Cloud Object Storage

開発者ガイド

製品ドキュメント

Copyright Notice

©2013-2024 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice

Tencent Cloud

All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

カタログ：

開発者ガイド
- リクエストの作成
  - リクエスト作成の概要
- バケット
  - バケットの概要
  - バケットの作成
  - バケットの削除
- オブジェクト
  - オブジェクトの概要
  - ストレージタイプ
    - ストレージタイプの概要
    - ディープアーカイブストレージ概要
    - INTELLIGENT_TIERINGストレージ概要
  - オブジェクトのアップロード
    - シンプルアップロード
    - マルチパートアップロード
    - 署名付きURLによるアップロード権限承認
  - オブジェクトのダウンロード
    - オブジェクトのシンプルダウンロード
    - 署名付きURLによるダウンロード権限承認
  - オブジェクトキーのリストアップ
  - オブジェクトのコピー
    - シンプルコピー
    - マルチパートコピー
  - オブジェクトを削除する
    - 単一オブジェクトの削除
    - 複数のオブジェクトの削除
- データ管理
  - ライフサイクル管理
    - ライフサイクルの概要
    - ライフサイクル設定の要素
    - ライフサイクルの設定
  - 静的ウェブサイトのホスティング
  - リスト機能の概要
  - バケットタグの概要

- オブジェクトタグの概要
- イベント通知
- データ検索
  - Selectの概要
  - Selectコマンド
  - SQL関数
  - 予約フィールド
  - データタイプ
  - 演算子
- ログ管理
  - ログ管理の概要
  - ログ管理の制限
  - COSを使用したクラウド製品ログのストレージ

データ災害復帰
- バージョン管理
  - バージョン管理の概要
  - タグの削除
  - バージョン管理の使用
    - バージョン管理の設定
- バケットコピー
  - バケットコピーの概要
  - コピーアクションの説明
  - バケットコピーの設定
- マルチAZ特性の概要

Data Security
- サーバー側の暗号化の概要
- バケット暗号化の概要

クラウドアクセスマネジメント
- アクセス権限設定の説明
  - アクセス管理の概要
    - アクセス制御の基本概念
    - COSの権限承認とID認証のフロー
    - 最小権限の原則説明
    - アクセスポリシーの評価フロー
  - 権限制御方法の紹介
    - バケットポリシー
    - ユーザーポリシー
    - ACL

      - タグに基づくプロジェクトリソースの管理
      - 権限承認方式の選択方法
    - アクセスポリシー言語
      - アクセスポリシーの言語概要
      - 発効条件
    - リクエスト方法の紹介
      - パーマネントキーを使用したCOSアクセス
      - 一時キーを使用したCOSアクセス
      - 署名付きURLを使用したCOSアクセス
      - 匿名でのCOSアクセス
  - CDNアクセラレーションを使用したアクセス
    - CDNアクセラレーションの概要
    - CDNアクセラレーションの設定
  - 単一リンクの速度制限
- バッチ処理
  - バッチ処理タスクの管理
  - バッチ処理の概要
  - バッチ処理操作
    - オブジェクトの一括コピー
    - アーカイブオブジェクトの一括復元
- グローバルアクセラレーション
  - グローバルアクセラレーションの概要
  - プライベートネットワークグローバルアクセラレーション
- 監視とアラーム

開発者ガイド

リクエストの作成

リクエスト作成の概要

最終更新日：：2024-06-26 10:57:13

基本概念

Tencent Cloud COSはHTTP/HTTPSプロトコルを使用してアクセスするWebストレージサービスです。COSにはREST APIまたはCOS SDKを使用してアクセスできます。

COSへのアクセスリクエストを発行する際は、COSによる承認および認証を経てからでなければリソースの操作を行うことはできません。 このためCOSへのアクセスリクエストは、IDが識別可能かどうかによって、匿名リクエストと署名リクエストの2種類に分けられます。

匿名リクエスト：リクエストにAuthorizationまたは関連パラメータが含まれないか、または関連の文字からユーザーのID特性を識別できない場合、リクエストは匿名リクエストとみなされて認証を受けます。

署名リクエスト：署名付きのリクエストはHTTPヘッダーまたはリクエストパケットにAuthorizationフィールドが含まれています。このフィールドの内容はTencent Cloudのセキュリティ証明書のSecretID、SecretKeyとリクエストを結び付けるいくつかの特性値であり、暗号化アルゴリズムによって生成されます。

COS SDKを使用してアクセスする場合は、セキュリティ証明書を設定するだけでリクエストを発行できます。

REST APIを使用してアクセスする場合は、リクエスト署名ドキュメントを参照して、ご自身でリクエスト署名を計算する必要があります。

セキュリティ証明書の取得

Tencent Cloud CAM（Cloud Access Management）はCOS向けにアカウントとセキュリティ証明書に関連する機能およびサービスを提供しています。これは主に、お客様がTencent Cloudアカウント下のリソースへのアクセス権限を安全に管理するために役立てられます。ユーザーはCAMによって、ユーザー（グループ）を作成、管理、および破棄できるほか、ID管理とポリシー管理を使用して、他のユーザーがTencent Cloudのリソースを使用する権限を制御できます。

ルートアカウントのセキュリティ証明書

ルートアカウントにログイン後、CAMのTencent Cloud APIキーページによって、ルートアカウントのセキュリティ証明書のSecretIDおよびSecretKey キーを取得し管理することができます。以下はキーの例です。

36文字のアクセスキーID（SecretID）：AKIDHZRLB9Ibhdp7Y7gyQq6BOk1997xxxxxx

32文字のアクセスキーKey（SecretKey）：LYaWIuQmCSZ5ZMniUM6hiaLxHnxxxxxx

アクセスキーは一意のアカウントを識別するために用いられます。キー署名を使用してリクエストを送信すると、Tencent Cloudはリクエスト発行者のIDを識別し、承認してから、プリンシパル、リソース、アクション、条件などの認証を行い、この操作の実行を許可するかどうかを判断します。

注意：

ルートアカウントキーはそのアカウント下のすべてのリソースの全操作権限を有しています。キーの漏洩はクラウド上の資産損失につながるおそれがあるため、サブアカウントを作成して適正な権限を割り当て、サブアカウントのキーを使用してリクエストを作成し、リソースへのアクセスと管理を行うことを強く推奨します。

サブアカウントのセキュリティ証明書

複数の次元でアカウント下のユーザーおよびクラウドリソースを管理したい場合は、ルートアカウント下に複数のサブアカウントを作成し、リソース権限に対する機能を複数の要員にそれぞれ管理させることができます。サブアカウント作成に関する説明については、CAMのユーザー管理の関連ドキュメントをご参照ください。

サブアカウントを使用してAPIリクエストを発行する前に、サブアカウントのセキュリティ証明書を作成する必要があります。そうすることでサブアカウントも独自のキーペアを持ち、ID識別に役立てることができます。サブアカウントごとにユーザーポリシーを作成し、リソースに対するそれぞれのアクセス権限を制御することができます。またユーザーグループを作成し、ユーザーグループに統一のアクセスポリシーを追加することで、要員のグループ化およびリソースの統一管理に役立てることもできます。

注意：

サブアカウントは対応する権限を割り当てられた場合、リソースの作成または変更が可能ですが、この時点でリソースは引き続きルートアカウントに属するため、このリソースによって発生する料金はルートアカウントが支払う必要があります。

一時的なセキュリティ証明書

ルートアカウントまたはサブアカウントのセキュリティ証明書を使用したリソースへのアクセスのほかに、Tencent Cloudは、ロールを作成し、一時的なセキュリティ証明書を使用してTencent Cloudのリソースを管理することもサポートしています。ロールの基本概念および使用方法に関しては、CAMのロール管理のドキュメントをご参照ください。

ロールは仮想IDであるため、ロールはパーマネントキーを所有しません。Tencent Cloud CAMは、一時的なセキュリティ証明書の発行に用いるSTS APIをご提供しています。

使用方法と事例の詳細については、ロールの使用のドキュメントを参照するか、またはロールの作成ドキュメントを参照し、一時的なセキュリティ証明書の発行方法をご確認ください。一時的なセキュリティ証明書には通常、**限定的なポリシー**（アクション、リソース、条件）および**限定的な時間**（有効期間の開始および終了時間）のみが含まれます。このため、発行されたセキュリティ証明書は自ら配信または直接使用することができます。

一時的なセキュリティ証明書の発行インターフェースを呼び出すと、1対の一時キー（tmpSecretId/tmpSecretKey）および1個のセキュリティトークン（sessionToken）を取得でき、これらがCOSへのアクセスに用いることができるセキュリティ証明書を構成します。その例を次に示します。

41文字のセキュリティトークン（SecurityToken）：5e776c4216ff4d31a7c74fe194a978a3ff2xxxxxx

36文字の一時アクセスキーID（SecretID）: AKIDcAZnqgar9ByWq6m7ucIn8LNEuYxxxxxx

32文字の一時アクセスキーKey（SecretKey）: VpxrX0IMCpHXWL0Wr3KQNCqJixxxxxxx
このインターフェースはまた、 `expiration` フィールドによって一時的なセキュリティ証明書の有効期間を返します。これは、この時間内においてのみ、このセキュリティ証明書を使用してリクエストを発行できることを意味します。
Tencent Cloud COSは一時キーの発行に使用できる、簡単なサーバーSDKをご提供しています。COS STS SDKにアクセスして取得できます。一時的なセキュリティ証明書を取得した後、REST APIを使用してリクエストを発行する際、HTTPヘッダーまたはPOSTリクエストパケットのform-dataに `x-cos-security-token` フィールドの、このリクエストの使用を表すセキュリティトークンを渡し、その後一時アクセスキーペアを使用してリクエスト署名を計算する必要があります。COS SDKを使用したアクセスについては、各SDKドキュメントの関連するパートをご参照ください。

