammeng estellten Archiv, der in diesem Teil hochgeladen wird. Amazon Glacier verwendet diese Informati onen, um das Archiv in der richtigen Reihenfolge zusammenzustellen. Das Format dieses Headers folgt RFC 2616. Ein Beispiel-Header ist: `Content-Range:bytes 0-4194303/*` . Typ: Zeichenfolge Standard: keiner Einschränkungen: Der Bereich darf nicht größer sein als die Teilegröße, die Sie bei der Initiierung des mehrteiligen Uploads angegeben haben.	Ja
`x-amz-content-sha256`	Die SHA256 Prüfsumme (ein linearer Hash) der hochgeladenen Nutzlast. Dieser Wert stimmt nicht mit dem im `x-amz-sha256-tree-hash` - Header angegebenen Wert überein. Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	Ja

Name	Beschreibung	Erforderlich
`x-amz-sha256-tree-hash`	Gibt einen SHA256 Struktur-Hash der hochgeladenen Daten an. Hinweise zur Berechnung eines SHA256 Baumhashs finden Sie unter Berechnen von Prüfsummen. Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	Ja

Anforderungstext

Der Anfragetext enthält die hochzuladenden Daten.

Antworten

Nach einem erfolgreichen Teil-Upload gibt Amazon Glacier eine `204 No Content` Antwort zurück.

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-sha256-tree-hash: ChecksumComputedByAmazonGlacier
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Beschreibung
`x-amz-sha256-tree-hash`	

Name	Beschreibung
	Der SHA256 Baumhash, den Amazon Glacier für den hochgeladenen Teil berechnet hat. Typ: Zeichenfolge

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Beispiel

Mit der folgenden Anfrage wird ein Teil von 4 MB hochgeladen. Die Anfrage stellt den Bytebereich so ein, dass dies der erste Teil im Archiv ist.

Beispielanfrage

Das Beispiel sendet eine HTTP-PUT-Anfrage zum Hochladen eines Teils von 4 MB. Die Anfrage wird an den URI der Upload-ID gesendet, die von der Initiate Multipart Upload-Anfrage zurückgegeben wurde. Der `Content-Range`-Header identifiziert den Teil als den ersten 4-MB-Datenteil des Archivs.

```
PUT /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Range:bytes 0-4194303/*
x-amz-sha256-tree-hash:c06f7cd4baacb087002a99a5f48bf953
x-amz-content-sha256:726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
Content-Length: 4194304
Authorization: Authorization=AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-
amz-glacier-
version,Signature=16b9a9e220a37e32f2e7be196b4ebb87120ca7974038210199ac5982e792cace
```

Um das nächste Teil hochzuladen, ist das Verfahren dasselbe. Sie müssen jedoch einen neuen SHA256 Tree-Hash für das Teil berechnen, das Sie hochladen, und außerdem einen neuen Bytebereich angeben, um anzugeben, wo das Teil in der Endmontage platziert werden soll. Die folgende Anfrage lädt einen anderen Teil mit derselben Upload-ID hoch. Die Anfrage gibt die nächsten 4 MB des Archivs nach der vorherigen Anfrage und eine Teilegröße von 4 MB an.

```
PUT /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Range:bytes 4194304-8388607/*
Content-Length: 4194304
x-amz-sha256-tree-hash:f10e02544d651e2c3ce90a4307427493
x-amz-content-sha256:726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
x-amz-glacier-version: 2012-06-01
Authorization: Authorization=AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20120525/
us-west-2/glacier/aws4_request, SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-
amz-glacier-version,
 Signature=16b9a9e220a37e32f2e7be196b4ebb87120ca7974038210199ac5982e792cace
```

Die Teile können in beliebiger Reihenfolge hochgeladen werden. Amazon Glacier verwendet die Bereichsspezifikation für jedes Teil, um die Reihenfolge zu bestimmen, in der sie zusammengebaut werden.

Beispielantwort

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
x-amz-sha256-tree-hash: c06f7cd4baacb087002a99a5f48bf953
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

Zugehörige Abschnitte

- Mehrteiligen Upload initiieren (POST multipart uploads)
- Upload Part (PUT uploadID)
- Complete Multipart Upload (POST uploadID)
- Abort Multipart Upload (DELETE uploadID)
- List Multipart Uploads (GET multipart-uploads)
- List Parts (GET uploadID)
- Hochladen von großen Archiven in Teilen (Multipart Upload)
- Identity and Access Management für Amazon Glacier

Auftragsoperationen

Im Folgenden sind die in Amazon Glacier verfügbaren Auftragsoperationen aufgeführt.

Themen

- Describe Job (GET JobID)
- Get Job Output (GET output)
- Initiate Job (POST jobs)
- Aufträge auflisten (GET jobs)

Describe Job (GET JobID)

Description

Dieser Vorgang gibt Informationen über einen Job zurück, den Sie zuvor initiiert haben, einschließlich des Initiierungsdatums des Jobs, des Benutzers, der den Job initiiert hat, des Job-Statuscodes/der Job-Nachricht und des Themas Amazon Simple Notification Service (Amazon SNS), das nach Abschluss des Auftrags durch Amazon Glacier (Amazon Glacier) benachrichtigt werden soll. Weitere Informationen zur Initiierung eines Auftrags finden Sie unter Initiate Job (POST jobs).

> **Note**
>
> Mit dieser Operation können Sie den Status Ihres Auftrags prüfen. Wir empfehlen Ihnen jedoch dringend, ein Amazon SNS SNS-Thema einzurichten und es in Ihrer Job-Initiation-Anfrage anzugeben, damit Amazon Glacier das Thema nach Abschluss des Jobs benachrichtigen kann.

Eine Job-ID läuft mindestens 24 Stunden nach Abschluss des Auftrags durch Amazon Glacier nicht ab.

Anforderungen

Syntax

Um Informationen zu einem Auftrag zu erhalten, verwenden Sie die HTTP-GET-Methode und beschränken Sie den Umfang der Anfrage auf den bestimmten Auftrag. Beachten Sie, dass der

relative URI-Pfad derselbe ist, den Amazon Glacier Ihnen zurückgegeben hat, als Sie den Job initiiert haben.

```
GET /AccountID/vaults/VaultName/jobs/JobID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: date
Authorization: signatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

> **Note**
>
> Wenn Sie in der Anfrage die `JobID` weglassen, gibt die Antwort eine Liste aller aktiven Aufträge für den angegebenen Tresor zurück. Weitere Informationen über das Auflisten von Aufträgen finden Sie unter Aufträge auflisten (GET jobs).

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
    "Action": "string",
    "ArchiveId": "string",
    "ArchiveSHA256TreeHash": "string",
    "ArchiveSizeInBytes": number,
    "Completed": boolean,
    "CompletionDate": "string",
    "CreationDate": "string",
    "InventoryRetrievalParameters": {
        "EndDate": "string",
        "Format": "string",
        "Limit": "string",
        "Marker": "string",
        "StartDate": "string"
    },
    "InventorySizeInBytes": number,
    "JobDescription": "string",
    "JobId": "string",
    "JobOutputPath": "string",
    "OutputLocation": {
        "S3": {
            "AccessControlList": [
                {
                    "Grantee": {
                        "DisplayName": "string",
                        "EmailAddress": "string",
                        "ID": "string",
                        "Type": "string",
                        "URI": "string"
                    },
                    "Permission": "string"
                }
            ],
            "BucketName": "string",
```

```
            "CannedACL": "string",
            "Encryption": {
                "EncryptionType": "string",
                "KMSContext": "string",
                "KMSKeyId": "string"
            },
            "Prefix": "string",
            "StorageClass": "string",
            "Tagging": {
                "string": "string"
            },
            "UserMetadata": {
                "string": "string"
            }
        }
    },
    "RetrievalByteRange": "string",
    "SelectParameters": {
        "Expression": "string",
        "ExpressionType": "string",
        "InputSerialization": {
            "csv": {
                "Comments": "string",
                "FieldDelimiter": "string",
                "FileHeaderInfo": "string",
                "QuoteCharacter": "string",
                "QuoteEscapeCharacter": "string",
                "RecordDelimiter": "string"
            }
        },
        "OutputSerialization": {
            "csv": {
                "FieldDelimiter": "string",
                "QuoteCharacter": "string",
                "QuoteEscapeCharacter": "string",
                "QuoteFields": "string",
                "RecordDelimiter": "string"
            }
        }
    },
    "SHA256TreeHash": "string",
    "SNSTopic": "string",
    "StatusCode": "string",
    "StatusMessage": "string",
```

```
    "Tier": "string",
    "VaultARN": "string"
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

Action (Aktion)

Der Auftragstyp. Er ist entweder `ArchiveRetrieval`, `InventoryRetrieval` oder `Select`.

Typ: Zeichenfolge

ArchiveId

Die Archiv-ID, die für eine Auswahl oder einen Archivabruf angefordert wurde. Andernfalls ist dieses Feld `null`.

Typ: Zeichenfolge

ArchivSHA256TreeHash

Der SHA256 Struktur-Hash des gesamten Archivs für einen Job zum Abrufen von Archiven. Für Inventarabrufaufträge ist dieses Feld `null`.

Typ: Zeichenfolge

ArchiveSizeInBytes

Für einen `ArchiveRetrieval`-Auftrag ist dies die Größe in Bytes des zum Herunterladen angeforderten Archivs. Für den `InventoryRetrieval`-Auftrag ist der Wert `null`.

Typ: Zahl

Completed

Der Auftragsstatus. Wenn ein Archiv- oder Inventarabrufauftrag abgeschlossen wurde, können Sie die Ausgabe des Auftrags über Get Job Output (GET output) abrufen.

Typ: Boolesch

CompletionDate

Uhrzeit der Fertigstellung der Auftragsanforderung in UTC (Universal Coordinated Time). Während der Auftrag in Bearbeitung ist, ist der Wert Null.

Typ: Zeichenfolge

CreationDate

Der Zeitpunkt (UTC) der Erstellung des Auftrags.

Typ: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

InventoryRetrievalParameters

Die Eingabeparameter für die Inventarabfrage für einzelne Bereiche

Typ: InventoryRetrievalJobInput Objekt

InventorySizeInBytes

Für einen `InventoryRetrieval`-Auftrag ist dies die Größe in Bytes des zum Herunterladen angeforderten Inventars. Für den `ArchiveRetrieval`- oder `Select`-Auftrag ist der Wert `null`.

Typ: Zahl

JobDescription

Die Auftragsbeschreibung, die Sie bei der Initiierung des Auftrags angegeben haben.

Typ: Zeichenfolge

JobId

Die ID, die den Job in Amazon Glacier identifiziert.

Typ: Zeichenfolge

JobOutputPath

Enthält den Speicherort für die Auftragsausgabe.

Typ: Zeichenfolge

OutputLocation

Ein Objekt, das Informationen zum Speicherort der Ergebnisse und Fehler des ausgewählten Jobs enthält

Typ: OutputLocation Objekt

RetrievalByteRange

Der abgerufene Bytebereich für Jobs zum Abrufen von Archiven in der Form "*StartByteValue*-“*EndByteValue*. Wenn Sie beim Archivabruf keinen Bereich angeben, wird das gesamte Archiv abgerufen. Dies StartByteValueentspricht ebenfalls 0 und EndByteValueentspricht der Größe des Archivs minus 1. Für Inventarabruf- und Auswahlaufträge ist dieses Feld `null`.

Typ: Zeichenfolge

SelectParameters

Ein Objekt mit Informationen zu den Parametern für eine Auswahl

Typ: SelectParameters Objekt

SHA256TreeHash

Der SHA256 Struktur-Hashwert für den angeforderten Bereich eines Archivs. Wenn in der Initiate Job (POST jobs)-Anfrage für ein Archiv ein mit dem Struktur-Hash abgeglichener Bereich angegeben wurde, gibt dieses Feld einen Wert zurück. Weitere Informationen zur Verwendung von mit dem Struktur-Hash abgeglichenen Bereichen beim Abrufen von Archivbereichen finden Sie unter Erhalten von Prüfsummen bei Daten-Download.

Wenn das gesamte Archiv abgerufen wird, ist dieser Wert mit dem `ArchiveSHA256TreeHash`-Wert identisch.

In den folgenden Situationen hat dieses Feld den Wert `null`.

- Aufträge zum Abruf von Archiven, die einen Bereich angeben, der nicht mit dem Struktur-Hash abgeglichen ist.
- Archivaufträge, die einen Bereich festlegen, der das gesamte Archiv einschließt, und deren Auftragsstatus `InProgress` lautet.
- Inventaraufträge.
- Wählen Sie die Aufträge aus.

Typ: Zeichenfolge

SNSTopic

Ein Amazon-SNS-Thema, das Benachrichtigungen empfängt

Typ: Zeichenfolge

StatusCode

Der Code, der den Status des Auftrags angibt

Zulässige Werte: `InProgress` | `Succeeded` | `Failed`

Typ: Zeichenfolge

StatusMessage

Eine freundliche Nachricht, die den Auftragsstatus beschreibt.

Typ: Zeichenfolge

Stufe

Das Datenzugriffskontingent, das für die Auswahl oder den Archivabruf verwendet wird

Zulässige Werte: `Bulk` | `Expedited` | `Standard`

Typ: Zeichenfolge

VaultARN

Der Amazon-Ressourcennamen (ARN) des Tresors, von dem der Auftrag eine Unterressource ist.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt die Anfrage für einen Auftrag, der ein Archiv abruft.

Beispielanfrage: Auftragsbeschreibung abrufen

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Der Antworttext enthält JSON zur Beschreibung des angegebenen Auftrags. Beachten Sie, dass für den Inventarabruf und für den Archivabruf die JSON-Felder gleich sind. Wenn jedoch ein Feld für den Auftragstyp nicht gültig ist, ist sein Wert `null`. Es folgt eine Beispielantwort auf einen Archivabrufauftrag. Beachten Sie Folgendes:

- Der Wert des `Action`-Felds ist `ArchiveRetrieval`.
- Das `ArchiveSizeInBytes`-Feld zeigt die Größe des im Archivabrufauftrag angeforderten Archivs.
- Das `ArchiveSHA256TreeHash` Feld zeigt den SHA256 Baumhash für das gesamte Archiv.
- Das `RetrievalByteRange`-Feld zeigt den in der Anfrage zur Auftragsinitiierung angeforderten Bereich. In diesem Beispiel wird das gesamte Archiv angefordert.
- Das `SHA256TreeHash` Feld zeigt den SHA256 Baumhash für den Bereich, der in der Anfrage „Job initiieren“ angefordert wurde. In diesem Beispiel ist es gleich dem `ArchiveSHA256TreeHash`-Feld, das heißt, dass das gesamte Archiv angefordert wurde.
- Der Wert des `InventorySizeInBytes`-Felds ist `null`, weil es für einen Archivabrufauftrag nicht gültig ist.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 419
{
  "Action": "ArchiveRetrieval",
  "ArchiveId": "NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
  "ArchiveSizeInBytes": 16777216,
  "ArchiveSHA256TreeHash":
 "beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60",
  "Completed": false,
  "CompletionDate": null,
  "CreationDate": "2012-05-15T17:21:39.339Z",
```

```
  "InventorySizeInBytes": null,
  "JobDescription": "My ArchiveRetrieval Job",
  "JobId": "HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID",
  "RetrievalByteRange": "0-16777215",
  "SHA256TreeHash": "beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60",
  "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
  "StatusCode": "InProgress",
  "StatusMessage": "Operation in progress.",
  "Tier": "Bulk",
  "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
}
```

Es folgt eine Beispielantwort auf einen Inventarabrufauftrag. Beachten Sie Folgendes:

- Der Wert des `Action`-Felds ist `InventoryRetrieval`.
- Die Werte der Felder `ArchiveSizeInBytes`, `ArchiveSHA256TreeHash` und `RetrievalByteRange` sind Null, weil diese Felder für einen Inventarabrufauftrag nicht gültig sind.
- Der Wert des `InventorySizeInBytes`-Felds ist `null`, weil der Auftrag noch in Bearbeitung ist und das Inventar noch nicht vollständig zum Herunterladen vorbereitet hat. Wenn der Auftrag vor Ihrer Anforderung zur Auftragsbeschreibung abgeschlossen war, zeigt dieses Feld Ihnen die Größe der Ausgabe an.

```
{
   "Action": "InventoryRetrieval",
   "ArchiveId": null,
   "ArchiveSizeInBytes": null,
   "ArchiveSHA256TreeHash": null,
   "Completed": false,
   "CompletionDate": null,
   "CreationDate": "2012-05-15T23:18:13.224Z",
   "InventorySizeInBytes": null,
   "JobDescription": "Inventory Description",
   "JobId": "HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID",
   "RetrievalByteRange": null,
   "SHA256TreeHash": null,
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
   "StatusCode": "InProgress",
   "StatusMessage": "Operation in progress.",
```

```
   "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
}
```

Es folgt eine Beispielantwort auf einen abgeschlossenen Bestandsabrufauftrag mit einer Markierung, durch die die Paginierung des Tresorinventarabrufs fortgesetzt wird.

```
{
    "Action": "InventoryRetrieval",
    "ArchiveId": null,
    "ArchiveSHA256TreeHash": null,
    "ArchiveSizeInBytes": null,
    "Completed": true,
    "CompletionDate": "2013-12-05T21:51:13.591Z",
    "CreationDate": "2013-12-05T21:51:12.281Z",
    "InventorySizeInBytes": 777062,
    "JobDescription": null,
    "JobId": "sCC2RZNBF2nildYD_roe0J9bHRdPQUbDRkmTdg-mXi2u3lc49uW6TcEhDF2D9pB2phx-
BN30JaBru7PMyOlfXHdStzu8",
    "NextInventoryRetrievalMarker": null,
    "RetrievalByteRange": null,
    "SHA256TreeHash": null,
    "SNSTopic": null,
    "StatusCode": "Succeeded",
    "StatusMessage": "Succeeded",
    "Tier": "Bulk",
    "VaultARN": "arn:aws:glacier-devo:us-west-2:836579025725:vaults/inventory-
icecube-2",
    "InventoryRetrievalParameters": {
        "StartDate": "2013-11-12T13:43:12Z",
        "EndDate": "2013-11-20T08:12:45Z",
        "Limit": "120000",
        "Format": "JSON",
        "Marker":
 "vyS0t2jHQe5qbcDggIeD50chS1SXwYMrkVKo0KHiTUjEYxBGCqRLKaiySzdN7QXGVVV5XZpNVG67pCZ_uykQXFMLaxOSu
    },
}
```

Zugehörige Abschnitte

- Get Job Output (GET output)
- Identity and Access Management für Amazon Glacier

Get Job Output (GET output)

Description

Diese Operation lädt die Ausgabe des Auftrags herunter, den Sie mithilfe von Initiate Job (POST jobs) initiiert haben. Abhängig von dem Auftragstyp, den Sie spezifiziert haben, als Sie den Auftrag initiierten, wird die Ausgabe entweder der Inhalt eines Archivs oder ein Tresorbestand sein.

Sie können die gesamte Auftragsausgabe oder optional nur einen Teil der Auftragsausgabe herunterladen, indem Sie einen Bytebereich festlegen. Sowohl für Archivierungs- als auch für Inventarabrufaufträge sollten Sie die heruntergeladene Größe mit der Größe vergleichen, die in den Headern von der Antwort Get Job Output response zurückgegeben wird.

Bei Archivabrufaufträgen sollten Sie auch überprüfen, ob die Größe Ihren Erwartungen entspricht. Wenn Sie einen Teil der Ausgabe herunterladen, basiert die erwartete Größe auf dem von Ihnen angegebenen Bytebereich. Wenn Sie z. B. einen Bereich von `bytes=0-1048575` angeben, sollten Sie überprüfen, ob die Größe Ihres Downloads 1 048 576 Bytes ist. Wenn Sie ein ganzes Archiv herunterladen, entspricht die erwartete Größe der Größe des Archivs, als Sie es auf Amazon Glacier (Amazon Glacier) hochgeladen haben. Die erwartete Größe wird auch in den Headern der Antwort Get Job Output zurückgegeben.

Bei einem Auftrag zum Abrufen von Archiven gibt Amazon Glacier je nach dem von Ihnen angegebenen Bytebereich die Prüfsumme für den Teil der Daten zurück. Um sicherzustellen, dass der Teil, den Sie heruntergeladen haben, die korrekten Daten enthält, berechnen Sie die Prüfsumme auf dem Client, überprüfen Sie, ob die Werte übereinstimmen und ob die Größe Ihren Erwartungen entspricht.

Eine Job-ID läuft mindestens 24 Stunden nach Abschluss des Auftrags durch Amazon Glacier nicht ab. Das heißt, Sie können die Auftragsausgabe innerhalb von 24 Stunden nach Abschluss des Auftrags durch Amazon Glacier herunterladen.

Anforderungen

Syntax

Um eine Auftragsausgabe abzurufen, senden Sie die HTTP-`GET`-Anfrage an den URI der `output` des jeweiligen Auftrags.

```
GET /AccountId/vaults/VaultName/jobs/JobID/output HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
```

```
Authorization: SignatureValue
Range: ByteRangeToRetrieve
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Bei diesem Vorgang werden zusätzlich zu den Anforderungs-Headern, die allen Vorgängen gemeinsam sind, die folgenden Anforderungs-Header verwendet. Weitere Informationen über allgemeine Anforderungsheader finden Sie unter Häufig verwendete Anforderungsheader.

Name	Description	Erforderlich
`Range`	Der Bytebereich, der von der Ausgabe abgerufen werden soll. Wenn Sie zum Beispiel die ersten 1 048 576 Bytes herunterladen möchten, geben Sie den Bereich an als `bytes=0-1048575` . Weitere Informationen finden Sie unter Range Header Field Definition. Der Bereich ist relativ zu jedem Bereich, der in der Anfrage zur Auftragsinitiierung angegeben wurde. Standardmäßig lädt diese Operation die gesamte Ausgabe herunter. Wenn die Auftragsausgabe groß ist, können Sie den Header der `Range`-Anforderung verwenden, um einen Teil der	Nein

Name	Description	Erforderlich
	Ausgabe abzurufen. Damit können Sie die gesamte Ausgabe in kleineren Byte-Blöcken herunterladen. Angenommen, Sie haben 1 GB Auftragsausgabe, die Sie herunterladen möchten, und Sie beschließen, jeweils Datenblöcke von 128 MB herunterzuladen, insgesamt also acht Anfragen zum Abrufen von Auftragsausgaben. Sie können die Auftragsausgabe mithilfe des folgenden Prozesses herunterladen: 1. Laden Sie einen Block von 128 MB der Ausgabe herunter, indem Sie den entsprechenden Bytebereich mithilfe des `Range`-Headers angeben. Überprüfen Sie, ob Sie alle 128 MB Daten erhalten haben. 2. Zusammen mit den Daten wird die Antwort eine Prüfsumme der Nutzlast enthalten. Sie berechnen die Prüfsumme der Nutzlast auf dem Client und vergleichen sie mit der Prüfsumme, die Sie in der Antwort erhalten haben, um sicherzustellen, dass Sie alle erwarteten Daten erhalten haben. 3. Wiederholen Sie die Schritte 1 und 2 für alle acht 128 MB großen Datenblöcke der Ausgabe, wobei Sie jeweils den entsprechenden Bytebereich angeben. 4. Nach dem Herunterladen aller Teile der Auftragsausgabe haben Sie eine Liste mit acht Prüfsummenwerten. Berechnen Sie den Struktur-Hash dieser Werte, um die Prüfsumme der gesamten Ausgabe zu ermitteln. Mit der Operation Describe Job (GET JobID) erhalten Sie Informationen zu dem Auftrag, der Ihnen die Ausgabe bereitgestellt hat. Die Antwort enthält die Prüfsumme des gesamten in Amazon Glacier gespeicherten Archivs. Sie vergleichen diesen Wert mit der von Ihnen berechneten Prüfsumme, um sicherzustellen, dass Sie den gesamten Archivinhalt fehlerfrei heruntergeladen haben.	

Name	Description	Erforderlich
	Typ: Zeichenfolge Standard: keiner Einschränkungen: Keine	

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

Bei einer Anfrage zum Abruf, die alle Auftragsdaten zurückgibt, gibt die Auftragsausgabeantwort einen `200 OK`-Antwortcode zurück. Wenn Teile des Inhalts angefragt werden, z. B. wenn Sie in der Anfrage den `Range`-Header angegeben haben, wird der Antwortcode `206 Partial Content` zurückgegeben.

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: ContentType
Content-Length: Length
x-amz-sha256-tree-hash: ChecksumComputedByAmazonGlacier

[Body containing job output.]
```

Antwort-Header

Header	Description
`Content-Range`	Der von Amazon Glacier zurückgegebene Bytebereich. Wenn nur ein Teil der Ausgabe heruntergeladen wird, enthält die Antwort den Bytebereich, den Amazon Glacier zurückgegeben hat.

Header	Description
	Zum Beispiel gibt `bytes 0-1048575/8388608` die ersten 1 MB von 8 MB zurück. Weitere Informationen über den `Content-Range` -Header finden Sie unter Content-Range Header Field Definition. Typ: Zeichenfolge
`Content-Type`	Der Inhaltstyp hängt davon ab, ob die Auftragsausgabe ein Archiv oder ein Tresorbestand ist. • Für Archivdaten ist der Inhaltstyp `application/octet-stream` . • Für einen Tresorbestand, wenn Sie bei der Initiierung des Auftrags das CSV-Format angefragt haben, ist der Inhaltstyp `text/csv`. Andernfalls wird der Tresorbestand standardmäßig als JSON zurückgegeben und der Inhaltstyp ist `application/json` . Typ: Zeichenfolge

Header	Description
`x-amz-sha256-tree-hash`	Die Prüfsumme der Daten in der Antwort. Dieser Header wird nur zurückgegeben, wenn die Ausgabe eines Archivabrufauftrags abgerufen wird. Darüber hinaus wird dieser Header angezeigt, wenn der abgerufene Datenbereich, der in der Anfrage zur Auftragsinitiierung angefragt wurde, mit dem Struktur-Hash abgeglichen ist und der herunterzuladende Bereich in „Get Job Output“ ebenfalls mit dem Struktur-Hash abgeglichen ist. Weitere Informationen zum Abgleichen von Bereichen mit dem Struktur-Hash finden Sie unter Erhalten von Prüfsummen bei Daten-Download. Wenn Sie z. B. in Ihrer Anfrage zur Auftragsinitiierung einen mit dem Struktur-Hash abgeglichenen Bereich angegeben haben (der das gesamte Archiv umfasst), dann erhalten Sie die Prüfsumme der von Ihnen heruntergeladenen Daten unter den folgenden Bedingungen: • Sie erhalten den gesamten Bereich der abgerufenen Daten. • Sie fordern einen Bytebereich der abgerufenen Daten an, der die Größe von einem Megabyte (1024 KB) multipliziert mit dem Faktor 2 hat und mit einem Vielfachen der Größe des angeforderten Bereichs beginnt und endet. Wenn Sie z. B. 3,1 MB an abgerufenen Daten haben und Sie einen Bereich für die Rückgabe bestimmen, der bei 1 MB beginnt und bei 2 MB endet, dann wird der `x-amz-sha256-tree-hash` als ein Antwort-Header zurückgegeben. • Sie fordern einen Bereich zur Rückgabe der abgerufenen Daten an, der bis zum Ende der Daten reicht, und der Beginn des Bereichs ist ein Vielfaches der Größe des abzurufenden Bereichs, aufgerundet bis zum nächsten Faktor von zwei, aber nicht kleiner als ein Megabyte (1024 KB). Wenn Sie z. B. 3,1 MB an abgerufenen Daten haben und Sie einen Bereich für die Rückgabe bestimmen, der bei 2 MB beginnt und bei 3.1 MB (dem Ende der Daten) endet, dann wird der `x-amz-sha256-tree-hash` als ein Antwort-Header zurückgegeben.