アクセスドメイン名

REST API

リージョンとアクセスドメイン名のドキュメントに、REST APIによるアクセスに用いることができるドメイン名をリストアップしています。
COSは仮想ホスティング形式のドメイン名を使用してバケットにアクセスすることを推奨しています。HTTPリクエストを発行する際には、 `<BucketName-APPID>.cos.<Region>.myqcloud.com` のように `Host` ヘッダーにアクセスしたいバケットを直接入力します。仮想ホスティング形式のドメイン名を使用すると、仮想サーバー「ルートディレクトリ」に類似した機能を実装でき、favicon.ico、robots.txt、crossdomain.xmlに類似したファイルのホスティングに用いることができます。これらのファイルは、多くのアプリケーションプログラムがウェブサイトのホスティングを識別する際、デフォルトで仮想サーバー「ルートディレクトリ」の位置から検索する内容となります。
パス形式のリクエストを使用してバケットにアクセスすることもできます。例えば、 `cos.<region>.myqcloud.com/<BucketName-APPID>/` のパスを使用してバケットにアクセスします。それに応じて、リクエストHostおよび署名には `cos.<region>.myqcloud.com` を使用する必要があります。当社のSDKはデフォルトではこのアクセス方式をサポートしていません。

静的ウェブサイトのドメイン名

バケットの静的ウェブサイト機能を有効にしている場合は、仮想ホスティング形式のドメイン名が割り当てられ、それによって関連機能を使用できます。静的ウェブサイトドメイン名の形式はREST APIとは異なる点があり、特定のインデックスページ、エラーページおよびリダイレクト設定のほか、静的ウェブサイトドメイン名ではGET/HEAD/OPTIONS Objectなどの数種類の操作のみがサポートされ、アップロードまたはリソースの設定などの操作はサポートされないという違いがあります。
静的ウェブサイトドメイン名は、例えば `<BucketName-APPID>.cos-website.<Region>.myqcloud.com` のようになります。コンソール上でバケットの「基本設定-静的ウェブサイト設定」

モジュールによってこのドメイン名を取得することもできます。

プライベートネットワークアクセス

Tencent Cloud COSのアクセスドメイン名にはインテリジェントDNS解決を使用しており、インターネットの様々なキャリア環境の下で、お客様のCOSアクセスにとって最適なリンクを検知および指定します。

Tencent Cloud内にデプロイしたCVMサービスをCOSへのプライベートネットワークアクセスに使用している場合は、まずCVMとCOSバケットが同じリージョンに所属していることを確認し、その後CVM上で `nslookup` コマンドを使用してCOSドメイン名を解決します。プライベートIPが返された場合、CVMとCOS間はプライベートネットワークアクセスであり、そうでない場合はパブリックネットワークアクセスであることがわかります。

Tencent Cloud内にデプロイしたCVMサービスのリージョンがCOSバケットの所属リージョンと異なるが、COSのアベイラビリティリージョンの範囲である場合は、COSのプライベートネットワークグローバルアクセラレーションドメイン名によってファイルにアクセスすることで、CVMとCOSのクロスリージョンアクセスを実現することができます。

プライベートネットワークアクセスの判断方法

同一リージョン内でのTencent Cloud製品へのアクセスは、プライベートネットワーク接続によって行うことができ、それによるプライベートネットワークトラフィックには料金が発生しません。このため、Tencent Cloudの製品をお選びになる際は、できるだけ同一のリージョンを選択すると料金の節約になります。

注意：

パブリッククラウドリージョンと金融クラウドリージョン間のプライベートネットワークは相互運用されていません。

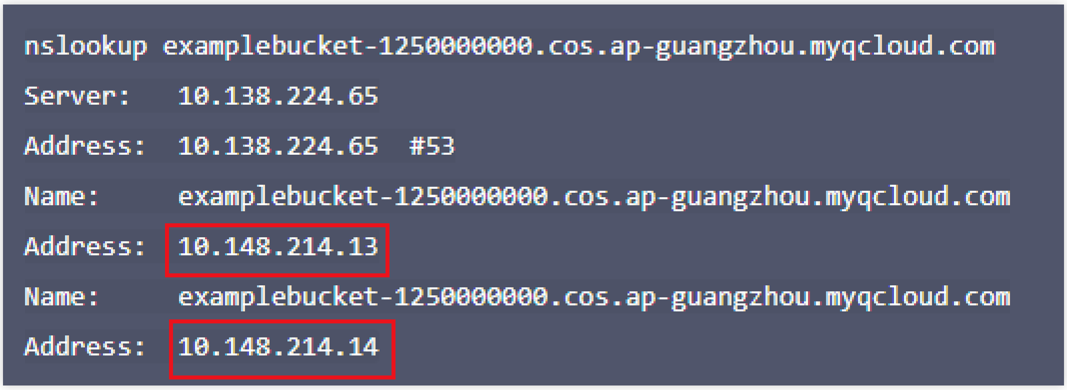
プライベートネットワークアクセスかどうかを確認するには、次の方法をご参照ください。

Tencent CVMのCOSアクセスを例にとると、プライベートネットワークを使用したCOSアクセスかどうかを判定するには、CVM上で `nslookup` コマンドを使用してCOSドメイン名を解決します。プライベートネットワークIPが返された場合、CVMとCOS間はプライベートネットワークアクセスであり、そうでない場合はパブリックネットワークアクセスであることがわかります。

説明：

プライベートIPアドレスの一般的な形式は `10.*.*.*` 、 `100.*.*.*` であり、VPCネットワークは一般的に `169.254.*.*` などです。これらの形式のIPはすべてプライベートネットワークに該当します。

`examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com` がターゲットバケットアドレスである場合、 `nslookup` コマンドを実行すると次の情報を見ることができます。



```
nslookup examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com
Server:   10.138.224.65
Address:  10.138.224.65  #53
Name:     examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com
Address:  10.148.214.13
Name:     examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com
Address:  10.148.214.14
```

この中の `10.148.214.13` と `10.148.214.14` という2つのIPが、プライベートネットワークによるCOSアクセスであることを表しています。

接続性のテスト

基本的な接続性テスト

COSはHTTPプロトコルを使用して外部へのサービス提供を行います。最も基本的な `telnet` ツールを使用して、COSアクセスドメイン名のポート80に対するアクセステストを行うことができます。
パブリックネットワークを経由するアクセスの例は次のとおりです。

```
telnet examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com 80
Trying 14.119.113.22...
Connected to gz.file.myqcloud.com.
Escape character is '^]'.
```

同リージョンのTencent Cloud CVM（Classic network）を経由するアクセスの例は次のとおりです。

```
telnet examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com 80
Trying 10.148.214.14...
Connected to 10.148.214.14.
Escape character is '^]'.
```

同リージョンのTencent Cloud CVM（VPCネットワーク）を経由するアクセスの例は次のとおりです。

```
telnet examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com 80
Trying 169.254.0.47....
Connected to 169.254.0.47.
Escape character is '^]'.
```

どのアクセス環境であっても、コマンドに`Escape character is '^]'.`フィールドが返された場合は接続に成功したことを表します。

インターネット接続テストの説明

インターネット経由でCOSにアクセスする場合はキャリアネットワークを経由しますが、キャリアネットワークはお客様がICMPプロトコルの `ping` または `traceroute` などのツールを使用して接続性テストを行うことを禁じている可能性があります。このため、TCPプロトコルのツールを使用して接続性テストを行うことをお勧めします。

注意：

インターネット経由のアクセスは様々なネットワーク環境の影響を受ける可能性があります。アクセスがスムーズにいかない場合は、ローカルネットワークリンクのトラブルシューティングを行うか、または現地のキャリアに連絡してフィードバックを受けてください。

ご利用のキャリアがICMPプロトコルを許可している場合は、 `ping` 、 `traceroute` または `mtr` ツールを使用してリンクの状況を確認することができます。キャリアがICMPプロトコルの使用を許可していない場合は、 `psping` （Windows環境。Microsoft公式サイトからダウンロードしてください）または `tcping` （クロスプラットフォームソフトウェア）などのツールを使用して遅延テストを行うことができます。

プライベートネットワーク接続テストの説明

同リージョンのTencent Cloud VPCネットワーク経由でCOSにアクセスする場合は、ICMPプロトコルの `ping` または `traceroute` などのツールを使用して接続性テストを行えない可能性があります。基本的な接続性テストの、 `telnet` コマンドを使用してテストを行うことをお勧めします。

`psping` または `tcping` などのツールを使用して、アクセスドメイン名のポート80に対する遅延テストを直接行ってみることもできます。テストの前に、 `nslookup` コマンドによってすでに問い合わせを行っており、アクセスドメイン名がプライベートネットワークアドレスに正しく解決されていることを確実に確認してください。

バケット

バケットの概要

最終更新日：：2024-06-26 10:57:13

概要

バケット（Bucket）とはオブジェクトのキャリアであり、オブジェクトを入れておくための「容器」と理解することができます。この「容器」には容量の上限がありません。オブジェクトはフラットな構造でバケットに保存され、フォルダやディレクトリの概念はありません。ユーザーはオブジェクトを単一または複数のバケットに保存することができます。

説明：

各バケットには任意の数量のオブジェクトを格納できますが、同じルートアカウント下のバケットは最大200個までしか作成できません。

バケット命名ルール

バケットの命名はバケット名（BucketName）とAPPIDの2つの部分を組み合わせ、それらをハイフン「-」でつないで行います。例えば `examplebucket-1250000000` の場合、examplebucketがユーザーのカスタマイズした文字列、1250000000がシステムの生成した数字列（APPID）です。API、SDKのサンプルでは、バケットの命名形式は `<BucketName-APPID>` となっています。

バケット名（BucketName）：ユーザーが手動で入力する一連の文字であり、命名ルールは次のとおりです。

小文字アルファベットと数字[a-z，0-9]、ダッシュ「-」とそれらの組み合わせのみサポートしています。

バケット名の最大文字数は、**リージョンの略称**および**APPID**の文字数の影響により、組み合わせたリクエストドメイン名全体の文字数合計が最大60文字までとなります。例えば、リクエストドメイン名 `123456789012345678901-1250000000.cos.ap-beijing.myqcloud.com` の場合は合計60文字です。

バケット名は「-」を先頭または末尾にすることはできません。

APPIDは、Tencent Cloudのアカウント申請が成功した後に取得するアカウント番号です。システムによって自動的に割り当てられ、固有性と一意性を有し、アカウント情報で確認できます。コンソール上でバケットを作成する場合はユーザーが入力する必要はありませんが、ツール、API、SDKを使用する場合はAPPIDを指定する必要があります。

有効なバケットの命名例を次に示します。

examplebucket-1-1250000000

mybucket123-1250000000

1-newproject-1250000000

バケットの所属リージョン

リージョン（Region）とは、COSのデータセンターがある地域を指します。COSではユーザーが異なるリージョンにバケットを作成することが可能であり、お客様はご自身の業務に最も近いリージョンを選択してバケットを作成することで、低遅延、低コストおよびコンプライアンスの要件を満たすことができます。

例えば、お客様の業務が華南地域に分布している場合は、広州リージョンでのバケット作成を選択することで、オブジェクトのアップロード、ダウンロード速度を向上させることができます。リージョンに関するその他の情報については、リージョンとアクセスドメイン名のドキュメントをご参照ください。

注意：

リージョンはバケットを作成する際に必ず指定するもので、一度指定すると変更できません。このバケット下のすべてのオブジェクトは対応するデータセンターに保存されます。現時点ではオブジェクトのレベル別のリージョン設定はサポートしていません。

権限の分類

バケットはデフォルトで、パブリック権限とユーザー権限という2種類の権限タイプをご提供します。

説明：

バケットがプライベート読み取り/書き込みの場合、または指定のアカウントに対してユーザー権限を承認している場合は、オブジェクトのリクエストの際に、ID認証用の署名を含める必要があります。署名に関する説明については、リクエスト署名をご参照ください。

バケットがパブリック読み取り/プライベート書き込みまたはパブリック読み取り/書き込みの場合、オブジェクトのリクエストの際に署名を含める必要はなく、匿名のユーザーが直接リンクを通じてオブジェクトにアクセスできます。データ漏洩のリスクがありますので、慎重に設定してください。

パブリック権限

パブリック権限には、プライベート読み取り/書き込み、パブリック読み取り・プライベート書き込み、パブリック読み取り/書き込みが含まれます。そのアクセス権限はCOSコンソール上のバケットの**権限管理**で変更できます。アクセス権限に関するその他の説明については、アクセス制御の基本概念をご参照ください。

プライベート読み取り/書き込み

そのバケットの作成者および権限を付与されたアカウントのみが、そのバケット内のオブジェクトに対する読み取り/書き込み権限を有し、それ以外のいかなる人もバケット内のオブジェクトに対する読み取り/書き込み権限を持ちません。バケットのアクセス権限はデフォルトではプライベート読み取り/書き込みとなっており、ご使用を推奨します。

パブリック読み取り・プライベート書き込み

あらゆる人（匿名のアクセス者を含む）がそのバケット内のオブジェクトに対する読み取り権限を有しますが、バケット内のオブジェクトに対する書き込み権限を持つのはバケットの作成者および権限を付与されたアカウントのみです。

パブリック読み取り/書き込み
あらゆる人（匿名のアクセス者を含む）がそのバケット内のオブジェクトに対する読み取り権限と書き込み権限を有します。使用は推奨しません。

ユーザー権限

ルートアカウントはデフォルトではバケットのすべての権限を有しています（完全制御）。このほかCOSはサブアカウントに対するデータ読み取り、データ書き込み、権限読み取り、権限書き込み、さらには完全制御の最高権限の付与もサポートしています。

バケットの操作

ユーザーはTencent Cloudコンソール、ツール、API、SDKなどの複数の方式によってバケットを管理し、属性を設定することができます。例えば、バケットを静的ウェブサイトのホスティング用に設定することや、バケットのアクセス権限設定などを行うことができます。次に一部機能の設定ガイドを列挙します。バケット機能設定に関するその他の情報については、バケット概要をご参照ください。
バケットの作成
静的ウェブサイトの設定
アクセス権限の設定
リンク不正アクセス防止の設定

関連説明

COSはフラットな構造でオブジェクトを保存しており、フォルダの概念はありません。詳細については、オブジェクトの概要のドキュメントの「フォルダとディレクトリ」のパートをご参照ください。
各ルートアカウント（同一のAPPID）は複数のバケットを作成可能です。数量の上限は200個まで（リージョンを区別せず）ですが、バケット内のオブジェクト数に制限はありません。
Tencent Cloud COSでは、同一のAPPID下のバケット名は一意のものであり、名前の変更はできません。
バケットは一度作成するとリネームできません。バケットを削除してから新たに作成し、再度命名することのみ可能です。
ユーザーがバケットを作成する際は、所属リージョンをよくご確認ください。リージョンは一度設定すると変更できません。

バケットの作成

最終更新日：：2024-06-26 10:57:13

ファイルをCOS（Cloud Object Storage）に保存する際、先にオブジェクトを保存するためのバケットを作成する必要があります。バケットの作成は、コンソール、ツール、APIまたはSDK方式で行うことができます。
バケットを作成すると、そのバケットにオブジェクトをアップロードできるようになります。また、バケットに他の機能を設定することもできます。例えば、静的ウェブサイトの設定、バケットタグの設定 、バケット暗号化の設定などです。その他の設定ガイドについては、コンソールの概要をご参照ください。

制限事項

1. 同一のルートアカウント下に作成できるバケット数は最大200個までです。
2. バケットはいったん作成すると、バケット名および所属リージョンの変更はできません。
3. 同一のルートアカウントにおけるバケット名は固有であり、リネームもサポートしていません。
4. バケット名は小文字アルファベットおよび数字、すなわち[a-z，0-9]、ダッシュ「-」およびそれらの組み合わせのみサポートしています。バケット名の最大文字数は、**リージョンの略称**および**APPID**の文字数の影響により、組み合わせたリクエストドメイン名全体の文字数合計が最大60文字までとなります。例えば、リクエストドメイン名 `12345678901234567890l-1250000000.cos.ap-beijing.myqcloud.com` の場合は合計60文字です。
5. バケット名は「-」を先頭または末尾にすることはできません。

利用方法

COSコンソールの使用

COSコンソールを使用してバケットを作成することができます。詳細については、バケットの作成のコンソールドキュメントをご参照ください。

ツールの使用

COSBrowserツール、COSCMDツールなどを使用してバケットを作成することができます。詳細については、ツールの概要をご参照ください。

REST APIの使用

REST APIを直接使用してバケット作成リクエストを送信できます。 詳細については、PUT BucketのAPIドキュメントをご参照ください。

SDKの使用

SDKのバケット作成メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

Node.js SDK

PHP SDK

Python SDK

ミニプログラムSDK

バケットの削除

最終更新日：：2024-06-26 10:57:13

特定の状況下でバケットを削除する必要がある場合、コンソール、ツール、APIまたはSDK方式によってバケットを削除することができます。

注意：

バケットは削除すると元に戻せませんので、慎重に操作してください。

制限事項

現時点ではデータが消去されているバケットの削除のみサポートしています。バケット内にオブジェクトまたはファイルフラグメントが残っている場合は削除に失敗します。バケットの削除を実行する前に、バケット内のオブジェクトが空になっていることをご確認ください。その方法についてお知りになりたい場合は、バケットのクリアにお進みください。

バケットを削除する際は、操作を行うIDがこの操作の権限承認をすでに受けていることを確実にするとともに、正しいバケット名（Bucket）およびリージョン（Region）パラメータを渡していることを確認する必要があります。

利用方法

COSコンソールの使用

COSコンソールを使用してバケットを削除することができます。詳細については、バケットの削除のコンソールガイドドキュメントをご参照ください。

ツールの使用

COSBrowserツール、COSCMDツールなどを使用してバケットを削除することができます。詳細については、ツールの概要をご参照ください。

REST APIの使用

REST APIを直接使用してバケット削除リクエストを送信できます。 詳細については、DELETE BucketのAPIドキュメントをご参照ください。

SDKの使用

SDKのバケット削除メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

- Android SDK
- C SDK
- C++ SDK
- .NET SDK
- Go SDK
- iOS SDK
- Java SDK
- JavaScript SDK
- Node.js SDK
- PHP SDK
- Python SDK
- Mini Program SDK

オブジェクト

オブジェクトの概要

最終更新日：：2024-06-26 10:57:13

概要

オブジェクト（Object）とはCloud Object Storage（COS）の基本ユニットであり、画像、ドキュメント、オーディオビデオファイルなどのあらゆるフォーマットタイプのデータであると理解することができます。バケット（Bucket）はオブジェクトのキャリアであり、各バケットには任意の数量のオブジェクトを格納できます。オブジェクトはCOSにおいて様々なストレージタイプを指定できます。詳細については、ストレージタイプの概要をご参照ください。

各オブジェクトはオブジェクトキー（ObjectKey）、オブジェクト値（Value）、オブジェクトメタデータ（Metadata）で構成されます。

オブジェクトキー（ObjectKey）：オブジェクトキーはオブジェクトのバケット内での固有識別子であり、わかりやすく言えば、ファイルパスであると理解することができます。API、SDKの例では、オブジェクトの命名フォーマットは `<ObjectKey>` となります。

オブジェクト値（Value）：アップロードしたオブジェクト自体であり、わかりやすく言えば、ファイルの内容（Object Content）であると理解することができます。

オブジェクトメタデータ（Metadata）：一対のキーバリューペアであり、わかりやすく言えば、ファイルの変更時間、ストレージタイプなどのファイルの属性であると理解することができます。オブジェクトをアップロード後に照会することができます。

オブジェクトキー

Tencent Cloud COS内のオブジェクトは有効なオブジェクトキーを持つ必要があります。オブジェクトキー（ObjectKey）はオブジェクトのバケット内での固有識別子です。

例：オブジェクトアクセスアドレス `examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com/folder/picture.jpg` において、オブジェクトキーは `folder/picture.jpg` です。

命名ルール

オブジェクトキーの名称には任意のUTF-8文字を使用できます。他のアプリケーションとの間での互換性を最大限に確保するため、名前には大文字と小文字のアルファベット、数字[a-z、A-Z、0-9]およびそれらの組み合わせを使用することをお勧めします。

コードの長さは最大850バイトまでです。
スラッシュ `/` またはバックスラッシュ `\\` で始めることはできません。
オブジェクトキーではASCII制御文字のうち、上矢印(↑)、下矢印(↓)、右矢印(→)、左矢印(←)をサポートしておらず、それぞれCAN(24)、EM(25)、SUB(26)、ESC(27)に対応します。
`*` 、 `%` などの特殊文字を直接ファイル名に使用することはできるだけ避けます。