Header	Description
	Typ: Zeichenfolge

Antworttext

Amazon Glacier gibt die Jobausgabe im Antworttext zurück. Je nach Auftragstyp kann die Ausgabe der Inhalt eines Archivs oder der Tresorbestand sein. Im Falle eines Tresorbestands wird standardmäßig die Inventarliste als folgender JSON-Text zurückgegeben.

```
{
 "VaultARN": String,
 "InventoryDate": String,
 "ArchiveList": [
      {"ArchiveId": String,
       "ArchiveDescription": String,
       "CreationDate": String,
       "Size": Number,
       "SHA256TreeHash": String
      },
      ...
    ]
}
```

Wenn Sie das CSV-Ausgabeformat (kommaseparierte Werte) bei der Initiierung des Tresorbestandsauftrags angefordert haben, wird der Tresorbestand im CSV-Format im Text zurückgegeben. Das CSV-Format hat fünf Spalten "ArchiveId„," ArchiveDescription „," CreationDate „, „Größe“ und "SHA256TreeHash" mit denselben Definitionen wie die entsprechenden JSON-Felder.

> **Note**
>
> In dem zurückgegebenen CSV-Format können Felder zurückgegeben werden, die vollständig in doppelten Anführungszeichen eingeschlossen sind. Felder, die ein Komma oder doppelte Anführungszeichen enthalten, werden immer eingeschlossen in doppelten Anführungszeichen zurückgegeben. Zum Beispiel wird `my archive description,1` als `"my archive description,1"` zurückgegeben. Doppelte Anführungszeichen, die innerhalb von zurückgegebenen Feldern sind, die in doppelten Anführungszeichen eingeschlossen sind, werden durch einen vorangestellten umgekehrten Schrägstrich

als Escape-Zeichen geschützt. Zum Beispiel wird `my archive description,1"2` zurückgegeben als `"my archive description,1\"2"` und `my archive description,1\"2` zurückgegeben als `"my archive description,1\\"2"`. Das umgekehrte Schrägstrichzeichen wird nicht durch ein Escape-Zeichen geschützt.

Der JSON-Antworttext enthält die folgenden JSON-Felder.

ArchiveDescription

Die Beschreibung eines Archivs.

Typ: Zeichenfolge

ArchiveId

Die ID eines Archivs.

Typ: Zeichenfolge

ArchiveList

Ein Array von Archivmetadaten. Jedes Objekt in dem Array steht für Metadaten für ein im Tresor enthaltenes Archiv.

Typ: Array

CreationDate

Datum und Uhrzeit (UTC), wann das Archiv erstellt wurde.

Typ: Eine Zeichenkettendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

InventoryDate

Datum und Uhrzeit (UTC) der letzten Inventur für den Tresor, die nach Änderungen am Tresor abgeschlossen wurde. Amazon Glacier erstellt zwar einmal täglich ein Tresorinventar, das Inventardatum wird jedoch nur aktualisiert, wenn seit der letzten Inventarisierung Archive hinzugefügt oder gelöscht wurden.

Typ: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

SHA256TreeHash

Der Struktur-Hash des Archivs.

Typ: Zeichenfolge

Größe

Die Größe des Archivs in Bytes.

Typ: Zahl

VaultARN

Die ARN-Ressource (Amazon-Ressourcenname), von der der Archivabruf angefordert wurde.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel zeigt die Anfrage für einen Auftrag, der ein Archiv abruft.

Beispiel 1: Ausgabe herunterladen

In diesem Beispiel werden Daten abgerufen, die von Amazon Glacier als Antwort auf Ihre Job-Anfrage zum Initiieren des Archivabrufs vorbereitet wurden.

Beispielanforderung

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID/output
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Es folgt eine Beispielantwort eines Archivabrufauftrags. Beachten Sie, dass der `Content-Type`-Header `application/octet-stream` ist und dass der `x-amz-sha256-tree-hash`-Header in der Antwort enthalten ist, das heißt, dass alle Auftragsdaten zurückgegeben wurden.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
x-amz-sha256-tree-hash:
 beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/octet-stream
Content-Length: 1048576

[Archive data.]
```

Es folgt eine Beispielantwort eines Inventarabrufauftrags. Beachten Sie, dass der `Content-Type`-Header `application/json` ist. Beachten Sie auch, dass die Antwort nicht den `x-amz-sha256-tree-hash`-Header enthält.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 906

{
 "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault",
 "InventoryDate": "2011-12-12T14:19:01Z",
 "ArchiveList": [
   {
     "ArchiveId": "DMTmICA2n5Tdqq5BV2z7og-
A20xnpAPKt3UXwWxdWsn_D6auTUrW6kwy5Qyj9xd1MCE1mBYvMQ63LWaT8yTMzMaCxB_9VBWrW4Jw4zsvg5kehAPDVKcppU
oA",
     "ArchiveDescription": "my archive1",
     "CreationDate": "2012-05-15T17:19:46.700Z",
     "Size": 2140123,
     "SHA256TreeHash":
 "6b9d4cf8697bd3af6aa1b590a0b27b337da5b18988dbcc619a3e608a554a1e62"
   },
   {
```

```
     "ArchiveId": "2lHzwhKhgF2JHyvCS-
ZRuF08IQLuyB4265Hs3AXj9MoAIhz7tbXAvcFeHusgU_hViO1WeCBe0N5lsYYHRyZ7rrmRkNRuYrXUs_sjl2K8ume_7mKO_
uHE1oHqaW9d37pabXrSA",
     "ArchiveDescription": "my archive2",
     "CreationDate": "2012-05-15T17:21:39.339Z",
     "Size": 2140123,
     "SHA256TreeHash":
 "7f2fe580edb35154041fa3d4b41dd6d3adaef0c85d2ff6309f1d4b520eeecda3"
   }
  ]
}
```

Beispiel 2: Ausgabe nur teilweise herunterladen

In diesem Beispiel wird nur ein Teil des Archivs abgerufen, das von Amazon Glacier als Antwort auf Ihre Anfrage zum Abrufen von Archiven vorbereitet wurde. Die Anfrage verwendet den optionalen `Range`-Header, um nur die ersten 1 024 Bytes abzurufen.

Beispielanforderung

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID/output
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Range: bytes=0-1023
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Die folgende erfolgreiche Antwort zeigt die `206 Partial Content`-Antwort. In diesem Fall enthält die Antwort auch einen `Content-Range` Header, der den Bytebereich angibt, den Amazon Glacier zurückgibt.

```
HTTP/1.1 206 Partial Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Range: bytes 0-1023/8388608
Content-Type: application/octet-stream
Content-Length: 1024
```

```
[Archive data.]
```

Zugehörige Abschnitte

- Describe Job (GET JobID)
- Initiate Job (POST jobs)
- Identity and Access Management für Amazon Glacier

Initiate Job (POST jobs)

Dieser Vorgang initiiert die folgenden Typen von Amazon Glacier-Aufträgen (Amazon Glacier):

- `archive-retrieval`: Ein Archiv abrufen
- `inventory-retrieval`: Bestand eines Tresors erfassen

Themen

- Initiieren eines Archiv- oder Tresorbestandsabrufs
- Anforderungen
- Antworten
- Beispiele
- Zugehörige Abschnitte

Initiieren eines Archiv- oder Tresorbestandsabrufs

Das Abrufen eines Archivs oder des Tresorbestands ist ein asynchroner Vorgang, für den Sie einen Auftrag initiieren müssen. Nach dem Start kann die Aufgabe nicht abgebrochen werden. Das Abrufen erfolgt in zwei Schritten:

1. Initiieren Sie einen Abrufauftrag mit der Operation Initiate Job (POST jobs).

> **Important**
>
> Eine Richtlinie für den Datenabruf kann dazu führen, dass die Anforderung zur Abrufinitiierung zu einer `PolicyEnforcedException`-Ausnahme führt und fehlschlägt.

Weitere Informationen zu Datenabrufrichtlinien finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf. Weitere Informationen zur `PolicyEnforcedException`-Ausnahme finden Sie unter Fehlermeldungen.

2. Laden Sie nach Abschluss des Auftrags die Bytes mit der Operation Get Job Output (GET output) herunter.

Die Abrufanforderung wird asynchron ausgeführt. Wenn Sie einen Abrufauftrag initiieren, erstellt Amazon Glacier einen Auftrag und gibt eine Auftrags-ID in der Antwort zurück. Wenn Amazon Glacier den Auftrag abgeschlossen hat, können Sie die Auftragsausgabe (Archiv- oder Inventardaten) abrufen. Weitere Informationen über das Abrufen der Auftragsausgabe finden Sie in den Hinweisen zum Get Job Output (GET output)-Vorgang.

Der Auftrag muss abgeschlossen sein, bevor Sie die Ausgabe abrufen können. Verwenden Sie eine der folgenden Optionen, um festzustellen, wann ein Auftrag abgeschlossen ist:

- Eine Amazon SNS SNS-Benachrichtigung verwenden — Sie können ein Amazon SNS SNS-Thema angeben, zu dem Amazon Glacier nach Abschluss des Auftrags eine Benachrichtigung veröffentlichen kann. Geben Sie dazu ein SNS-Thema für jede Auftragsanforderung an. Die Benachrichtigung wird erst gesendet, nachdem Amazon Glacier den Auftrag abgeschlossen hat. Zusätzlich zur Angabe eines SNS-Themas für jede Auftragsanforderung können Sie Tresorbenachrichtigungen für einen Tresor konfigurieren, sodass Auftragsbenachrichtigungen für alle Abrufvorgänge gesendet werden. Weitere Informationen finden Sie unter Set Vault Notification Configuration (PUT notification-configuration).
- Auftragsdetails abrufen: Sie können eine Describe Job (GET JobID)-Anforderung senden, um Informationen zum Auftragsstatus zu erhalten, während der Auftrag noch ausgeführt wird. Es ist jedoch effizienter, wenn Sie sich durch eine Amazon-SNS-Benachrichtigung darüber informieren lassen, dass ein Auftrag abgeschlossen ist.

> **Note**
>
> Die Informationen, die Sie in der Benachrichtigung erhalten, sind identisch mit den Informationen, die durch den Describe Job (GET JobID)-Aufruf ermittelt werden.

Wenn Sie für ein bestimmtes Ereignis sowohl die Benachrichtigungskonfiguration im Tresor hinzufügen als auch ein SNS-Thema in Ihrer Job-Initiation-Anfrage angeben, sendet Amazon Glacier beide Benachrichtigungen. Weitere Informationen finden Sie unter Set Vault Notification Configuration (PUT notification-configuration).

Die Inventarliste des Tresors

Amazon Glacier aktualisiert ein Tresorinventar ungefähr einmal täglich, beginnend mit dem Tag, an dem Sie ein Archiv zum ersten Mal in den Tresor hochladen. Wenn dem Tresor seit der letzten Inventur keine Archive hinzugefügt oder aus ihm gelöscht wurden, wird das Inventurdatum nicht aktualisiert. Wenn Sie einen Job für ein Tresorinventar initiieren, gibt Amazon Glacier das zuletzt generierte Inventar zurück. Dabei handelt es sich um einen point-in-time Snapshot und nicht um Echtzeitdaten.

Nachdem Amazon Glacier das erste Inventar für den Tresor erstellt hat, dauert es in der Regel einen halben Tag und bis zu einem Tag, bis dieses Inventar abgerufen werden kann.

Möglicherweise finden Sie es nicht nützlich, einen Tresorbestand für jeden Archiv-Upload abzurufen. Nehmen wir jedoch an, dass Sie auf der Clientseite eine Datenbank verwalten, die Metadaten zu den Archiven verknüpft, die Sie auf Amazon Glacier hochladen. Dann finden Sie es möglicherweise nützlich, wenn die Tresorinventur die Informationen in Ihrer Datenbank nach Bedarf mit dem aktuellen Tresorbestand abgleicht. Weitere Informationen über die Datenfelder, die bei einer Inventarabfrage zurückgegeben werden, finden Sie unter Antworttext.

Inventarabfrage für einzelne Bereiche

Sie können die Anzahl der abgerufenen Tresorbestandselemente beschränken, indem Sie nach dem Archiverstellungsdatum filtern oder ein Limit festlegen.

Filtern nach dem Archiverstellungsdatum

Sie können Inventarelemente für Archive abrufen, die zwischen dem `StartDate` und dem `EndDate` erstellt wurden, indem Sie die Werte für diese Parameter in der Initiate Job-Anforderung angeben. Dadurch werden Archive zurückgegeben, die am oder nach dem `StartDate` und vor dem `EndDate` erstellt wurden. Wenn Sie nur das `StartDate` ohne das `EndDate` angeben, wird die Inventarliste für alle Archive abgerufen, die am oder nach dem `StartDate` erstellt wurden. Wenn Sie nur das `EndDate` ohne das `StartDate` angeben, wird die Inventarliste für alle Archive erstellt, die vor dem `EndDate` erstellt wurden.

Beschränken der Inventarelemente für einzelne Abrufvorgänge

Sie können die Anzahl der zurückgegebenen Inventarelemente beschränken, indem Sie den `Limit`-Parameter in der Initiate Job-Anforderung festlegen. Die Ausgabe für den Inventarauftrag enthält Inventarelemente bis zum festgelegten `Limit`. Wenn mehr Inventarelemente vorhanden sind, wird das Ergebnis paginiert. Nachdem ein Auftrag abgeschlossen ist, können Sie mithilfe der Describe Job (GET JobID)-Operation eine Markierung abrufen, die Sie für eine nachfolgende Initiate Job-Anforderung verwenden können. Die Markierung kennzeichnet den Startpunkt, ab dem die nächste Gruppe der Inventarelemente abgerufen wird. Sie können durch das gesamte Inventar blättern, indem Sie wiederholt Initiate Job-Anforderungen mit der Markierung aus der vorherigen Describe Job-Ausgabe ausführen. Verfahren Sie so, bis eine Markierung von Describe Job mit Null zurückgegeben wird. Dies bedeutet, dass keine Inventarelemente mehr verfügbar sind.