説明：

ユーザーがアップロードしたファイルまたはフォルダの名前に中国語が含まれている場合、このファイルまたはフォルダへのアクセスおよびリクエストの際、中国語部分はURL Encodeルールに従ってパーセントエンコーディングに変換されます。
例： `文档.doc` にアクセスする場合、オブジェクトキーは `文档.doc` ですが、実際に読みとるものはURL Encodeルールに従って変換されたパーセントエンコーディングの `%e6%96%87%e6%a1%a3.doc` となります。
有効なオブジェクトキー命名の例を次に示します。
doc/exampleobject
my.great_photos-2016/01/me.jpg
videos/2016/birthday/video.wmv

特殊文字

文字の中には、オブジェクトキー内では16進数形式でURL内でエンコードまたは引用しなければならないものがあり、中には印刷できないものもあります。このためこのような文字はブラウザで処理できない可能性があり、特殊な処理が必要となります。特殊処理が必要な可能性がある文字は次のとおりです。

,	:	;	=
&	$	@	+
?	ASCII文字範囲：00-1F 16進数（0-31 10進数）および7F（127 10進数）	（スペース）	

また、いくつかの文字は、アプリケーションプログラム間で整合性を保つために大量の特殊処理を必要とするため、使用を避けることをお勧めします。避ける必要がある文字は次のとおりです。

`	^	"	\\
{	}	[	]
~	%	#	|
*	>	<	ASCII 128-255 10進数

オブジェクトアクセスアドレス

オブジェクトのアクセスアドレスはバケットアクセスアドレスとオブジェクトキーで構成されます。その構造形式は `<バケットドメイン名>/<オブジェクトキー>` です。
例：オブジェクト `exampleobject.txt` を広州（華南）のバケット `examplebucket-1250000000` にアップロードする場合、 `exampleobject.txt` のアクセスアドレスは `examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com/exampleobject.txt` となります。

フォルダとディレクトリ

COS自体にはフォルダやディレクトリの概念がないため、COSはオブジェクト `project/a.txt` をアップロードしても、 `project` フォルダを作成するわけではありません。ユーザーの使用上の習慣に沿って、COSはコンソール、COS browserなどのグラフィックツールで「フォルダ」または「ディレクトリ」という表示方法を疑似的に再現しています。具体的には、キーの値が `project/` 、内容が空のオブジェクトを作成することで、従来のフォルダを模した表示方法を実現します。
例：API、SDKによってオブジェクト `project/doc/a.txt` をアップロードし、区切り記号 `/` で「フォルダ」の表示方式を再現すると、コンソール上で「フォルダ」 `project` および `doc` が見えるようになります。このうち `doc` は `project` の1つ下のレベルの「フォルダ」であり、 `a.txt` が含まれます。

注意：

バケット内で様々なオブジェクトが様々な分散型クラスター内にフラットに分散している場合は、あるオブジェクトキーのプレフィックス容量のサイズを直接取得することはできず、各オブジェクトのサイズを累計することでしか取得できません。

フォルダおよびディレクトリの削除操作を行う場合は、状況は比較的複雑になります。詳細は次のとおりです。

削除経路	オブジェクトまたはフォルダの削除	結果
コンソール	フォルダ `project`	オブジェクトキーのプレフィックスが `project/` である全オブジェクトが削除されます
コンソール	オブジェクト `project/doc/a.txt`	`project` および `doc` フォルダは維持されます
API、SDK	オブジェクト `project/` または `project/doc/`	オブジェクト `project/doc/a.txt` は維持されます。フォルダ内のオブジェクトを一括削除したい場合は、 コードトラバーサルによってフォルダ内のオブジェクトの削除機能を実現する必要があります

オブジェクトメタデータ

オブジェクトメタデータはオブジェクト内の一対のキーバリューペアであり、サーバーがHTTPプロトコルによってHTML資料をブラウザに渡す前に送信する文字列で、HTTP Headerとも呼ばれます。オブジェクトのアップロード時にHTTP Headerを変更することで、ページのレスポンス形式を変更したり、キャッシュ時間の変更などの設定情報を伝達したりすることもできます。

オブジェクトメタデータには、システムメタデータとユーザー定義のメタデータという2種類のメタデータがあります。

説明：

オブジェクトのHTTP Headerを変更してもオブジェクト自体は変更されません。

システムメタデータ

オブジェクトの属性情報、例えばオブジェクトの変更時間、オブジェクトサイズ、ストレージタイプなどを指します。

名称	説明
Content-Length	RFC 2616で定義されたHTTPリクエスト内容の長さ（バイト）
Last-Modified	オブジェクトの作成日または前回変更日（どちらか遅い方）
x-cos-version-id	オブジェクトバージョンID。バケット上でバージョン管理を有効にしている場合は、オブジェクトのバージョンIDが返され、同一オブジェクトの過去のバージョンの識別に用いることができます
ETag	PUT Objectによってアップロードしたオブジェクトの場合は、アップロードされたファイルコンテンツのMD5値であり、マルチパートアップロードまたは過去のバージョンのAPIを使用してアップロードしたオブジェクトの場合は、アップロードされたファイルコンテンツの一意のIDです。検証機能は有していません

ユーザー定義のメタデータ

Content-Type，Cache-Control，Expires，x-cos-meta-* などの、オブジェクトのカスタマイズパラメータのことです。具体的な情報についてはカスタムオブジェクトHeadersをご参照ください。

名称	説明
Cache-Control	RFC 2616で定義されたキャッシュポリシー。オブジェクトメタデータとして保存されます
Content-Disposition,Content-Encoding,Content-Type	RFC 2616で定義されたファイル名/エンコード形式/コンテンツタイプ（MIME）。オブジェクトメタデータとして保存されます

Expires	RFC 2616で定義されたキャッシュ失効時間。オブジェクトメタデータとして保存されます
x-cos-acl	オブジェクトのアクセス制御リスト（ACL）の属性を定義します。有効値：private、public-read-write、public-read。デフォルト値：private
x-cos-grant-*	被付与者にある種の権限を付与します
x-cos-meta-*	ユーザーによるカスタマイズが許可されたヘッダー情報。オブジェクトメタデータとして返されます（サイズ制限2K）
x-cos-storage-class	オブジェクトのストレージレベルを設定します。列挙値：STANDARD、STANDARD_IA、ARCHIVE、デフォルト値：STANDARD
x-cos-server-side-encryption	オブジェクトに対しサーバー側での暗号化を有効にするかどうか、および暗号化方式を指定します

オブジェクト操作

ユーザーはTencent Cloudコンソール、ツール、API、SDKなどの複数の方式でオブジェクトを管理することができます。

注意：

アップロード方式はサポートする最大アップロードファイルサイズによって、シンプルアップロードとマルチパートアップロードの2種類に分けられます。

シンプルアップロードを使用する場合、オブジェクトサイズは5GB以内に制限されます。

マルチパートアップロードを使用する場合、各パートのサイズ制限は5GB以内であり、パート数は10000未満とする必要があるため、アップロードできるオブジェクトは最大で約48.82TBとなります。制限に関するその他の説明については、仕様と制限をご参照ください。

オブジェクトのアップロード完了後、ユーザーはコンソール上でオブジェクトについて関連の設定を行うことができます。具体的には次の内容をご参照ください。

オブジェクトの検索
オブジェクト情報の確認
オブジェクトのアクセス権限の設定
カスタムHeadersの設定

オブジェクトサブリソース

COSにはバケットとオブジェクトにバインドされたサブリソースがあります。サブリソースはオブジェクトに従属するものであり、サブリソースはそれ自体で存在することはなく、常に他のエンティティ（オブジェクトまたはバケットなど）にバインドされます。アクセス制御リスト（Access Control List）は特定のオブジェクトへのアクセス制御情報のリストであり、これはCOS内のオブジェクトのサブリソースです。
アクセス制御リストには、権限の被付与者およびその付与された許可を識別できる権限リストが含まれ、これによってオブジェクトへのアクセス制御を実現しています。オブジェクトの作成時に、ACLはオブジェクトを完全に制御できるオブジェクト所有者を識別します。ユーザーはオブジェクトのACLを検索することや、ACLを更新された権限リストに置き換えることができます。

説明：

ACLに更新があった場合は必ず現在のACLを置き換える必要があります。

アクセス権限のタイプ

COSではオブジェクトに、**パブリック権限**と**ユーザー権限**という2種類の権限タイプを設定することができます。
パブリック権限：継承権限、プライベート読み取り/書き込みおよびパブリック読み取り/プライベート書き込みが含まれます。
継承権限：オブジェクトがバケットの権限を引き継ぎ、バケットのアクセス権限と一致させます。オブジェクトにアクセスした際、COSが読み取ったオブジェクトの権限がバケットの継承権限であれば、バケットの権限とマッチさせ、アクセスに応答します。何らかの新しいオブジェクトが追加された場合も、バケットの権限はデフォルトで継承されます。
プライベート読み取り/書き込み：オブジェクトにアクセスした際、COSが読み取ったオブジェクトの権限がプライベート読み取り/書き込みであれば、バケットにどのような権限が設定されていても、リクエスト署名がなければオブジェクトにアクセスできません。
パブリック読み取り/プライベート書き込み：オブジェクトにアクセスした際、COSが読み取ったオブジェクトの権限がパブリック読み取りであれば、バケットにどのような権限が設定されていても、オブジェクトはすべて直接ダウンロードできます。
ユーザー権限：ルートアカウントはデフォルトではオブジェクトのすべての権限を所有しています（完全制御）。このほかCOSはサブアカウントに対するデータ読み取り、データ書き込み、権限読み取り、権限書き込み、さらには完全制御の最高権限の付与もサポートしています。

適用ケース

プライベート読み取り/書き込みのバケットで、特定のオブジェクトにパブリックアクセス許可設定を行うこと、またはパブリック読み取り/書き込みのバケットで、特定のオブジェクトへのアクセスに認証が必要となるように設定することができます。

ストレージタイプ

ストレージタイプの概要

最終更新日：：2024-06-26 10:57:13

ストレージタイプは、オブジェクトのCloud Object Storage（COS）におけるストレージのレベルおよびアクティブ度を表すことができます。アクセス頻度の高低および耐障害性の程度に基づいて区分し、COSは、標準ストレージ（マルチAZ）、低頻度ストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ、標準ストレージ、低頻度ストレージ、アーカイブストレージ、ディープアーカイブストレージという複数のオブジェクトストレージタイプをご提供します。それぞれのストレージタイプによって、例えばオブジェクトのアクセス頻度、データ耐久性、データ可用性、アクセス遅延などの特性が異なります。ユーザーは、どのストレージタイプでデータをCOSにアップロードするかを、シーンに応じて選択することができます。