Sie können die `Limit`-Parameter zusammen mit den Datenbereichsparametern verwenden.

Abrufen eines Archivbereichs

Sie können den Abrufvorgang für das gesamte Archiv oder einen Bereich des Archivs initiieren. Beim Abrufen von Archivbereichen haben Sie die Möglichkeit einen Bytebereich anzugeben oder das gesamte Archiv abzurufen. Der Bereich muss in Megabytewerten (MB) angegeben werden. Das bedeutet, der Startwert muss 1 MB oder ein Vielfaches davon sein und der Bereichsendwert plus 1 muss durch 1 MB teilbar sein oder dem Ende des Archivs entsprechen. Wenn beim Abrufen eines Archivbereichs keine ganzen Megabytewerte angegeben werden, wird als Antwort `400` zurückgegeben. Um sicherzustellen, dass Sie Prüfsummenwerte für Daten erhalten, die Sie mit der Funktion Get Job Output (Get Job Output (GET output)) herunterladen, muss der Bereich mit dem Struktur-Hash abgeglichen sein. Weitere Informationen zum Abgleichen von Bereichen mit dem Struktur-Hash finden Sie unter Erhalten von Prüfsummen bei Daten-Download.

Expedited-, Standard- und Bulk-Kontingente

Beim Initiieren eines Archivabrufauftrags können Sie eine der folgenden Optionen im Feld `Tier` des Anforderungstexts angeben:

- **`Expedited`** – „Expedited“-Abrufe ermöglichen Ihnen den schnellen Zugriff auf Ihre Daten, wenn gelegentliche dringende Anforderungen zum Wiederherstellen von Archiven erforderlich sind. Daten, die unter Verwendung des "Expedited"-Kontingents abgerufen werden, stehen normalerweise innerhalb von 1 bis 5 Minuten zur Verfügung, außer wenn es sich um sehr große Archive (250 MB und mehr) handelt.
- **`Standard`** – Mit „Standard“-Abrufen können Sie innerhalb von mehreren Stunden auf jedes Ihrer Archive zugreifen. Daten, die über das Standardkontingent abgerufen werden, sind in der Regel

innerhalb von drei bis fünf Stunden verfügbar. Diese Option ist für Auftragsanforderungen, für die kein Kontingent angegeben wird, das Standardkontingent.

- **`Bulk`**— Bulk ist die kostengünstigste Stufe für Amazon Glacier, mit der Sie große Datenmengen, sogar Petabyte, kostengünstig an einem Tag abrufen können. Daten, die über das Bulk-Kontingent abgerufen werden, sind in der Regel innerhalb von 5 bis 12 Stunden verfügbar.

Weitere Informationen zu Expedited- und Bulk-Abrufen finden Sie unter Amazon Glacier-Archive abrufen.

Anforderungen

Verwenden Sie zum Initiieren eines Auftrags die HTTP-`POST`-Methode und beschränken Sie den Umfang der Anforderung auf die `jobs`-Unterressource des Tresors. Sie legen die Details der Auftragsanforderung im JSON-Dokument für Ihre Anforderung fest. Der Auftragstyp wird im Feld `Type` angegeben. Optional können Sie ein `SNSTopic` Feld angeben, um ein Amazon SNS SNS-Thema anzugeben, zu dem Amazon Glacier nach Abschluss des Auftrags eine Benachrichtigung senden kann.

> **Note**
>
> Damit eine Benachrichtigung an Amazon SNS übermittelt wird, müssen Sie das Thema selbst erstellen, falls es noch nicht vorhanden ist. Amazon Glacier erstellt das Thema nicht für Sie. Das Thema muss über Berechtigungen zum Empfangen von Veröffentlichungen aus einem Amazon Glacier-Tresor verfügen. Amazon Glacier überprüft nicht, ob der Tresor berechtigt ist, das Thema zu veröffentlichen. Wenn die Berechtigungen nicht ordnungsgemäß konfiguriert sind, erhalten Sie unter Umständen selbst nach Beendigung des Auftrags keine Benachrichtigung.

Syntax

Nachfolgend sehen Sie die Anforderungssyntax für das Initiieren eines Auftrags.

```
POST /AccountId/vaults/VaultName/jobs HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

```
{
   "jobParameters": {
      "ArchiveId": "string",
      "Description": "string",
      "Format": "string",
      "InventoryRetrievalParameters": {
         "EndDate": "string",
         "Limit": "string",
         "Marker": "string",
         "StartDate": "string"
      },
      "OutputLocation": {
         "S3": {
            "AccessControlList": [
               {
                  "Grantee": {
                     "DisplayName": "string",
                     "EmailAddress": "string",
                     "ID": "string",
                     "Type": "string",
                     "URI": "string"
                  },
                  "Permission": "string"
               }
            ],
            "BucketName": "string",
            "CannedACL": "string",
            "Encryption": {
               "EncryptionType": "string",
               "KMSContext": "string",
               "KMSKeyId": "string"
            },
            "Prefix": "string",
            "StorageClass": "string",
            "Tagging": {
               "string" : "string"
            },
            "UserMetadata": {
               "string" : "string"
            }
         }
      },
      "RetrievalByteRange": "string",
```

```
      "SelectParameters": {
         "Expression": "string",
         "ExpressionType": "string",
         "InputSerialization": {
            "csv": {
               "Comments": "string",
               "FieldDelimiter": "string",
               "FileHeaderInfo": "string",
               "QuoteCharacter": "string",
               "QuoteEscapeCharacter": "string",
               "RecordDelimiter": "string"
            }
         },
         "OutputSerialization": {
            "csv": {
               "FieldDelimiter": "string",
               "QuoteCharacter": "string",
               "QuoteEscapeCharacter": "string",
               "QuoteFields": "string",
               "RecordDelimiter": "string"
            }
         }
      },
      "SNSTopic": "string",
      "Tier": "string",
      "Type": "string"
   }
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf diese keine Bindestriche („-“) enthalten.

Anforderungstext

Die Anforderung akzeptiert die folgenden Daten im JSON-Format im Text der Anforderung.

jobParameters

Bietet Optionen für die Angabe von Auftragsinformationen.

Typ: jobParameters Objekt

Erforderlich: Ja

Antworten

Amazon Glacier schafft den Job. In der Antwort wird der URI des Auftrags zurückgegeben.

Syntax

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: location
x-amz-job-id: jobId
x-amz-job-output-path: jobOutputPath
```

Antwort-Header

Header	Description
`Location`	Der relative URI-Pfad des Auftrags. Sie können diesen URI-Pfad verwenden , um den Auftragsstatus zu ermitteln. Weitere Informationen finden Sie unter Describe Job (GET JobID). Typ: Zeichenfolge Standard: keiner
`x-amz-job-id`	Die ID des Auftrags. Dieser Wert wird ebenfalls als Teil des `Location`-Headers angegeben. Typ: Zeichenfolge Standard: keiner

Header	Description
`x-amz-job-output-path`	Der Pfad des Speicherorts, an dem die Auswahlergebnisse gespeichert werden Typ: Zeichenfolge Standard: keiner

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Dieser Vorgang umfasst zusätzlich zu den möglichen Fehlern, die allen Amazon Glacier-Vorgängen gemeinsam sind, die folgenden Fehler oder Fehler. Informationen zu Amazon Glacier-Fehlern und eine Liste der Fehlercodes finden Sie unterFehlermeldungen.

Code	Description	HTTP-Statuscode	Typ
`InsufficientCapacityException`	Wird zurückgegeben, wenn die vorhandene Kapazität nicht ausreicht, um diese beschleunigte Anforderung zu verarbeiten. Dieser Fehler gilt nur für beschleunigte Abrufe und nicht für Standard- oder Massenabrufe.	`503 Service Unavailable`	Server

Beispiele

Beispielanforderung: Initiieren eines Auftrags zum Abrufen eines Archivs

```
POST /-/vaults/examplevault/jobs HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
```

```
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Type": "archive-retrieval",
  "ArchiveId": "NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
  "Description": "My archive description",
  "SNSTopic": "arn:aws:sns:us-west-2:111111111111:Glacier-ArchiveRetrieval-topic-
Example",
  "Tier" : "Bulk"
}
```

Das folgende Beispiel zeigt den Text einer Anforderung, durch die ein abzurufender Archivbereich mithilfe des `RetrievalByteRange`-Feldes festgelegt wird.

```
{
  "Type": "archive-retrieval",
  "ArchiveId": "NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
  "Description": "My archive description",
  "RetrievalByteRange": "2097152-4194303",
  "SNSTopic": "arn:aws:sns:us-west-2:111111111111:Glacier-ArchiveRetrieval-topic-
Example",
  "Tier" : "Bulk"
}
```

Beispielantwort

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-id: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

Beispielanforderung: Initiieren eines Auftrags zum Abrufen des Inventars

Die folgende Anforderung initiiert einen Auftrag für eine Inventarabfrage, durch die eine Liste der Archive im `examplevault`-Tresor angefordert wird. Da das `Format` im Text der Anforderung auf `CSV` festgelegt ist, wird das Inventar im CSV-Format zurückgegeben.

```
POST /-/vaults/examplevault/jobs HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Content-Type: application/x-www-form-urlencoded
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Type": "inventory-retrieval",
  "Description": "My inventory job",
  "Format": "CSV",
  "SNSTopic": "arn:aws:sns:us-west-2:111111111111:Glacier-InventoryRetrieval-topic-
Example"
}
```

Beispielantwort

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-id: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

Beispielanforderungen: Initiieren eines Inventarabrufs unter Verwendung von Datenfiltern mit festem Limit und einer Folgeanforderung zum Abrufen der nächsten Seite der Inventarelemente

Die folgende Anforderung initiiert einen Auftrag zum Abrufen des Tresorbestands durch Filtern der Daten und Festlegen eines Limits.

```
{
```

```
    "ArchiveId": null,
    "Description": null,
    "Format": "CSV",
    "RetrievalByteRange": null,
    "SNSTopic": null,
    "Type": "inventory-retrieval",
    "InventoryRetrievalParameters": {
        "StartDate": "2013-12-04T21:25:42Z",
        "EndDate": "2013-12-05T21:25:42Z",
        "Limit" : "10000"
    },
}
```

Die folgende Anforderung ist ein Beispiel für eine Folgeanforderung zum Abrufen der nächsten Seite der Inventarelemente mithilfe einer Markierung, die durch Describe Job (GET JobID) übermittelt wurde.

```
{
    "ArchiveId": null,
    "Description": null,
    "Format": "CSV",
    "RetrievalByteRange": null,
    "SNSTopic": null,
    "Type": "inventory-retrieval",
    "InventoryRetrievalParameters": {
        "StartDate": "2013-12-04T21:25:42Z",
        "EndDate": "2013-12-05T21:25:42Z",
        "Limit": "10000",
        "Marker":
 "vyS0t2jHQe5qbcDggIeD50chS1SXwYMrkVKo0KHiTUjEYxBGCqRLKaiySzdN7QXGVVV5XZpNVG67pCZ_uykQXFMLaxOSu
    },
}
```

Beispielantwort

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-id: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

```
x-amz-job-output-path: test/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID/
```

Zugehörige Abschnitte

- Describe Job (GET JobID)
- Get Job Output (GET output)
- Identity and Access Management für Amazon Glacier

Aufträge auflisten (GET jobs)

Description

Durch diese Operation wird eine Liste der Aufträge für einen Tresor erstellt, einschließlich der Aufträge, die gerade ausgeführt werden, und der vor kurzem abgeschlossenen Aufträge.

> **Note**
>
> Amazon Glacier (Amazon Glacier) speichert kürzlich abgeschlossene Jobs für einen bestimmten Zeitraum, bevor sie gelöscht werden. Schließlich werden jedoch abgeschlossene Jobs entfernt. Die Ausgabe der abgeschlossenen Aufträge kann abgerufen werden. Da abgeschlossene Aufträge nicht sofort entfernt werden, haben Sie die Möglichkeit, die Auftragsausgabe auch dann noch herunterzuladen, wenn Sie die Benachrichtigung über den Abschluss des Auftrags verpasst haben oder wenn Ihr erster Versuch, die Daten herunterzuladen, nicht gelingt. Angenommen, Sie starten einen Auftrag zum Abrufen eines Archivs, um ein Archiv herunterzuladen. Sie starten den Download, nachdem der Auftrag abgeschlossen ist, aber gerade in diesem Moment tritt ein Netzwerkfehler auf. Solange der Auftrag noch vorhanden ist, können Sie erneut versuchen, das Archiv herunterzuladen.

Die `List Jobs`-Operation unterstützt Paginierung. Sie sollten immer das `Marker`-Antwortfeld überprüfen. Wenn keine Aufträge mehr vorhanden sind, wird das `Marker`-Feld auf `null` gesetzt. Wenn weitere Aufträge aufgeführt werden können, zeigt das `Marker`-Feld einen anderen Wert als null an. Sie können diesen Wert verwenden, um die Paginierung der Liste fortzusetzen. Wenn eine Liste aller Aufträge ab einem bestimmten Auftrag zurückgegeben werden soll, legen Sie den `marker`-

Anforderungsparameter auf den `Marker`-Wert fest, den Sie für den entsprechenden Auftrag bei einer vorherigen `List Jobs`-Anforderung ermittelt haben.

Mit dem `limit`-Parameter in der Anforderung können Sie eine Höchstgrenze für die Anzahl der Aufträge festlegen, die in der Antwort zurückgegeben werden. Das Standardlimit ist 50. Die Anzahl der zurückgegebenen Aufträge kann kleiner sein als das festgelegte Limit, aber die Anzahl der zurückgegebenen Aufträge überschreitet das Limit nie.

Darüber hinaus können Sie die zurückgegebene Auftragsliste filtern, indem Sie wahlweise den Parameter `statuscode` und/oder den Parameter `completed` festlegen. Bei Verwendung des Parameters `statuscode` können Sie weiterhin angeben, dass nur die Aufträge zurückgegeben werden sollen, die einen der folgenden Status aufweisen: `InProgress`, `Succeeded` oder `Failed`. Bei Verwendung des Parameters `completed` können Sie angeben, dass entweder nur abgeschlossene (`true`) oder nur die noch nicht abgeschlossenen Aufträge (`false`) zurückgegeben werden sollen.

Anforderungen

Syntax

Um eine Liste der Aufträge jedes Typs abzurufen, senden Sie eine `GET`-Anforderung an den URI der `jobs`-Unterressource des Tresors.

```
GET /AccountId/vaults/VaultName/jobs HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID des Kontos, dem der Tresor gehört. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie eine Konto-ID verwenden, darf die ID keine Bindestriche („-“) enthalten.

Anforderungsparameter

Name	Description	Erforderlich
`completed`	Der Status der zurückzugebenden Aufträge. Sie können `true` oder `false` angeben. Typ: Boolescher Wert Einschränkungen: Keine	Nein
`limit`	Die maximale Anzahl der zurückzugebenden Aufträge. Das Standardlimit ist 50. Die Anzahl der zurückgegebenen Aufträge kann kleiner sein als das festgelegte Limit, aber die Anzahl der zurückgegebenen Aufträge überschreitet das Limit nie. Typ: Zeichenfolge Beschränkungen: Mindestganzzahlwert 1. Höchstganzzahlwert 50.	Nein
`marker`	Eine opake Zeichenfolge für die Paginierung, durch die der Auftrag festgelegt wird, mit dem die Liste der Aufträge beginnen soll. Sie können den Wert für `marker` der Antwort auf eine vorherige `List Jobs`-Anforderung entnehmen. Sie müssen `marker` nur dann einschließen, wenn Sie die Paginierung der Ergebnisse fortsetzen, die in einer vorherigen „“-Anforderung gestartet wurde.`List Jobs` Typ: Zeichenfolge Einschränkungen: Keine	Nein
`statuscode`	Der Auftragsstatus, der zurückgegeben werden soll. Typ: Zeichenfolge	Nein

Name	Description	Erforderlich
	Einschränkungen: Einer der folgenden Werte: `InProgress` , `Succeeded` oder `Failed`.	

Anforderungsheader

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
Content-Type: application/json
Content-Length: Length

{
    "JobList": [
        {
            "Action": "string",
            "ArchiveId": "string",
            "ArchiveSHA256TreeHash": "string",
            "ArchiveSizeInBytes": number,
            "Completed": boolean,
            "CompletionDate": "string",
            "CreationDate": "string",
            "InventoryRetrievalParameters": {
                "EndDate": "string",
                "Format": "string",
                "Limit": "string",
                "Marker": "string",
```

```
            "StartDate": "string"
        },
        "InventorySizeInBytes": number,
        "JobDescription": "string",
        "JobId": "string",
        "JobOutputPath": "string",
        "OutputLocation": {
            "S3": {
                "AccessControlList": [
                    {
                        "Grantee": {
                            "DisplayName": "string",
                            "EmailAddress": "string",
                            "ID": "string",
                            "Type": "string",
                            "URI": "string"
                        },
                        "Permission": "string"
                    }
                ],
                "BucketName": "string",
                "CannedACL": "string",
                "Encryption": {
                    "EncryptionType": "string",
                    "KMSContext": "string",
                    "KMSKeyId": "string"
                },
                "Prefix": "string",
                "StorageClass": "string",
                "Tagging": {
                    "string": "string"
                },
                "UserMetadata": {
                    "string": "string"
                }
            }
        },
        "RetrievalByteRange": "string",
        "SelectParameters": {
            "Expression": "string",
            "ExpressionType": "string",
            "InputSerialization": {
                "csv": {
                    "Comments": "string",
```

```
                        "FieldDelimiter": "string",
                        "FileHeaderInfo": "string",
                        "QuoteCharacter": "string",
                        "QuoteEscapeCharacter": "string",
                        "RecordDelimiter": "string"
                    }
                },
                "OutputSerialization": {
                    "csv": {
                        "FieldDelimiter": "string",
                        "QuoteCharacter": "string",
                        "QuoteEscapeCharacter": "string",
                        "QuoteFields": "string",
                        "RecordDelimiter": "string"
                    }
                }
            },
            "SHA256TreeHash": "string",
            "SNSTopic": "string",
            "StatusCode": "string",
            "StatusMessage": "string",
            "Tier": "string",
            "VaultARN": "string"
        }
    ],
    "Marker": "string"
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

JobList

Eine Liste der Auftragsobjekte. Jedes Auftragsobjekt enthält Metadaten, die den Auftrag beschreiben.

Typ: Array von GlacierJobDescription-Objekten

Marker

Eine opake Zeichenfolge, die angibt, wo die Paginierung der Ergebnisse fortgesetzt werden soll. Verwenden Sie den `marker`-Wert in einer neuen `List Jobs`-Anforderung, um weitere Aufträge in der Liste zu erhalten. Wenn keine weiteren Aufträge vorhanden sind, ist dieser Wert `null`.

Typ: Zeichenfolge

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Die folgenden Beispiele zeigen, wie Informationen über Tresoraufträge zurückgegeben werden. Im ersten Beispiel wird eine Liste mit zwei Aufträgen zurückgegeben, während das zweite Beispiel zeigt, wie eine Untermenge von Aufträgen zurückgegeben wird.

Beispiel: Alle Aufträge zurückgeben

Beispielanforderung

Die folgende `GET`-Anforderung gibt die Aufträge für einen Tresor zurück.

```
GET /-/vaults/examplevault/jobs  HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Die folgende Antwort enthält einen Auftrag zum Abrufen eines Archivs und einen Tresorbestandsabrufauftrag mit einer Markierung, durch die die Paginierung des Tresorbestandsabrufs fortgesetzt wird. Die Antwort zeigt außerdem, dass das `Marker`-Feld auf `null` gesetzt wurde, wodurch angezeigt wird, dass keine weiteren Aufträge vorhanden sind.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
```

```
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 1444

{
  "JobList": [
    {
      "Action": "ArchiveRetrieval",
      "ArchiveId": "BDfaUQul0dVzYwAMr8YSa_6_8abbhZq-
i1oT69g8ByClfJyBgAGBkWl2QbF5os851P7Y7KdZDOHWJIn4rh1ZHaOYD3MgFhK_g0oDPesW34uHQoVGwoIqubf6BgUEfQm
      "ArchiveSizeInBytes": 1048576,
      "ArchiveSHA256TreeHash":
 "25499381569ab2f85e1fd0eb93c5406a178ab77c5933056eb5d6e7d4adda609b",
      "Completed": true,
      "CompletionDate": "2012-05-01T00:00:09.304Z",
      "CreationDate": "2012-05-01T00:00:06.663Z",
      "InventorySizeInBytes": null,
      "JobDescription": null,
      "JobId": "hDe9t9DTHXqFw8sBGpLQQOmIM0-
JrGtu1O_YFKLnzQ64548qJc667BRWTwBLZC76Ygy1jHYruqXkdcAhRsh0hYv4eVRU",
      "RetrievalByteRange": "0-1048575",
      "SHA256TreeHash":
 "25499381569ab2f85e1fd0eb93c5406a178ab77c5933056eb5d6e7d4adda609b",
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "Tier": "Bulk",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "Action": "InventoryRetrieval",
      "ArchiveId": null,
      "ArchiveSizeInBytes": null,
      "ArchiveSHA256TreeHash": null,
      "Completed": true,
      "CompletionDate": "2013-05-11T00:25:18.831Z",
      "CreationDate": "2013-05-11T00:25:14.981Z",
      "InventorySizeInBytes": 1988,
      "JobDescription": null,
      "JobId":
 "2cvVOnBL36btzyP3pobwIceiaJebM1bx9vZOOUtmNAr0KaVZ4WkWgVjiPldJ73VU7imlm0pnZriBVBebnqaAcirZq_C5"
      "RetrievalByteRange": null,
      "SHA256TreeHash": null,
      "SNSTopic": null,
```

```
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
      "InventoryRetrievalParameters": {
          "StartDate": "2013-11-12T13:43:12Z",
          "EndDate": "2013-11-20T08:12:45Z",
          "Limit": "120000",
          "Format": "JSON",
          "Marker":
 "vyS0t2jHQe5qbcDggIeD50chS1SXwYMrkVKo0KHiTUjEYxBGCqRLKaiySzdN7QXGVVV5XZpNVG67pCZ_uykQXFMLaxOSu
    }
  ],
  "Marker": null
}
```

Beispiel: Teilliste der Aufträge zurückgeben

Beispielanforderung

Mit der folgenden GET-Anforderung wird der Auftrag zurückgegeben, der durch den `marker`-Parameter festgelegt wurde. Wenn der `limit`-Parameter auf gesetzt wird, werden bis zu zwei Aufträge zurückgegeben.2

```
GET /-/vaults/examplevault/jobs?marker=HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID&limit=2
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Die folgende Antwort zeigt zwei zurückgegebene Aufträge und das `Marker`-Feld enthält einen anderen Wert als null. Dieser Wert kann verwendet werden, um die Paginierung der Auftragsliste fortzusetzen.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
```

```
Content-Length: 1744

{
  "JobList": [
    {
      "Action": "ArchiveRetrieval",
      "ArchiveId": "58-3KpZfcMPUznvMZNPaKyJx9wODCsWTnqcjtx2CjKZ6b-
XgxEuA8yvZOYTPQfd7gWR4GRm2XR08gcnWbLV4VPV_kDWtZJKi0TFhKKVPzwrZnA4-
FXuIBfViYUIVveeiBE51FO4bvg",
      "ArchiveSizeInBytes": 8388608,
      "ArchiveSHA256TreeHash":
 "106086b256ddf0fedf3d9e72f461d5983a2566247ebe7e1949246bc61359b4f4",
      "Completed": true,
      "CompletionDate": "2012-05-01T00:25:20.043Z",
      "CreationDate": "2012-05-01T00:25:16.344Z",
      "InventorySizeInBytes": null,
      "JobDescription": "aaabbbccc",
      "JobId": "s4MvaNHIh6mOa1f8iY4ioG2921SDPihXxh3Kv0FBX-
JbNPctpRvE4c2_BifuhdGLqEhGBNGeB6Ub-JMunR9JoVa8y1hQ",
      "RetrievalByteRange": "0-8388607",
      "SHA256TreeHash":
 "106086b256ddf0fedf3d9e72f461d5983a2566247ebe7e1949246bc61359b4f4",
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "Tier": "Bulk",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "Action": "ArchiveRetrieval",
      "ArchiveId": "2NVGpf83U6qB9M2u-
Ihh61yoFLRDEoh7YLZWKBn80A2i1xG8uieBwGjAr4RkzOHA0E07ZjtI267R03Z-6Hxd8pyGQkBdciCSH1-
Lw63Kx9qKpZbPCdU0uTW_WAdwF6lR6w8iSyKdvw",
      "ArchiveSizeInBytes": 1048576,
      "ArchiveSHA256TreeHash":
 "3d2ae052b2978727e0c51c0a5e32961c6a56650d1f2e4ceccab6472a5ed4a0",
      "Completed": true,
      "CompletionDate": "2012-05-01T16:59:48.444Z",
      "CreationDate": "2012-05-01T16:59:42.977Z",
      "InventorySizeInBytes": null,
      "JobDescription": "aaabbbccc",
      "JobId":
 "CQ_tf6fOR4jrJCL61Mfk6VM03oY8lmnWK93KK4gLig1UPAbZiN3UV4G_5nq4AfmJHQ_dOMLOX5k8ItFv0wCPN0oaz5dG"
      "RetrievalByteRange": "0-1048575",
```

```
      "SHA256TreeHash":
 "3d2ae052b2978727e0c51c0a5e32961c6a56650d1f2e4ceccab6472a5ed4a0",
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "Tier": "Standard",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    }
  ],
  "Marker":
 "CQ_tf6fOR4jrJCL61Mfk6VM03oY8lmnWK93KK4gLig1UPAbZiN3UV4G_5nq4AfmJHQ_dOMLOX5k8ItFv0wCPN0oaz5dG"
}
```

Zugehörige Abschnitte

- Describe Job (GET JobID)
- Identity and Access Management für Amazon Glacier

In Auftragsoperationen verwendete Datentypen

Die folgenden Datentypen werden bei den Auftragsvorgängen in Amazon Glacier verwendet.

Themen

- CSVInput
- CSVOutput
- Verschlüsselung
- GlacierJobDescription
- Gewährung
- Grantee
- InputSerialization
- InventoryRetrievalJobInput
- jobParameters
- OutputLocation
- OutputSerialization
- S3Location

- SelectParameters

CSVInput

Enthält Informationen zur CSV-Datei.

Inhalt

Kommentare

Ein einzelnes Zeichen, mit dem angegeben wird, dass eine Zeile ignoriert werden soll, wenn das Zeichen am Anfang der Reihe vorhanden ist

Typ: Zeichenfolge

Erforderlich: Nein

FieldDelimiter

Ein einzelnes Zeichen, das zum Trennen einzelner Felder innerhalb eines Datensatzes verwendet wird Das Zeichen muss `\n`, `\r` oder ein ASCII-Zeichen im Bereich 32–126 sein. Standardmäßig wird ein Komma (`,`) verwendet.

Typ: Zeichenfolge

Standard: ,

Erforderlich: Nein

FileHeaderInfo

Ein Wert zur Beschreibung, wie mit der ersten Zeile der Eingabe verfahren werden soll

Typ: Zeichenfolge

Zulässige Werte: `Use` | `Ignore` | `None`

Erforderlich: Nein

QuoteCharacter

Ein einzelnes Zeichen, das als Escape-Zeichen verwendet wird, wenn das Feldtrennzeichen Teil des Werts ist

Typ: Zeichenfolge

Erforderlich: Nein

QuoteEscapeCharacter

Ein einzelnes Zeichen, das als Escape-Zeichen für Anführungszeichen innerhalb eines Werts verwendet wird, in dem bereits ein Escape-Zeichen verwendet wird

Typ: Zeichenfolge

Erforderlich: Nein

RecordDelimiter

Ein einzelnes Zeichen, das zum Trennen einzelner Datensätze voneinander verwendet wird

Typ: Zeichenfolge

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

CSVOutput

Enthält Informationen zum CSV-Format, in dem die Auftragsergebnisse gespeichert sind.