注意：

オブジェクトをアップロードする際、ストレージタイプを設定しなければ、デフォルトのストレージタイプは**標準ストレージ**となります。

マルチAZに関するその他の説明については、マルチAZの特徴の概要をご参照ください。

標準ストレージ（マルチAZ）/標準ストレージ

標準ストレージ（マルチAZ）（MAZ_STANDARD）と標準ストレージ（STANDARD）はどちらもホットデータタイプに該当します。両者はアクセス低遅延、高スループットなどの性能を有し、信頼性と可用性の高い、ハイパフォーマンスなCOSサービスをユーザーにご提供しています。

標準ストレージ（STANDARD）と比較して、標準ストレージ（マルチAZ）はより高いデータ耐久性とサービス可用性を有します。標準ストレージ（マルチAZ）は異なるストレージメカニズムを使用し、データを同一都市の異なるデータセンターに保存することで、同一リージョンの1つのデータセンターに障害が発生しても影響を受けず、ユーザーの業務の安定性をさらに保障することができます。

適用ケース

標準ストレージ（マルチAZ）と標準ストレージはどちらも、ホットビデオ、SNS画像、モバイルアプリケーション、ゲームプログラミング、静的ウェブサイトなどといった、大量のホットファイルへのリアルタイムアクセス、頻繁なデータインタラクションなどの業務シナリオに適しています。

標準ストレージ（STANDARD）は大多数のユースケースをカバーし、ストレージコストは標準ストレージ（マルチAZ）より低く、汎用型のストレージタイプであると言えます。

一方、標準ストレージ（多 AZ）はより高いデータ耐久性とサービス可用性を有し、例えば重要ファイル、商用データ、機微な情報などの、より要件の厳しい業務シナリオに適します。

低頻度ストレージ（マルチAZ）/低頻度ストレージ

低頻度ストレージ（マルチAZ）（MAZ_STANDARD_IA）と低頻度ストレージ（STANDARD_IA）はどちらも、信頼性が高く、ストレージコストが比較的低コストで、アクセス遅延が比較的少ないCOSサービスを提供可能です。両者はストレージ価格を抑えることを基本とした上で、Time To First Byte (TTFB)をミリ秒レベルに保ち、ユーザーがデータ取得シーンで待たされることなく、高速で読み取りを行えるよう保証します。標準ストレージとの明らかな違いは、ユーザーのデータアクセス時にデータ取得料金が発生する点です。

低頻度ストレージ（マルチAZ）は低頻度ストレージとは異なるストレージメカニズムを使用し、データを同一都市の異なるデータセンターに保存することで、ユーザーの業務の安定性をさらに保障し、1つのデータセンターに障害が発生しても影響を受けないようにすることができます。

適用ケース

低頻度ストレージ（マルチAZ）と低頻度ストレージはどちらもアクセス頻度が低い（例えば毎月平均1～2回のアクセス頻度）業務シナリオに適しています。例えば、クラウドストレージデータ、ビッグデータ分析、政府・企業業務データ、低頻度ファイル、監視データなどです。

注意：

低頻度ストレージ（マルチAZ）と低頻度ストレージにはどちらも保存期間およびストレージユニットの最低制限があり、保存期間が30日未満の場合は30日として計算します。単一のストレージファイルが64KB未満の場合は64KBとして計算し、64KB以上の場合は実際のサイズに基づいて計算します。価格に関する情報は、COS価格をご参照ください。

INTELLIGENT_TIERINGストレージ（マルチAZ）/INTELLIGENT_TIERINGストレージ

INTELLIGENT_TIERINGストレージ（マルチAZ）（MAZ_INTELLIGENT_TIERING）タイプのオブジェクトは、標準ストレージ（マルチAZ）レイヤーと低頻度ストレージ（マルチAZ）レイヤーの2つのストレージレイヤーに保存できます。INTELLIGENT_TIERINGストレージ（INTELLIGENT_TIERING）タイプのオブジェクトは標準ストレージレイヤーと低頻度ストレージレイヤーの2つのストレージレイヤーに保存できます。COSはINTELLIGENT_TIERINGストレージ（マルチAZ）/INTELLIGENT_TIERINGストレージのオブジェクトのアクセス頻度に応じて、対応する2つのストレージレイヤーの間で自動的に切り替えを行います。データ取得料金がかからないため、ユーザーのストレージコストを削減できます。INTELLIGENT_TIERINGストレージに関するより詳しい説明は、 INTELLIGENT_TIERINGストレージの概要をご参照ください。

INTELLIGENT_TIERINGストレージ（マルチAZ）はINTELLIGENT_TIERINGストレージとは異なるストレージメカニズムを使用し、データを同一都市の異なるデータセンターに保存することで、ユーザーの業務の安定性をさらに保障し、1つのデータセンターに障害が発生しても影響を受けないようにすることができます。

適用ケース

INTELLIGENT_TIERINGストレージ（マルチAZ）とINTELLIGENT_TIERINGストレージはどちらもデータアクセスモードが一定ではないシナリオに適しています。業務のコスト面での要件が比較的厳しく、かつファイル読み取り性能の感度がそれほど高くなくてもよい場合は、このタイプのストレージを使用することでコストを削減できます。

注意：

INTELLIGENT_TIERINGストレージ（マルチAZ）とINTELLIGENT_TIERINGストレージタイプのファイルについては、単一のストレージファイルのサイズにかかわらず、すべて実際のデータサイズに基づいて課金されます。価格に関する情報は、COS価格をご参照ください。

アーカイブストレージ

アーカイブストレージ（ARCHIVE）はコールドデータタイプに該当し、データ取得時は事前に復元（解凍）する必要があります。信頼性が高く、ストレージコストが極めて低い、長期保存型のCOSサービスの提供が可能です。アーカイブストレージには90日間の最低保存期間の要件があり、かつデータの読み取り前にデータの復元（解凍）を行っておく必要があります。

復元操作は3種類のモードをサポートしています。

高速取得モード：復元タスクは1～5分以内に完了します。

標準取得モード：復元タスクは3～5時間以内に完了します。

バッチ取得モード：復元タスクは5～12時間以内に完了します。

説明：

復元の操作ガイドについては、アーカイブオブジェクトの復元をご参照ください。

データ復元リクエストにはQPS制限が存在します。制限は100回/秒です。

適用ケース

アーカイブストレージは、ファイルデータ、医療用画像、科学資料といったコンプライアンス関連ファイル、ライフサイクル関連ファイル、操作ログなどのアーカイブやリモートディザスタリカバリなど、データを長期保存する必要がある業務シーンに適しています。

注意：

アーカイブストレージには保存期間およびストレージユニットの最低制限があり、保存期間が90日未満の場合は90日として計算します。単一のストレージファイルが64KB未満の場合は64KBとして計算し、64KB以上の場合は実際のサイズに基づいて計算します。価格に関する情報は、COS価格をご参照ください。

ディープアーカイブストレージ

ディープアーカイブストレージ（DEEP_ARCHIVE）では、ユーザーに、高信頼性でありながら、他のストレージの種類と比べてストレージコストが低い、長期保存型のオブジェクトストレージサービスを提供しています。

ディープアーカイブストレージには最低180日の保存期間要件があり、かつデータ読み取りの前にデータの復元を

行っておく必要があります。ディープアーカイブストレージのより詳しい説明は、ディープアーカイブストレージ概要をご参照ください。

復元操作は2種類のモードをサポートしています。

標準取得モード：復元時間は12～24時間です。

バッチ取得モード：復元時間は24～48時間です。

説明：

データ復元リクエストにはQPS制限が存在します。制限は100回/秒です。

適用ケース

ディープアーカイブストレージは、医療用画像データ、ビューデータ、ログデータなどといった、データを長期保存する必要がある業務シーンに適しています。

注意：

ディープアーカイブストレージには保存期間およびストレージユニットの最低制限があり、保存期間が180日未満の場合は180日として計算します。単一のストレージファイルが64KB未満の場合は64KBとして計算し、64KB以上の場合は実際のサイズに基づいて計算します。価格に関する情報は、COS価格をご参照ください。

ストレージタイプの比較

比較項目	標準ストレージ（マルチAZ）	低頻度ストレージ（マルチAZ）	INTELLIGENT_TIERINGストレージ （マルチAZ）	INTELLIGENT_TIE ストレージ
ストレージタイプパラメータ	MAZ_STANDARD	MAZ_STANDARD_IA	MAZ_INTELLIGENT_TIERING	INTELLIGENT_TIE
データ耐久性	99.9999999999%	99.9999999999%	99.9999999999%	99.999999999%
サービス可用性	99.995%	99.995%	99.995%	99.99%
応答	ミリ秒レベル	ミリ秒レベル	ミリ秒レベル	ミリ秒レベル

オブジェクトの最小計算単位	オブジェクトの実際のサイズによって計算	64KB	オブジェクトの実際のサイズによって計算	オブジェクトの実イズによって計算
最短課金期間	制限なし	30日	制限なし	制限なし
サポートするリージョン	現時点では北京、上海、広州、中国香港、シンガポールリージョンのみ	現時点では北京、上海、広州、中国香港、シンガポールリージョンのみ	現時点では北京、上海、広州、シンガポールリージョンのみ	現時点では北京、上海、広州、成都、慶、東京、シンガリージョンのみ
ストレージ料金	比較的高い	比較的高い	切り替え後のストレージタイプのものと同じ	切り替え後のストタイプのものと同
データ取得料金	なし	比較的低い。実際の読み取りデータ量に応じて課金	なし	なし

リクエスト料金	標準	比較的高い	比較的高い。INTELLIGENT_TIERINGオブジェクト監視料金が別途必要	比較的高い。 INTELLIGENT_TIE オブジェクト監視料 別途必要
データ処理	サポートあり	サポートあり	サポートあり	サポートあり

ストレージタイプの切り替え

COSは複数のオブジェクトストレージタイプをご提供します。ストレージタイプは、オブジェクトのCOSにおけるストレージのレベルおよびアクティブ度を表すことができます。実際のご使用中に、ニーズに応じてオブジェクトのストレージタイプを変更したり、オブジェクトを比較的アクティブ度の低いストレージタイプ（低頻度ストレージ、アーカイブストレージ、ディープアーカイブストレージなど）に移行したりすることもできます。