Inhalt

FieldDelimiter

Ein einzelnes Zeichen, das zum Trennen einzelner Felder innerhalb eines Datensatzes verwendet wird

Typ: Zeichenfolge

Erforderlich: Nein

QuoteCharacter

Ein einzelnes Zeichen, das als Escape-Zeichen verwendet wird, wenn das Feldtrennzeichen Teil des Werts ist

Typ: Zeichenfolge

Erforderlich: Nein

QuoteEscapeCharacter

Ein einzelnes Zeichen, das als Escape-Zeichen für Anführungszeichen innerhalb eines Werts verwendet wird, in dem bereits ein Escape-Zeichen verwendet wird

Typ: Zeichenfolge

Erforderlich: Nein

QuoteFields

Ein Wert, der angibt, ob alle Ausgabefelder in Anführungszeichen aufgeführt werden sollen

Gültige Werte: `ALWAYS` | `ASNEEDED`

Typ: Zeichenfolge

Erforderlich: Nein

RecordDelimiter

Ein einzelnes Zeichen, das zum Trennen einzelner Datensätze voneinander verwendet wird

Typ: Zeichenfolge

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

Verschlüsselung

Enthält Informationen zur Verschlüsselung, mit der die Auftragsergebnisse in Amazon S3 gespeichert werden.

Inhalt

Verschlüsselung

Der serverseitige Verschlüsselungsalgorithmus, der zum Speichern von Auftragsergebnissen in Amazon S3 verwendet wird. Standardmäßig wird keine Verschlüsselung verwendet.

Typ: Zeichenfolge

Gültige Werte: `aws:kms` | `AES256`

Erforderlich: Nein

KMSContext

Optional. Wenn der Verschlüsselungstyp `aws:kms`, ist, können Sie diesen Wert verwenden, um den Verschlüsselungskontext für die Auftragsergebnisse festzulegen.

Typ: Zeichenfolge

Erforderlich: Nein

KMSKeyId

Die AWS Key Management Service (AWS KMS) Schlüssel-ID, die für die Objektverschlüsselung verwendet werden soll.

Typ: Zeichenfolge

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

GlacierJobDescription

Enthält die Beschreibung eines Amazon Glacier-Jobs (Amazon Glacier).

Inhalt

Action (Aktion)

Der Auftragstyp. Er ist entweder `ArchiveRetrieval`, `InventoryRetrieval` oder `Select`.

Typ: Zeichenfolge

ArchiveId

Die Archiv-ID, die für eine Auswahl oder einen Archivabruf angefordert wurde. Andernfalls ist dieses Feld `null`.

Typ: Zeichenfolge

ArchivSHA256TreeHash

Der SHA256 Struktur-Hash des gesamten Archivs für einen Archivabruf. Für Inventarabrufaufträge ist dieses Feld `null`.

Typ: Zeichenfolge

ArchiveSizeInBytes

Für einen `ArchiveRetrieval`-Auftrag ist dies die Größe in Bytes des zum Herunterladen angeforderten Archivs. Für den `InventoryRetrieval`-Auftrag ist der Wert `null`.

Typ: Zahl

Completed

`true`, wenn der Auftrag abgeschlossen ist; anderenfalls `false`.

Typ: Boolesch

CompletionDate

Das Datum, an dem der Auftrag abgeschlossen wurde

Uhrzeit der Fertigstellung der Auftragsanforderung in UTC (Universal Coordinated Time). Während der Auftrag in Bearbeitung ist, ist der Wert Null.

Typ: Eine Zeichenkettendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

CreationDate

Datum und Uhrzeit des Starts des Auftrags in UTC (Universal Coordinated Time).

Typ: Eine Zeichenkettendarstellung im ISO 8601-Datumsformat, z. B. `2013-03-20T17:03:43.221Z`.

InventoryRetrievalParameters

Die Eingabeparameter für die Inventarabfrage für einzelne Bereiche

Typ: InventoryRetrievalJobInput Objekt

InventorySizeInBytes

Für einen `InventoryRetrieval`-Auftrag ist dies die Größe in Bytes des zum Herunterladen angeforderten Inventars. Für den `ArchiveRetrieval`- oder `Select`-Auftrag ist der Wert `null`.

Typ: Zahl

JobDescription

Die Auftragsbeschreibung, die Sie bei der Initiierung des Auftrags angegeben haben

Typ: Zeichenfolge

JobId

Die ID, die den Job in Amazon Glacier identifiziert.

Typ: Zeichenfolge

JobOutputPath

Enthält den Speicherort für die Auftragsausgabe.

Typ: Zeichenfolge

OutputLocation

Ein Objekt, das Informationen zum Speicherort der Ergebnisse und Fehler des ausgewählten Jobs enthält

Typ: OutputLocation Objekt

RetrievalByteRange

Der abgerufene Bytebereich für Jobs zum Abrufen von Archiven in der Form "*`StartByteValue`*-“*`EndByteValue`*. Wenn beim Archivabruf kein Bereich angegeben wurde, wird das gesamte Archiv abgerufen. Es StartByteValueentspricht 0 und EndByteValueentspricht der Größe des Archivs minus 1. Für Inventarabrufaufträge ist dieses Feld `null`.

Typ: Zeichenfolge

SelectParameters

Ein Objekt mit Informationen zu den Parametern für eine Auswahl

Typ: SelectParameters Objekt

SHA256TreeHash

Der SHA256 Struktur-Hashwert für den angeforderten Bereich eines Archivs. Wenn in der Initiate Job (POST jobs)-Anfrage für ein Archiv ein mit dem Struktur-Hash abgeglichener Bereich angegeben wurde, gibt dieses Feld einen Wert zurück. Weitere Informationen zur Verwendung von mit dem Struktur-Hash abgeglichenen Bereichen beim Abrufen von Archivbereichen finden Sie unter Erhalten von Prüfsummen bei Daten-Download.

Wenn das gesamte Archiv abgerufen wird, ist dieser Wert mit dem `ArchiveSHA256TreeHash`-Wert identisch.

In den folgenden Situationen hat dieses Feld den Wert `null`.

- Aufträge zum Abruf von Archiven, die einen Bereich angeben, der nicht mit dem Struktur-Hash abgeglichen ist.
- Archivaufträge, die einen Bereich festlegen, der das gesamte Archiv einschließt, und deren Auftragsstatus `InProgress` lautet.
- Inventaraufträge.
- Wählen Sie die Aufträge aus.

Typ: Zeichenfolge

SNSTopic

Der Amazon-Ressourcenname (ARN), der das Amazon-SNS-Thema darstellt, an das nach Beendigung oder Fehlschlagen eines Auftrags eine Benachrichtigung gesendet wird, sofern Benachrichtigungen bei der Initiierung des Auftrags (Initiate Job (POST jobs)) konfiguriert wurden.

Typ: Zeichenfolge

StatusCode

Der Code, der den Status des Auftrags angibt

Zulässige Werte: `InProgress` | `Succeeded` | `Failed`

Typ: Zeichenfolge

StatusMessage

Die Nachricht für den Auftragsstatus.

Typ: Zeichenfolge

Stufe

Das Datenzugriffskontingent, das für die Auswahl oder den Archivabruf verwendet wird

Zulässige Werte: `Expedited` | `Standard` | `Bulk`

Typ: Zeichenfolge

VaultARN

Der ARN des Tresors, von dem der Auftrag eine Unterressource ist

Typ: Zeichenfolge

Weitere Infos

- Initiate Job (POST jobs)

Gewährung

Enthält Informationen zu einer Zuwendung.

Inhalt

Grantee

Das Empfänger einer Zuwendung

Typ: Grantee-Objekt

Erforderlich: Nein

Berechtigung

Die Berechtigung, die der Empfänger erhält

Typ: Zeichenfolge

Zulässige Werte: `FULL_CONTROL` | `WRITE` | `WRITE_ACP` | `READ` | `READ_ACP`

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

Grantee

Enthält Informationen zu einem Zuwendungsempfänger.

Inhalt

DisplayName

Der Anzeigename des Empfängers

Typ: Zeichenfolge

Erforderlich: Nein

EmailAddress

Die E-Mail-Adresse des Empfängers

Typ: Zeichenfolge

Erforderlich: Nein

ID (ID)

Die kanonische Benutzer-ID des Empfängers

Typ: Zeichenfolge

Erforderlich: Nein

Typ

Der Typ des Empfängers

Typ: Zeichenfolge

Zulässige Werte: `AmazonCustomerByEmail` | `CanonicalUser` | `Group`

Erforderlich: Nein

URI

Die URI der Empfängergruppe

Typ: Zeichenfolge

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

InputSerialization

Beschreibt, wie das Archiv serialisiert wird.

Inhalt

CSV

Ein Objekt, das die Serialisierung eines CSV-kodierten Objekts beschreibt

Typ: CSVInput Objekt

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

InventoryRetrievalJobInput

Bietet Optionen für die Angabe eines Bereichsinventarabrufs.

Inhalt

EndDate

Das Ende des Datumsbereichs (in UTC-Zeit) für den Abruf eines Tresorbestands, der Archive umfasst, die vor diesem Datum erstellt wurden

Gültige Werte: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat (`YYYY-MM-DDThh:mm:ssTZD`) in Sekunden, zum Beispiel `2013-03-20T17:03:43Z`.

Typ: Zeichenfolge. Gültige Werte: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat (`YYYY-MM-DDThh:mm:ssTZD`) in Sekunden, zum Beispiel `2013-03-20T17:03:43Z`.

Erforderlich: Nein

Format

Das Ausgabeformat für die Tresorbestandsliste, das beim Initiieren des Auftrags zum Abrufen einer Tresorbestandsliste durch die Initiate Job (POST jobs)-Anfrage festgelegt wird.

Gültige Werte: `CSV` | `JSON`

Erforderlich: Nein

Typ: Zeichenfolge

Limit

Die Höchstzahl der Bestandselemente, die bei jeder Anforderung zum Abrufen des Tresorbestands zurückgegeben werden können

Gültige Werte: Eine Ganzzahl gleich oder größer als 1.

Typ: Zeichenfolge

Erforderlich: Nein

Marker

Eine opake Zeichenfolge, die angibt, wo die Paginierung der Ergebnisse des Abrufs des Tresorbestands fortgesetzt werden soll. Sie verwenden diese Markierung in einer neuen `Initiate Job`-Anforderung, um zusätzliche Inventarelemente zu erhalten. Wenn keine weiteren Inventarelemente vorhanden sind, ist dieser Wert null.

Typ: Zeichenfolge

Erforderlich: Nein

StartDate

Der Beginn des Datumsbereichs (in UTC-Zeit) für den Abruf eines Tresorbestands, der Archive umfasst, die an oder nach diesem Datum erstellt wurden

Gültige Werte: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat (`YYYY-MM-DDThh:mm:ssTZD`) in Sekunden, zum Beispiel `2013-03-20T17:03:43Z`.

Typ: Zeichenfolge. Gültige Werte: Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat (`YYYY-MM-DDThh:mm:ssTZD`) in Sekunden, zum Beispiel `2013-03-20T17:03:43Z`.

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

jobParameters

Bietet Optionen zum Definieren von Jobs

Inhalt

ArchiveId

Die ID des gewünschten Archivs. Dieses Feld ist erforderlich, wenn das Feld `Type` auf `select` oder `archive-retrieval` eingestellt ist. Wenn Sie dieses Feld bei einer Auftragsanforderung für eine Inventarabfrage festlegen, tritt ein Fehler auf.

Gültige Werte: Muss eine gültige Archiv-ID sein, die Sie aus einer früheren Anfrage an Amazon Glacier (Amazon Glacier) erhalten haben.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `Type` auf `select` oder `archive-retrieval` eingestellt ist.

Beschreibung

Geben Sie die optionale Beschreibung für den Auftrag ein.

Gültige Werte: Die Beschreibung darf höchstens 1 024 Bytes umfassen. Zulässig sind 7-Bit-ASCII-Zeichen ohne Steuercodes, also die ASCII-Dezimalwerte 32 bis 126 bzw. die ASCII-Hexadezimalwerte 0x20 bis 0x7E.

Typ: Zeichenfolge

Erforderlich: Nein

Format

(Optional) Das Ausgabeformat, das beim Initiieren des Auftrags zum Abrufen einer Tresorbestandsliste festgelegt wird. Wenn Sie einen Inventarauftrag initiieren und kein `Format`-Feld angeben, wird JSON als Standardformat verwendet.

Gültige Werte: `CSV` | `JSON`

Typ: Zeichenfolge

Erforderlich: Nein

InventoryRetrievalParameters

Die Eingabeparameter für die Inventarabfrage für einzelne Bereiche

Typ: InventoryRetrievalJobInput Objekt

Erforderlich: Nein

OutputLocation

Ein Objekt, das Informationen zum Speicherort der Ergebnisse des ausgewählten Jobs enthält

Typ: OutputLocation Objekt

Erforderlich: Ja, für `select`-Aufträge

RetrievalByteRange

Der Bytebereich, der für eine abgerufen werden soll`archive-retrieval`, in der Form "*`StartByteValue`*-*`EndByteValue`*“. Wenn kein Bereich angegeben ist, wird das gesamte Archiv abgerufen. Wenn ein Feldwert angegeben wird, muss der Bereich in Megabytewerten (1024*1024) angegeben werden. Megabyte-ausgerichtet bedeutet, dass der Wert durch 1 MB teilbar sein StartByteValuemuss und EndByteValueplus 1 durch 1 MB teilbar sein muss oder das Ende des Archivs sein muss, das als Wert für die Archiv-Bytegröße minus 1 angegeben ist. Wenn nicht auf Megabyte RetrievalByteRangeausgerichtet ist, gibt diese Operation eine Antwort zurück. `400`

Wenn Sie dieses Feld bei einer `inventory-retrieval`- oder `select`-Auftragsanforderung festlegen, tritt ein Fehler auf.