説明：

ストレージタイプを切り替える際は、現在のオブジェクトが保存されているバケットがそのストレージタイプに対応しているかどうかをまず確認してください。

オブジェクトが**アーカイブストレージ/ディープアーカイブストレージ**タイプの場合は、標準ストレージタイプに復元してからでなければ他のストレージタイプに切り替えることができません。詳細については、アーカイブオブジェクトの復元をご参照ください。

ストレージタイプの切り替えに関する状況は下表のとおりです。

ストレージタイプ	変更可能なストレージタイプ	移行可能なストレージタイプおよび移行の優先順位
標準ストレージ（マルチAZ）	低頻度ストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ（マルチAZ）	標準ストレージ（マルチAZ） > 低頻度ストレージ（マルチAZ） > INTELLIGENT_TIERINGストレージ（マルチAZ）
低頻度ストレージ（マルチAZ）	標準ストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ	低頻度ストレージ（マルチAZ） > INTELLIGENT_TIERINGストレージ

	（マルチAZ）	（マルチAZ）
INTELLIGENT_TIERINGストレージ（マルチAZ）	標準ストレージ（マルチAZ）、低頻度ストレージ（マルチAZ）	なし
INTELLIGENT_TIERINGストレージ	標準ストレージ、低頻度ストレージ、アーカイブストレージ、ディープアーカイブストレージ	INTELLIGENT_TIERINGストレージ＞アーカイブストレージ＞ディープアーカイブストレージ
標準ストレージ	INTELLIGENT_TIERINGストレージ、低頻度ストレージ、アーカイブストレージ、ディープアーカイブストレージ	標準ストレージ＞低頻度ストレージ＞INTELLIGENT_TIERINGストレージ＞アーカイブストレージ＞ディープアーカイブストレージ
低頻度ストレージ	INTELLIGENT_TIERINGストレージ、標準ストレージ、アーカイブストレージ、ディープアーカイブストレージ	低頻度ストレージ＞INTELLIGENT_TIERINGストレージ＞アーカイブストレージ＞ディープアーカイブストレージ
アーカイブストレージ	復元後、INTELLIGENT_TIERINGストレージ、標準ストレージ、低頻度ストレージ、ディープアーカイブストレージに変更可能	アーカイブストレージ＞ディープアーカイブストレージ
ディープアーカイブストレージ	復元後、INTELLIGENT_TIERINGストレージ、標準ストレージ、低頻度ストレージ、アーカイブストレージに変更可能	なし

操作ガイドの詳細は次のドキュメントをご参照ください。

ストレージタイプの変更

ライフサイクルの設定

ディープアーカイブストレージ概要

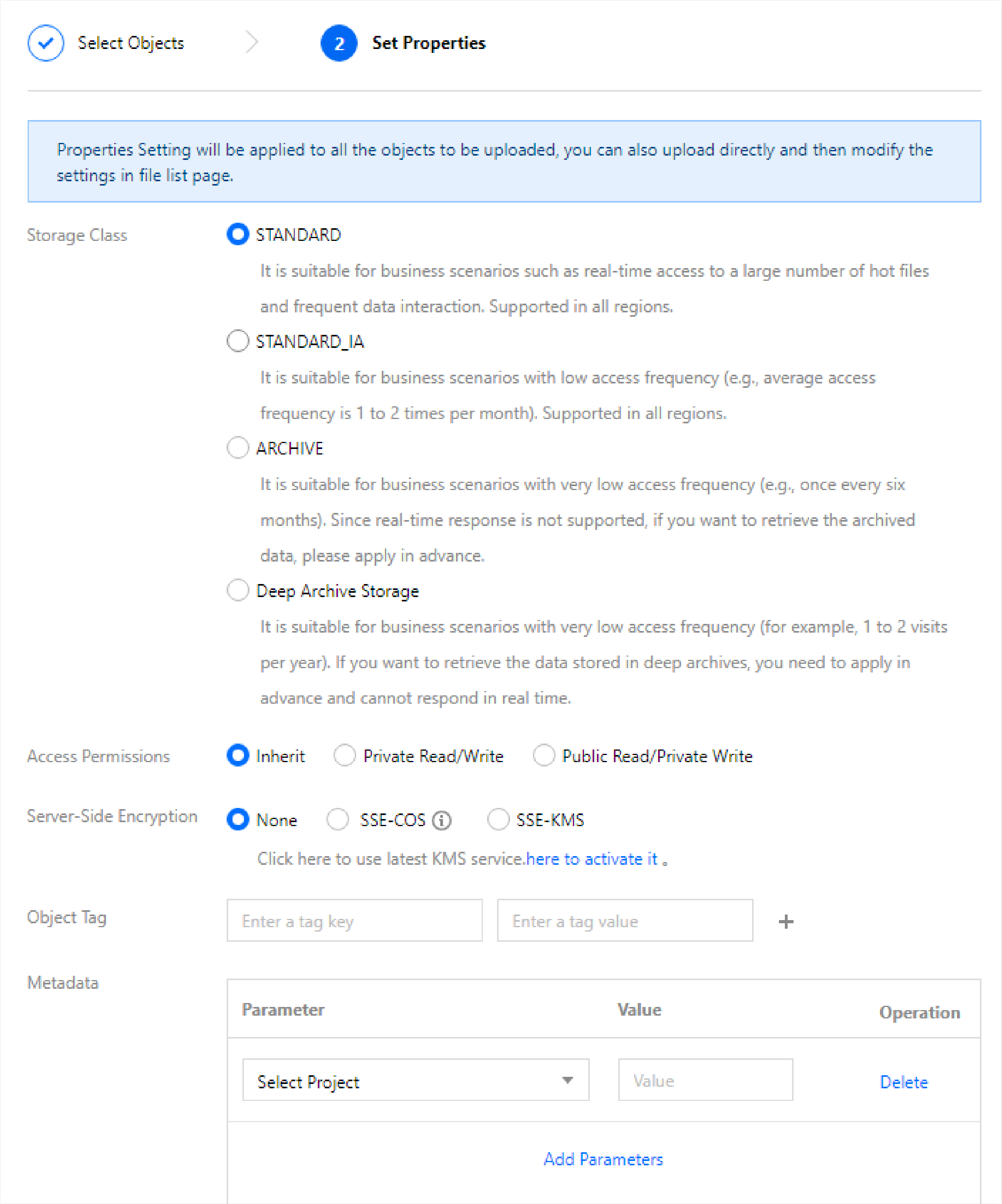
最終更新日：：2024-06-26 10:57:13

概要

ディープアーカイブストレージ（Deep Archive）はCloud Object Storage（COS）が提供する、大量のデータの長期的なアーカイブを可能にするストレージサービスです。ディープアーカイブストレージのストレージ単価はテープストレージレベルであり、ユーザーデータの長期保存に適した低コストのソリューションをご提供します。ユーザーはローカルで複雑なテープライブラリ設定のメンテナンスを行う必要がなく、基盤ストレージメディアのアップデートを気にする必要もありません。COSが提供するAPI、SDK、エコシステムツール、コンソールなどの豊富なヒューマンマシンインタラクション手段により、データ管理を便利かつ低コストに行うことができます。
ディープアーカイブストレージは、データへのアクセス頻度が極めて低いものの、長期的に保存する必要があるケースに適しています。ログのコールドバックアップのケースでは、企業は現地の法律法令の要件に従って、毎日発生するログデータに対しコールドバックアップストレージを行い、分析と追跡に役立てる必要があります。ビューデータや自動運転などの業務では、企業には大量の画像、ビデオなどのメディアファイルが蓄積されますが、それらのデータは使用された後も長期的なアーカイブ保存を必要とします。ディープアーカイブストレージを利用することで、企業はこれらのデータをクラウド上に保存し、必要な場合にのみ復元することで、ストレージコストと運用保守管理の難度を低下させることができます。
ディープアーカイブストレージは99.999999999%（イレブンナイン）のデータ耐久性および99.95%の業務設計上の可用性をご提供します。また、COSがご提供するバージョン管理、地域間コピーなどの機能を利用することで、ユーザーのデータセキュリティはさらに保障されます。

利用方法

1. コンソールを使用したアップロード
1. COSコンソールにログインします。
2. **バケットリスト**をクリックし、サポートされているリージョン（北京リージョンなど）のバケットを作成します。
3. 作成完了後、**ファイルリスト**画面で、**ファイルのアップロード**をクリックします。
4. ローカルファイルを選択してアップロードし、オブジェクトの属性設定で、ストレージタイプに**ディープアーカイブストレージ**を選択します。詳細については、オブジェクトのアップロードをご参照ください。



Select Objects
2 Set Properties

Properties Setting will be applied to all the objects to be uploaded, you can also upload directly and then modify the settings in file list page.

Storage Class

● STANDARD

It is suitable for business scenarios such as real-time access to a large number of hot files and frequent data interaction. Supported in all regions.

○ STANDARD_IA

It is suitable for business scenarios with low access frequency (e.g., average access frequency is 1 to 2 times per month). Supported in all regions.

○ ARCHIVE

It is suitable for business scenarios with very low access frequency (e.g., once every six months). Since real-time response is not supported, if you want to retrieve the archived data, please apply in advance.

○ Deep Archive Storage

It is suitable for business scenarios with very low access frequency (for example, 1 to 2 visits per year). If you want to retrieve the data stored in deep archives, you need to apply in advance and cannot respond in real time.