Typ: Zeichenfolge

Erforderlich: Nein

SelectParameters

Ein Objekt mit Informationen zu den Parametern für eine Auswahl

Typ: SelectParameters Objekt

Erforderlich: Nein

SNSTopic

Der Amazon-Ressourcenname (ARN) des Amazon SNS-Themas, an das Amazon Glacier eine Benachrichtigung sendet, wenn der Job abgeschlossen ist und die Ausgabe für Sie zum Herunterladen bereit ist. Das angegebene Thema veröffentlicht die Benachrichtigung an die Abonnenten.

Das SNS-Thema muss vorhanden sein. Wenn nicht, erstellt Amazon Glacier es nicht für Sie. Das SNS-Thema muss außerdem über eine Richtlinie verfügen, die dem Konto, über das der Auftrag erstellt wurde, erlaubt, Nachrichten im Thema zu veröffentlichen. Informationen zu SNS-Themennamen finden Sie CreateTopicin der Amazon Simple Notification Service API-Referenz.

Typ: Zeichenfolge

Erforderlich: Nein

Stufe

Das Kontingent, das für eine Auswahl oder einen Archivabruf verwendet wird. Der Standardwert ist `Standard`.

Zulässige Werte: `Expedited` | `Standard` | `Bulk`

Typ: Zeichenfolge

Erforderlich: Nein

Typ

Der Auftragstyp. Sie können eine -Aufgabe initiieren, um eine Auswahlabfrage an ein Archiv zu senden, ein Archiv abzurufen oder ein Inventar des Tresors zu ermitteln.

Zulässige Werte: `select` | `archive-retrieval` | `inventory-retrieval`

Typ: Zeichenfolge

Erforderlich: Ja

Weitere Infos

- Initiate Job (POST jobs)

OutputLocation

Enthält Informationen zum Speicherort, wo die Ergebnisse und Fehler des Auftrags gespeichert werden.

Inhalt

S3

Ein Objekt, das einen Amazon-S3-Speicherort beschreibt, an dem die Ergebnisse der Wiederherstellungsanforderung gespeichert werden

Typ: S3Location

Erforderlich: Ja

Weitere Infos

- Initiate Job (POST jobs)

OutputSerialization

Beschreibt, wie die Ausgabe serialisiert wird.

Inhalt

CSV

Ein Objekt, das die Serialisierung der CSV-kodierten Abfrageergebnisse beschreibt

Typ: CSVOutput Objekt

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

S3Location

Enthält Informationen zum Speicherort in Amazon S3, an dem die Ergebnisse des Auftrags gespeichert werden.

Inhalt

AccessControlList

Eine Liste mit Berechtigungen, über die der Zugriff auf die gespeicherten Ergebnisse gesteuert wird

Typ: Array von Gewährung-Objekten

Erforderlich: Nein

BucketName

Der Name des Amazon-S3-Buckets, in dem die Auftragsergebnisse gespeichert werden. Der Bucket muss sich in derselben AWS Region befinden wie der Tresor, der das Eingabearchivobjekt enthält.

Typ: Zeichenfolge

Erforderlich: Ja

CannedACL

Die vordefinierte Zugriffskontrollliste (ACL), die auf die Auftragsergebnisse angewendet wird

Typ: Zeichenfolge

Zulässige Werte: `private` | `public-read` | `public-read-write` | `aws-exec-read` | `authenticated-read` | `bucket-owner-read` | `bucket-owner-full-control`

Erforderlich: Nein

Verschlüsselung

Ein Objekt, das Informationen zur Verschlüsselung enthält, mit der die Auftragsergebnisse in Amazon S3 gespeichert werden

Typ: Verschlüsselung Objekt

Erforderlich: Nein

Präfix

Das Präfix, das den Ergebnissen für diese Anforderung vorangestellt wird. Die maximale Länge für das Präfix beträgt 512 Byte.

Typ: Zeichenfolge

Erforderlich: Ja

StorageClass

Die Speicherklasse, die zum Speichern der Auftragsergebnisse verwendet wird

Typ: Zeichenfolge

Zulässige Werte: `STANDARD` | `REDUCED_REDUNDANCY` | `STANDARD_IA`

Erforderlich: Nein

Tagging

Die Tag-Menge, die auf die Auftragsergebnisse angewendet wird

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

UserMetadata

Eine Zuordnung der Metadaten, die mit den Auftragsergebnissen in Amazon S3 gespeichert werden.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Weitere Infos

- Initiate Job (POST jobs)

SelectParameters

Dieser Artikel enthält Informationen zu den Parametern für die Auswahl.

Inhalt

Expression

Der Ausdruck, der zur Auswahl des Objekts verwendet wird. Der Ausdruck darf das Kontingent von 128.000 Zeichen nicht überschreiten.

Typ: Zeichenfolge

Erforderlich: Ja

ExpressionType

Der Typ des bereitgestellten Ausdrucks, beispielsweise SQL

Gültige Werte: SQL

Typ: Zeichenfolge

Erforderlich: Ja

InputSerialization

Beschreibt das Serialisierungsformat des Objekts in der Auswahl.

Typ: InputSerialization Objekt

Erforderlich: Nein

OutputSerialization

Beschreibt, wie die Ergebnisse des Auswahlauftrags serialisiert werden.

Erforderlich: Nein

Typ: OutputSerialization Objekt

Weitere Infos

- Initiate Job (POST jobs)

Datenabrufoperationen

Im Folgenden sind die in Amazon Glacier verfügbaren Vorgänge im Zusammenhang mit dem Datenabruf aufgeführt.

Themen

- Get Data Retrieval Policy (GET policy)
- List Provisioned Capacity (GET provisioned-capacity)
- Purchase Provisioned Capacity (POST provisioned-capacity)
- Set Data Retrieval Policy (PUT policy)

Get Data Retrieval Policy (GET policy)

Beschreibung

Dieser Vorgang gibt die aktuelle Datenabrufrichtlinie für die in der AWS-Konto Anforderung angegebene AWS Region und zurück. GET Weitere Informationen zu Datenabrufrichtlinien finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf.

Anforderungen

Senden Sie eine HTTP-GET-Anforderung an den URI der Richtlinie für den Datenabruf, damit die aktuelle Datenabruf-Richtlinie zurückgegeben wird. Orientieren Sie sich dabei an dem folgenden Syntaxbeispiel.

Syntax

```
GET /AccountId/policies/data-retrieval HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
```

```
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Syntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length
{
  "Policy":
    {
```

```
      "Rules":[
         {
            "BytesPerHour": Number,
            "Strategy": String
         }
       ]
    }
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

BytesPerHour

Die Höchstanzahl an Bytes, die in einer Stunde abgerufen werden können.

Dieses Feld ist nur vorhanden, wenn der Wert im Strategy-Feld `BytesPerHour` ist.

Typ: Zahl

Regeln

Die Richtlinienregel. Obwohl es sich um einen Listentyp handelt, gibt es derzeit nur eine Regel, die ein Strategiefeld und optional ein BytesPerHour Feld enthält.

Typ: Array

Strategie

Der Typ der Richtlinie für den Datenabruf.

Typ: Zeichenfolge

Gültige Werte: `BytesPerHour`|`FreeTier`|None. `BytesPerHour` entspricht der Auswahl von Max Retrieval Rate (Max. Abrufrate) in der Konsole. `FreeTier` entspricht der Auswahl von Free Tier Only (Nur kostenloses Kontingent) in der Konsole. None entspricht der Auswahl von

No Retrieval Policy (Keine Abrufrichtlinie) in der Konsole. Weitere Informationen zur Auswahl von Datenabrufrichtlinien in der Konsole finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Das folgende Beispiel demonstriert, wie eine Richtlinie für den Datenabruf abgerufen wird.

Beispielanfrage

In diesem Beispiel wird eine GET-Anforderung an den URI für den Speicherort der Richtlinie gesendet.

```
GET /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Falls die Anforderung erfolgreich beantwortet wird, wird die Richtlinie für den Datenabruf im Antworttext im JSON-Format angezeigt.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 85

{
  "Policy":
    {
      "Rules":[
```

```
          {
            "BytesPerHour":10737418240,
            "Strategy":"BytesPerHour"
           }
        ]
    }
}
```

Zugehörige Abschnitte

- Set Data Retrieval Policy (PUT policy)
- Initiate Job (POST jobs)

List Provisioned Capacity (GET provisioned-capacity)

Dieser Vorgang führt die bereitgestellten Kapazitätseinheiten für das angegebene AWS-Konto auf. Weitere Informationen über bereitgestellte Kapazität finden Sie unter Archiv-Abrufoptionen.

Eine bereitgestellte Kapazitätseinheit ist ab dem Datum und der Uhrzeit des Kaufs (dem Startdatum), einen Monat lang gültig. Die Einheit läuft am Ablaufdatum ab, das auf die nächste Sekunde gerundet genau einen Monat nach dem Startdatum liegt.

Wenn das Startdatum der 31. Tag eines Monats ist, dann ist der letzte Tag des nächsten Monats das Ablaufdatum. Beispiel: Ist das Startdatum der 31. August, dann ist der 30. September das Ablaufdatum. Ist das Startdatum der 31. Januar, dann ist der 28. Februar das Ablaufdatum. Diese Funktionalität wird im Beispielantwort veranschaulicht.

Anforderungssyntax

Wenn Sie die bereitgestellte Abrufkapazität für ein Konto aufführen möchten, senden Sie eine HTTP GET-Anfrage an den URI der bereitgestellten Kapazität, wie im folgenden Syntax-Beispiel verdeutlicht wird.

```
GET /AccountId/provisioned-capacity HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
```

```
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '`-`' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Wenn die Operation erfolgreich ist, sendet der Service eine HTTP-`200 OK`-Antwort zurück.

Antwortsyntax

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length
{
   "ProvisionedCapacityList":
```

```
      {
         "CapacityId" : "string",
         "StartDate" : "string"
         "ExpirationDate" : "string"
      }
}
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Der Antworttext enthält die folgenden JSON-Felder.

CapacityId

Die ID, mit der die bereitgestellte Kapazitätseinheit identifiziert wird.

Typ: Zeichenfolge.

StartDate

Das Datum (Coordinated Universal Time – UTC), an dem die bereitgestellte Kapazitätseinheit gekauft wurde.

Typ: Zeichenfolge. Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, zum Beispiel `2013-03-20T17:03:43.221Z`.

ExpirationDate

Das Datum (Coordinated Universal Time – UTC), an dem die bereitgestellte Kapazitätseinheit abläuft wurde.

Typ: Zeichenfolge. Eine Zeichenfolgendarstellung im ISO 8601-Datumsformat, zum Beispiel `2013-03-20T17:03:43.221Z`.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Es folgt ein Beispiel für das Aufführen der bereitgestellten Kapazitätseinheiten für ein Konto.

Beispielanforderung

In diesem Beispiel wird eine GET-Anfrage gesendet, um eine Liste der bereitgestellten Kapazitätseinheiten für das angegebene Konto abzurufen.

```
GET /123456789012/priority-capacity HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier (Amazon Glacier) a `HTTP 200 OK` mit einer Liste der bereitgestellten Kapazitätseinheiten für das Konto zurück, wie im folgenden Beispiel gezeigt.

Die zuerst angeführte bereitgestellte Kapazitätseinheit ist ein Beispiel einer Einheit mit dem 31. Januar 2017 als Startdatum und dem 28. Februar 2017 als Ablaufdatum. Wie bereits erwähnt, ist das Ablaufdatum bei einem Startdatum am 31. Tag eines Monats der letzte Tag des nächsten Monats.

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: length

{
   "ProvisionedCapacityList",
      {
         "CapacityId": "zSaq7NzHFQDANTfQkDen4V7z",
         "StartDate": "2017-01-31T14:26:33.031Z",
         "ExpirationDate": "2017-02-28T14:26:33.000Z",
      },
      {
```

```
         "CapacityId": "yXaq7NzHFQNADTfQkDen4V7z",
         "StartDate": "2016-12-13T20:11:51.095Z"",
         "ExpirationDate": "2017-01-13T20:11:51.000Z" ",
      },
      ...
}
```

Zugehörige Abschnitte

- Purchase Provisioned Capacity (POST provisioned-capacity)

Purchase Provisioned Capacity (POST provisioned-capacity)

Dieser Vorgang kauft eine bereitgestellte Kapazitätseinheit für ein AWS-Konto.

Eine bereitgestellte Kapazitätseinheit ist ab dem Datum und der Uhrzeit des Kaufs (dem Startdatum), einen Monat lang gültig. Die Einheit läuft am Ablaufdatum ab, das auf die nächste Sekunde gerundet genau einen Monat nach dem Startdatum liegt.

Wenn das Startdatum der 31. Tag eines Monats ist, dann ist der letzte Tag des nächsten Monats das Ablaufdatum. Beispiel: Ist das Startdatum der 31. August, dann ist der 30. September das Ablaufdatum. Ist das Startdatum der 31. Januar, dann ist der 28. Februar das Ablaufdatum.

Die bereitgestellte Kapazität hilft sicherzustellen, dass für Expedited-Abrufe Abrufkapazität verfügbar ist, wenn Sie sie benötigen. Jede Kapazitätseinheit stellt sicher, dass mindestens drei beschleunigte Abrufe alle fünf Minuten durchgeführt werden können, und bietet einen Abrufdurchsatz von bis zu 150 MB/s %. Weitere Informationen über bereitgestellte Kapazität finden Sie unter Archiv-Abrufoptionen.