Access Permissions ● Inherit ○ Private Read/Write ○ Public Read/Private Write

Server-Side Encryption ● None ○ SSE-COS ○ SSE-KMS

Click here to use latest KMS service.here to activate it 。

Object Tag
Enter a tag key
Enter a tag value
+
Metadata
Parameter
Value
Operation
Select Project
Value
Delete
Add Parameters

5. APIを使用したアップロード

PUT Object、POST ObjectまたはInitiate Multipart Uploadインターフェースで、 `x-cos-storage-class` を `DEEP_ARCHIVE` に設定すると、ディープアーカイブへの直接伝送を実現できます。

注意：

Append Objectはディープアーカイブタイプへの直接伝送をサポートしていません。

6. SDKを使用したアップロード
現在COSがリリースしているSDKはすべて、ディープアーカイブへの直接伝送をサポートしています。具体的な方法は、ファイルをアップロードする際に `StorageClass` パラメータを `DEEP_ARCHIVE` に設定し、ディープアーカイブへの直接伝送を実現するものです。
7. ツールを使用したアップロード
COSBrowser、COSCMDツールはディープアーカイブへの直接伝送をサポートしています。具体的な方法は、ファイルをアップロードする際に、ヘッダーフィールドに `x-cos-storage-class` を追加し、 `DEEP_ARCHIVE` に設定してディープアーカイブへの直接伝送を実現するものです。

使用制限

ディープアーカイブストレージには次のような制限があります。ユーザーはご利用中、関連の制限がストレージコストおよびパフォーマンスにもたらす影響にご注意ください。

最小ストレージユニットの制限：最小ストレージユニットは64KBです。64KB未満のオブジェクトは64KBとしてストレージ容量を計算します。

最短保存期間の制限：最短保存期間は180日間です。180日未満の場合は180日として課金されます。

説明：

保存日数は論理上の日数（満24時間を1日とする）であり、カウント開始時間はファイルの変更時間を基に計算します。

取得リクエストのQPS制限：100回/秒です。

リージョンの制限：現在は北京、南京、上海、広州、成都、重慶、東京、シンガポールリージョンでのみサポートしています。その他のリージョンでも順次サポート予定です。

使用制限：ディープアーカイブストレージは現時点ではマルチAZをサポートしていません。すなわち、マルチAZ設定を有効にしたバケットは、現時点ではディープアーカイブストレージタイプを使用することができなくなります。

操作の制限：追加でアップロードしたオブジェクトをディープアーカイブストレージタイプとすることはできません。

取得リクエストの制限：各オブジェクトは1回につき1つの取得リクエストしか実行できず、複数の取得リクエストが繰り返されるとそれらは統合され、最も速いリカバリモードで処理されます。N+1回目に送信された取得リクエストのリカバリモードがN回目より速い場合は、新しい取得リクエストに従って実行されます。そうでない場合は、N+1回目の取得リクエストは失敗します。

データ取得：ディープアーカイブストレージタイプのファイルは、標準取得モードと一括取得モードをサポートしています。標準モードでの復元時間は12～24時間、一括モードでの復元時間は24～48時間です。

INTELLIGENT_TIERINGストレージ概要

最終更新日：：2024-06-26 10:57:13

概要

INTELLIGENT_TIERINGストレージタイプはコールドデータとホットデータの階層分離メカニズムをご提供します。ユーザーデータへのアクセスパターンに応じて、データのホット階層とコールド階層を自動的に切り替えることで、ユーザーデータのストレージコストを削減することができます。

INTELLIGENT_TIERING ストレージは、アクセス方式が固定しない、または予測不可能なデータに適しており、COS はオブジェクトへのアクセス状況を監視し、ストレージ料金はデータが実際に保持されている階層（高頻度アクセス階層、低頻度アクセス階層、アーカイブ階層、ディープアーカイブ階層）に応じて課金されます。ユーザーはビジネスニーズに応じて、アクセスパターンが固定されていないデータを標準ストレージタイプからINTELLIGENT_TIERINGストレージタイプに変換し、クラウドのストレージコストを削減することができます。

注意：

INTELLIGENT_TIERING ストレージは現在、北京、南京、上海、広州、成都、重慶、シンガポール、ムンバイ、東京、フランクフルトでのみサポートされています。

INTELLIGENT_TIERINGストレージタイプは独立したストレージタイプであり、ご利用の際はINTELLIGENT_TIERINGストレージ容量料金およびINTELLIGENT_TIERINGオブジェクト監視料金が発生します。このうちINTELLIGENT_TIERINGストレージ容量料金については、INTELLIGENT_TIERINGストレージ容量パッケージから差し引くことができます。課金に関するより詳しい情報については、製品価格をご参照ください。

MAZ バケットは、INTELLIGENT_TIERING のアーカイブ/ディープアーカイブ階層設定の有効化をサポートしていません。

メリット

ユーザーがデータをアップロードする際に、INTELLIGENT_TIERINGストレージタイプを選択してCOSに保存すると、COSはデータへのアクセス回数を定期的にモニタリングし、一定期間データへのアクセスがない場合、データをよりストレージコストの低いアクセス層に 移動させます。データが再びアクセスされた場合は、再び高頻度アクセス層に移動させることで、データの読み取りパフォーマンスを保障します。INTELLIGENT_TIERINGはコールドデータとホットデータの階層分離ストレージにより、ユーザーがストレージコストと読み取り/書き込みパフォーマンスの最適なバランスをとることができるよう支援します。INTELLIGENT_TIERINGストレージの利用には次のようなメリットがあります。

コストの集約：データを永続ストレージとしてINTELLIGENT_TIERINGストレージタイプで保存する場合、保存期間が長くなるほど、標準ストレージに保存した場合に比べてコストが低下し、最大で約20%のストレージコストを削減することができます。INTELLIGENT_TIERINGストレージタイプはCOSライフサイクルのフローにも関

与します。ユーザーは必要に応じてINTELLIGENT_TIERINGストレージをアーカイブストレージに移行させることで、データストレージコストをさらに低下させることができます。
安定性と耐久性：INTELLIGENT_TIERINGストレージでは、標準ストレージと同等の低レイテンシー、高スループットを体感していただくことができます。また、INTELLIGENT_TIERINGストレージはイレイジャーコーディングによる冗長ストレージ方式を採用し、99.999999999%（イレブンナイン）のデータ信頼性を実現しています。また、データの分割ストレージ、読み取り/書き込み同時実行により、業務可用性は99.99%に達しています。マルチAZアーキテクチャにはINTELLIGENT_TIERINGの特性が同期されているため、設計上のデータ信頼性は99.9999999999%（トゥエルブナイン）に、設計上の業務可用性は99.995%に達します。
利便性・使いやすさ：データのオブジェクトストレージタイプを指定するだけで、INTELLIGENT_TIERINGの特性を適用できます。INTELLIGENT_TIERINGストレージはストレージタイプの一種のため、COSのAPI、SDK、ツール、エコシステムアプリケーションと合わせて用いることが可能です。ユーザーは必要に応じてこれらを使用し、クラウド上に保存したデータの管理に役立てることができます。

サポートするストレージ階層

ストレージ階層の紹介

説明：
64KB 以下の小さなファイルは標準階層に保持され、低頻度アクセス階層、アーカイブ階層、ディープアーカイブ階層にトランジションすることはありません。
ユーザーは、API HeadObject によって返されるヘッダー x-cos-storage-tier を使用してINTELLIGENT_TIERING オブジェクトがどの階層にあるかを確認できます。また、API GetBucket と GetBucketVersions の呼び出しによって返されるオブジェクトリストでは、INTELLIGENT_TIERING オブジェクトは、オブジェクトがどの階層にあるかを確認するためのフィールド StorageTier を返します。

アクセス階層	x-cos-storage-tier
高頻度アクセス階層	FREQUENT
低頻度アクセス階層	INFREQUENT
アーカイブ階層	ARCHIVE_ACCESS
ディープアーカイブ階層	DEEP_ARCHIEVE_ACCESS

高頻度アクセス階層（デフォルトで有効になる）

INT オブジェクトをアップロードした後、オブジェクトはデフォルトで高頻度アクセス階層（FREQUENT）状態になります。オブジェクトがこの階層にある場合、ストレージ料金は現在の地域における標準ストレージ公表価格に従って課金されます。

低頻度アクセス階層（デフォルトで有効になる）

バケットの INTELLIGENT_TIERING 設定を有効にする場合、低頻度アクセス階層に変換する日数を選択する必要があります。オプションには 30 日、60 日、90 日が含まれています。一度設定した日数は変更できません。
オブジェクトが連続 30 日間（または 60 日間または 90 日間）アクセスされない場合、オブジェクトは高頻度アクセス階層から低頻度アクセス階層（INFREQUENT）に変換されます。オブジェクトがこの階層にある場合、ストレージ料金は現在の地域における低頻度ストレージの公表価格で課金されます。この階層にあるオブジェクトがアクセスされると、再び高頻度アクセス階層に戻ります。

アーカイブ階層（オプション）

アーカイブ階層のオブジェクトは、アクセスする前にリカバリーする必要があります。そのため、アーカイブ階層の有効化はオプションであり、ユーザーは 1 つ以上の INTELLIGENT_TIERING アーカイブ設定ルールを追加することによって、指定されたプレフィックスとフラグを持つオブジェクトに対してアーカイブ階層を有効化し、変換日数を設定することができます。アーカイブ階層の変換日数は少なくとも 91 日で、最大値は 730 日です。

説明：

現在の地域がアーカイブストレージをサポートしていない場合、コンソールではアーカイブ設定ルールを追加できず、INTELLIGENT_TIERING オブジェクトはアーカイブ階層にトランジションすることはありません。
INTELLIGENT_TIERING オブジェクトがアーカイブ階層にある場合、ストレージ料金は現在の地域におけるアーカイブストレージの公表価格で請求され、早期削除料金は発生しません。

アーカイブ階層のオブジェクトのリカバリー

オブジェクトが連続 N 日間アクセスされない場合、オブジェクトは低頻度アクセス階層からアーカイブ階層に変換される。アーカイブ階層に入ると、データを読み込む前に PostObjectRestore でリカバリーする必要があります。
通常のアーカイブタイプとは異なり、INT アーカイブ階層のオブジェクトのリカバリーは、標準タイプのレプリカを生成するのではなく、オブジェクト自体が直接高頻度アクセス階層に戻ります。
通常のアーカイブタイプと同様に、INT アーカイブ階層のオブジェクトは、高速、標準、バッチの 3 種類の取得方式をサポートしています。**料金については、高速取得料金のみが請求され、公表価格は同じ**地域における**アーカイブストレージの高速取得料金と同じです。すべての方式の取得リクエスト料金については、標準、バッチ方式の取得料金は無料です。**

ディープアーカイブ階層（オプション）

同様に、ディープアーカイブ階層のオブジェクトは、アクセスする前にリカバリーする必要があります。 ユーザーは、1 つ以上の INTELLIGENT_TIERING ディープアーカイブ設定ルールを追加することで、指定されたプレフィックスとフラグを持つオブジェクトに対してディープアーカイブ階層を有効化することができます。同じルールで、アーカイブ階層とディープアーカイブ階層の変換を同時に設定できます。なお、ディープアーカイブ階層の変換日数は少なくとも 180 日、最大値は 730 日で、**アーカイブ階層の変換日数より大きくなければならない**ことに注意してください。