> **Note**
>
> Es gibt eine Obergrenze von zwei bereitgestellten Kapazitätseinheiten pro Einheit. AWS-Konto

Anforderungen

Um eine bereitgestellte Kapazitätseinheit für einen zu erwerben, AWS-Konto senden Sie eine `POST` HTTP-Anfrage an die URI für bereitgestellte Kapazität.

Syntax

```
POST /AccountId/provisioned-capacity HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> Der `AccountId` Wert ist die ID. AWS-Konto Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Dieser Vorgang besitzt keinen Anforderungstext.

Antworten

Wenn die Anfrage erfolgreich ist, gibt der Service die HTTP-Antwort `201 Created` zurück.

Syntax

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
```

```
Date: Date
x-amz-capacity-id: CapacityId
```

Antwort-Header

Eine erfolgreiche Antwort umfasst zusätzlich zu den Antwort-Headern, die allen Vorgängen gemeinsam sind, die folgenden Antwort-Header. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Name	Description
`x-amz-capacity-id`	Die ID, mit der die bereitgestellte Kapazitätseinheit identifiziert wird. Typ: Zeichenfolge

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Dieser Vorgang umfasst zusätzlich zu den möglichen Fehlern, die allen Amazon Glacier-Vorgängen gemeinsam sind, die folgenden Fehler oder Fehler. Informationen zu Amazon Glacier-Fehlern und eine Liste der Fehlercodes finden Sie unterFehlermeldungen.

Code	Description	HTTP-Statuscode	Typ
`LimitExceededException`	Wird zurückgegeben, wenn die gestellte Anfrage das Limit für Einheiten bereitgestellter Kapazität des Kontos überschreiten würde.	`400 Bad Request`	Client

Beispiele

Das Folgende ist ein Beispiel für den Kauf bereitgestellter Kapazität für ein Konto.

Beispielanforderung

Im folgenden Beispiel wir eine HTTP-POST-Anfrage gesendet, um eine Einheit bereitgestellter Kapazität zu kaufen.

```
POST /123456789012/provisioned-capacity HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01
```

Beispielantwort

Wenn die Anfrage erfolgreich war, gibt Amazon Glacier (Amazon Glacier) eine `HTTP 201 Created` Antwort zurück, wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
x-amz-capacity-id: zSaq7NzHFQDANTfQkDen4V7z
```

Zugehörige Abschnitte

- List Provisioned Capacity (GET provisioned-capacity)

Set Data Retrieval Policy (PUT policy)

Beschreibung

Durch diesen Vorgang wird eine Datenabrufrichtlinie in der in der Anforderung angegebenen AWS Region festgelegt und anschließend umgesetzt. PUT Sie können eine Richtlinie pro AWS Region für eine festlegen. AWS-Konto Die Richtlinie wird bei einer erfolgreichen PUT-Operation innerhalb weniger Minuten umgesetzt.

Die Operation zur Einrichtung der Richtlinie wirkt sich nicht auf Abrufaufträge aus, die in Bearbeitung waren, bevor die Richtlinie umgesetzt wurde. Weitere Informationen zu Datenabrufrichtlinien finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf.

Anforderungen

Syntax

Senden Sie eine HTTP-PUT-Anfrage an den URI der Datenabrufrichtlinie, um eine Datenabrufrichtlinie einzurichten. Orientieren Sie sich dabei an dem folgenden Syntaxbeispiel.

```
PUT /AccountId/policies/data-retrieval HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
  "Policy":
    {
      "Rules":[
         {
             "Strategy": String,
             "BytesPerHour": Number
         }
       ]
    }
}
```

> **Note**
>
> Der `AccountId` Wert ist die AWS-Konto ID. Dieser Wert muss mit der AWS-Konto ID übereinstimmen, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Sie können entweder eine AWS-Konto ID oder optional ein einzelnes '-' (Bindestrich) angeben. In diesem Fall verwendet Amazon Glacier die AWS-Konto ID, die den Anmeldeinformationen zugeordnet ist, die zum Signieren der Anfrage verwendet wurden. Wenn Sie Ihre Konto-ID angeben, müssen Sie sie ohne Bindestriche („-“) eingeben.

Anforderungsparameter

Dieser Vorgang verwendet keine Anforderungsparameter.

Anforderungsheader

Für diesen Vorgang werden ausschließlich Anforderungs-Header verwendet, die für alle Vorgänge gelten. Weitere Informationen über allgemeine Anforderungs-Header finden Sie unter Häufig verwendete Anforderungsheader.

Anforderungstext

Der Anforderungstext enthält die folgenden JSON-Felder.

BytesPerHour

Die Höchstanzahl an Bytes, die in einer Stunde abgerufen werden können.

Dieses Feld ist nur dann ein Pflichtfeld, wenn der Wert im Strategy-Feld `BytesPerHour` ist. Ihre PUT-Operation wird abgelehnt, wenn das Strategy-Feld nicht auf `BytesPerHour` festgelegt ist und Sie dieses Feld festlegen.

Typ: Zahl

Erforderlich: Ja, wenn das Strategie-Feld auf `BytesPerHour` festgelegt ist. Ansonsten nein.

Gültige Werte: Mindestganzzahlwert 1. Höchstganzzahlwert 2^63 - 1 inklusive.

Regeln

Die Richtlinienregel. Obwohl es sich um einen Listentyp handelt, darf es derzeit nur eine Regel geben, die ein Strategiefeld und optional ein BytesPerHour Feld enthält.

Typ: Array

Erforderlich: Ja

Strategie

Der Typ der einzurichtenden Datenabrufrichtlinie.

Typ: Zeichenfolge

Erforderlich: Ja

Gültige Werte: `BytesPerHour`|`FreeTier`|None. `BytesPerHour` entspricht der Auswahl von Max Retrieval Rate (Max. Abrufrate) in der Konsole. `FreeTier` entspricht der Auswahl von Free Tier Only (Nur kostenloses Kontingent) in der Konsole. None entspricht der Auswahl von

No Retrieval Policy (Keine Abrufrichtlinie) in der Konsole. Weitere Informationen zur Auswahl von Datenabrufrichtlinien in der Konsole finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf.

Antworten

Syntax

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

Antwort-Header

Für diesen Vorgang werden ausschließlich Antwort-Header verwendet, die für die meisten Antworten gelten. Weitere Informationen über allgemeine Antwort-Header finden Sie unter Allgemeine Antwort-Header.

Antworttext

Dieser Vorgang gibt keinen Antworttext zurück.

Fehler

Informationen zu Amazon Glacier-Ausnahmen und Fehlermeldungen finden Sie unterFehlermeldungen.

Beispiele

Beispielanfrage

Im folgenden Beispiel wird eine HTTP-PUT-Anfrage gesendet, mit dem Strategy-Feld festgelegt auf `BytesPerHour`.

```
PUT /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

```
{
  "Policy":
    {
      "Rules":[
         {
             "Strategy":"BytesPerHour",
             "BytesPerHour":10737418240
          }
       ]
    }
}
```

Im folgenden Beispiel wird eine HTTP-PUT-Anfrage gesendet, mit dem Strategy-Feld festgelegt auf `FreeTier`.

```
PUT /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Policy":
    {
      "Rules":[
         {
             "Strategy":"FreeTier"
          }
       ]
    }
}
```

Im folgenden Beispiel wird eine HTTP-PUT-Anfrage gesendet, mit dem Strategy-Feld festgelegt auf `None`.

```
PUT /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Policy":
    {
      "Rules":[
         {
             "Strategy":"None"
          }
       ]
    }
}
```

Beispielantwort

Wenn die Anfrage erfolgreich war, legt Amazon Glacier (Amazon Glacier) die Richtlinie fest und gibt a zurück, `HTTP 204 No Content` wie im folgenden Beispiel gezeigt.

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

Zugehörige Abschnitte

- Get Data Retrieval Policy (GET policy)
- Initiate Job (POST jobs)

Dokumentverlauf

- Aktuelle Produktversion: 2012-06-01

In der folgenden Tabelle werden die wichtigen Änderungen in den einzelnen Versionen des Amazon Glacier Developer Guide ab dem 5. Juli 2018 beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Verbesserte Startzeiten für Standard-Wiederherstellungsanfragen, die über S3 Batch Operations gestellt wurden	Standardabrufe für Wiederherstellungsanfragen, die über S3 Batch Operations gestellt wurden, können jetzt innerhalb von Minuten beginnen. Weitere Informationen finden Sie unter Optionen für den Archivabruf.	9. August 2023
Amazon S3 unterstützt höhere Geschwindigkeiten für Wiederherstellungsanforderungen für S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive	Amazon S3 unterstützt Wiederherstellungsanforderungen mit einer Geschwindigkeit von bis zu 1 000 Transaktionen pro Sekunde, je AWS-Konto für die Speicherklassen S3 Glacier Flexible Retrieval und S3 Glacier Deep Archive.	15. November 2022
Amazon-Glacier-Namensänderung	Amazon Glacier ist jetzt Amazon Glacier, um die Integration von Glacier mit Amazon S3 besser widerzuspiegeln.	20. November 2018
Updates jetzt über RSS verfügbar	Sie können jetzt einen RSS-Feed abonnieren, um	5. Juli 2018

Benachrichtigungen über Aktualisierungen des Amazon Glacier Developer Guide Guide zu erhalten.

Frühere Aktualisierungen

In der folgenden Tabelle werden die wichtigen Änderungen beschrieben, die in den einzelnen Versionen des Amazon Glacier Developer Guide vor dem 5. Juli 2018 vorgenommen wurden.

Änderungen	Beschreibung	Veröffentlichungsdatum
Expedited- und Bulk-Datenabrufvorgänge	Amazon Glacier unterstützt jetzt zusätzlich zu Standardabrufen auch beschleunigte Datenabrufe und Massenabrufe. Weitere Informationen finden Sie unter Archiv-Abrufoptionen.	21. November 2016
Vault Lock	Amazon Glacier unterstützt jetzt Vault Lock, sodass Sie mithilfe einer Vault Lock-Richtlinie auf einfache Weise Compliance-Kontrollen für einzelne Amazon Glacier-Tresore einrichten und durchsetzen können. Weitere Informationen erhalten Sie unter Amazon Glacier Tresorschloss und Tresorverriegelungs-Richtlinien.	8. Juli 2015
Markieren von Tresoren	Mit Amazon Glacier können Sie jetzt Ihre Amazon Glacier-Tresore taggen, um das Ressourcen- und Kostenmanagement zu vereinfachen. Tags sind Beschriftungen, die Sie definieren und mit Ihren Tresoren verknüpfen können. Durch die Verwendung von Tags können Vorgänge wie AWS Kostenberichte um Filterfunktionen erweitert werden. Weitere Informationen erhalten Sie unter Taggen von Amazon Glacier-Ressourcen und Kennzeichnen Ihrer Amazon Glacier-Tresore.	22. Juni 2015

Änderungen	Beschreibung	Veröffentlichungsdatum
Richtlinien für den Tresorzugriff	Amazon Glacier unterstützt jetzt die Verwaltung des Zugriffs auf Ihre individuellen Amazon Glacier-Tresore mithilfe von Tresorzugriffsrichtlinien. Sie können jetzt eine Zugriffsrichtlinie direkt auf einem Tresor definieren, um Benutzern und Geschäftsgruppen in Ihrem Unternehmen sowie externen Geschäftspartnern einfacher Zugriff auf den Tresor zu gewähren. Weitere Informationen finden Sie unter Richtlinien für den Tresorzugriff.	27. April 2015
Datenabrufrichtlinien und Prüfprotokollierung	Amazon Glacier unterstützt jetzt Richtlinien für den Datenabruf und die Auditprotokollierung. Mit Datenabrufrichtlinien können Sie einfacher Datenabruflimits einrichten und das Kostenmanagement für den Datenabruf vereinfachen. Sie können Ihre eigenen Grenzwerte für den Datenabruf mit wenigen Klicks in der AWS-Managementkonsole oder mithilfe der Amazon Glacier-API definieren. Weitere Informationen finden Sie unter Amazon Glacier-Richtlinien für den Datenabruf. Darüber hinaus unterstützt Amazon Glacier jetzt die Audit-Protokollierung mit AWS CloudTrail, die Amazon Glacier-API-Aufrufe für Ihr Konto aufzeichnet und die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket übermittelt. Weitere Informationen finden Sie unter Protokollieren von Amazon Glacier-API-Aufrufen mit AWS CloudTrail.	11. Dezember 2014
Updates für Java-Beispiele	Die Java-Codebeispiele in diesem Handbuch, in denen AWS SDK für Java verwendet wird, wurden aktualisiert.	27. Juni 2014

Änderungen	Beschreibung	Veröffentlichungsdatum
Begrenzen des Abrufs von Tresorbeständen	Sie können jetzt die Anzahl der abgerufenen Elemente des Tresorbestands begrenzen, indem Sie nach dem Archiverstellungsdatum filtern oder ein Limit festlegen . Weitere Informationen zur Begrenzung des Inventarabrufs finden Sie unter Inventarabfrage für einzelne Bereiche im Thema Initiate Job (POST jobs).	31. Dezember 2013
Veraltete Version entfernt URLs	Die Seite URLs , die auf die alte Seite mit den Sicherheitsanmeldedaten verweist, wurde aus den Codebeispielen entfernt.	26. Juli 2013
Unterstützung für Bereichsabrufe	Amazon Glacier unterstützt jetzt das Abrufen bestimmter Bereiche Ihrer Archive. Sie können einen Job starten, in dem Amazon Glacier aufgefordert wird, ein ganzes Archiv oder einen Teil des Archivs für den späteren Download vorzubereiten. Wenn ein Archiv sehr groß ist, können unter Umständen Kosten sparen, indem Sie zur Vorbereitung des Archivs mehrere Aufträge in Folge vorbereiten. Weitere Informationen finden Sie unter Ein Archiv in Amazon Glacier herunterladen.	13. November 2012
Neues Handbuch	Dies ist die erste Version des Amazon Glacier Developer Guide.	20. August 2012

AWS Glossar

Die neueste AWS Terminologie finden Sie im AWS Glossar in der AWS-Glossar Referenz.