説明：

現在の地域がディープアーカイブストレージをサポートしていない場合、コンソールではディープアーカイブ設定ルールを追加できず、INTELLIGENT_TIERING のオブジェクトはディープアーカイブ階層にトランジションすることはありません。
INT オブジェクトがディープアーカイブ階層にある場合、ストレージ料金は現在の地域におけるディープアーカイブストレージの公表価格で請求され、早期削除コストは発生しません。

ディープアーカイブ階層のオブジェクトのリカバリー

オブジェクトが連続 M 日間アクセスされない場合、オブジェクトは低頻度アクセス階層/アーカイブ階層からディープアーカイブ階層に変換されます。 一度ディープアーカイブ階層に入ると、データを読み込む前に PostObjectRestore でリカバリーする必要があります。
通常のディープアーカイブタイプとは異なり、INT ディープアーカイブ階層のオブジェクトのリカバリーは、標準タイプのレプリカを生成するのではなく、オブジェクト自体が直接高頻度アクセス階層に戻ります。
通常のディープアーカイブタイプと同様に、INTディープアーカイブ階層のオブジェクトは、標準とバッチの 2 つの取得方式をサポートし、**取得料金と取得リクエスト料金はかかりません。**

INTELLIGENT_TIERING アーカイブ階層とディープアーカイブ階層の設定の有効化

ルールの説明：

バケットが INTELLIGENT_TIERING 設定を有効化すると、ユーザがアップロードした INTELLIGENT_TIERING オブジェクトは、デフォルトでは高頻度アクセス階層と低頻度アクセス階層の間で切り替えられるだけですが、INTELLIGENT_TIERING 設定にアーカイブ階層とディープアーカイブ階層のルールを追加すると、アーカイブ階層とディープアーカイブ階層の間の切り替えを有効化できます。各バケットはアーカイブ階層とディープアーカイブ階層に対して最大 1000 の設定ルールを追加することができます。ルールには以下の要素が含まれています。詳細については、API ドキュメンテーション PUT Bucket IntelligentTieringを参照してください。

ルール名（Id）

アーカイブとディープアーカイブのルールを一意に識別するために使用されます。

状態（Status）

有効化（Enabled）または無効化（Disabled）をサポートします。無効化した状態では、アーカイブ/ディープアーカイブ階層のルールが設定されていても、実際にはアーカイブまたはディープアーカイブ階層に変換されません。

適用範囲（Filter）

ルールの有効範囲を定め、プレフィックスフィルタリングとタグフィルタリングをサポートし、プレフィックス数は 1 を超えず、タグ数は 10 を超えないものとします。

階層設定

1 つのルールでアーカイブ階層とディープアーカイブ階層の変換時間を設定することをサポートします。対応するルールを設定すると、INT オブジェクトはアーカイブ階層、ディープアーカイブ階層へのトランジション変換を有効化します。そうしないと、高頻度アクセス階層と低頻度アクセス階層の間のみで変換を行います。

アーカイブ階層（ARCHIVE_ACCESS）：91 日から 730 日までの設定をサポートします。

ディープアーカイブ階層（DEEP_ARCHIEVE_ACCESS）：180 日から 730 日までの設定をサポートします。

ストレージ階層変換順序

ユーザーは、アーカイブ階層のみの変換を有効化したり、ディープアーカイブ階層のみの変換を有効化したり、アーカイブ階層とディープアーカイブ階層の変換を同時に有効化したりすることができます。設定された低頻度アクセス階層、アーカイブ階層、ディープアーカイブ階層の変換日数に基づき、階層ごとに順次トランジションしていきます。

設定の例	設定の詳細	発効結果
例 1	低頻度アクセス階層は 30 日間、アーカイブ階層は 100 日間、ディープアーカイブ階層は 190 日間です。	連続 30 日間アクセスされない場合、標準アクセス階層から低頻度アクセス階層に変換されます。 連続 100 日間アクセスされない場合、低頻度アクセス階層からアーカイブレベルに変換されます。 連続190 日間アクセスされない場合、アーカイブ階層からディープアーカイブ階層に変換されます。
例 2	低頻度アクセス階層は 30 日間、ディープアーカイブ階層は 190 日間です。	連続 30 日間アクセスされない場合、標準アクセス階層から低頻度アクセス階層に変換されます。 連続190 日間アクセスされない場合、低頻度アクセス階層からディープアーカイブ階層に変換されます。
例 3	低頻度アクセス階層は 60 日間、アーカイブ階層は 91 日間です。	連続 60 日間アクセスされない場合、標準アクセス階層から低頻度アクセス階層に変換されます。 連続 91 日間アクセスされない場合、低頻度アクセス階層からアーカイブレベルに変換されます。

アーカイブ階層とディープアーカイブ階層の INT オブジェクトのリカバリー

オブジェクトが何日も連続してアクセスされない場合、オブジェクトは低頻度アクセス階層からアーカイブ/ディープアーカイブ階層に変換されます。 一度アーカイブ階層とディープアーカイブ階層に入ると、データを読み込む前に PostObjectRestore でリカバリーする必要があります。

通常のアーカイブタイプやディープアーカイブタイプのオブジェクトのリカバリーとは異なり、INT アーカイブ階層とディープアーカイブ階層のオブジェクトのリカバリーでは、**標準タイプのレプリカが生成されず、オブジェクト自体が直接高頻度アクセス階層に戻ります。そのため、INT アーカイブ階層/ディープアーカイブ階層のオブジェクトの取得リクエストを開始する際に、取得日数を指定する必要はありません。**

通常のアーカイブタイプと同様に、INT アーカイブ階層のオブジェクトは、高速、標準、バッチの取得方式をサポートします。通常のディープアーカイブタイプと同様に、INT ディープアーカイブ階層のオブジェクトは、標準とバッチの 2 つの取得方式をサポートします。

料金に関しては、アーカイブ階層の高速取得料金（定価は同じ地域におけるアーカイブストレージの高速取得料金と同じ）を除き、その他の取得料金、取得リクエスト料金、リトリーブレプリカのストレージ料金などは一切かかりません。

		アーカイブストレージタイプ	INTELLIGENT_TIERING オブジェクト-アーカイブ階層	ディープアーカイブストレージタイプ	INTELLIGENT_TIERING オブジェクト-ディープアーカイブ階層
ストレージ料金		現在の地域におけるアーカイブストレージタイプのストレージ料金に基づいて課金されます。		現在の地域におけるディープアーカイブストレージタイプのストレージ料金に基づいて課金されます。	
取得にはレプリカが生成されるかどうか。		はい、保持日数を指定する必要があり、標準ストレージの定価に基づいてレプリカストレージ料金が発生します。	いいえ、レプリカストレージ料金はかかりません。	はい、保持日数を指定する必要があり、標準ストレージの定価に基づいてレプリカストレージ料金が発生します。	いいえ、レプリカストレージ料金はかかりません。
取得方式		高速方式 標準方式 バッチ方式	高速方式 標準方式 バッチ方式	標準方式 バッチ方式	標準方式 バッチ方式
取得料金（単位：USD/GB）	高速	アーカイブ高速取得料金	INTELLIGENT_TIERING アーカイブ階層高速取得料金（定価はアーカイブストレージと同じ）	\\	\\
	標準	アーカイブ標準取得料金	無料	ディープアーカイブ標準取得料金	無料
	バッ	アーカイ	無料	ディープ	無料

	チ	ブバッチ取得料金		アーカイブバッチ取得料金	
取得リクエスト料金（単位：USD/万回）		無料	無料	ディープアーカイブ標準取得リクエスト料金 ディープアーカイブバッチ取得リクエスト料金	無料
読み書きリクエスト料金		リカバリ後の標準ストレージ料金と一致します	INTELLIGENT_TIERING 読み書きリクエスト料金	ディープアーカイブ読み書きリクエスト料金	INTELLIGENT_TIERING 読み書きリクエスト料金

リカバリー中に、HeadObject で INT アーカイブ階層/ディープアーカイブ階層オブジェクトのリカバリー状態を表示できます。

リカバリー状態では、HEAD Object のレスポンスヘッダーには x-cos-restore と x-cos-restore-status が含まれます。例えば、x-cos-restore：ongoing-request="true"，cos-restore-status：tier="bulk"; request-date="Mon, 18 Nov 2019 09:34:50 GMT"。

リカバリーが完了すると、INT オブジェクトは直接高頻度アクセス階層に戻り、HEAD Object の対応するヘッダー x-cos-storage-tier は FREQUENT となります。

利用方法

データをINTELLIGENT_TIERINGストレージタイプとしてCOSに保存するには、まず初めにバケットでINTELLIGENT_TIERING設定を有効にする必要があります。有効にした後、オブジェクトをアップロードする際に**ストレージタイプ**をINTELLIGENT_TIERINGストレージタイプに指定すれば利用できます。

COSコンソールの使用

オブジェクトをアップロードする際にINTELLIGENT_TIERINGストレージとして設定する

ユーザーは次の手順を参照し、オブジェクトをINTELLIGENT_TIERINGストレージタイプとして保存することができます。

1. バケット設定ページでINTELLIGENT_TIERINGストレージ設定を有効にします。詳細なフローについては、INTELLIGENT_TIERINGストレージの設定のドキュメントをご参照ください。
2. ファイルをアップロードし、その際にファイルのストレージタイプを指定します。ファイルのアップロードガイドについては、オブジェクトのアップロードのドキュメントをご参照ください。

注意：

バケットのINTELLIGENT_TIERINGストレージ設定は有効にすると無効化できませんので、慎重に設定してください。

クラウド上のデータをINTELLIGENT_TIERINGストレージに切り替える

ユーザーは次の手順を参照し、アップロード済みの既存データをINTELLIGENT_TIERINGストレージタイプに切り替えることができます。

1. バケット設定ページでライフサイクルルールを作成します。詳細なフローについては、ライフサイクルの設定のドキュメントをご参照ください。
2. 指定するルールの適用範囲を設定し、データをINTELLIGENT_TIERINGストレージに移行します。

INTELLIGENT_TIERING アーカイブとディープ アーカイブ階層の有効化

ユーザーは、必要に応じてバケットの INTELLIGENT_TIERING アーカイブとディープアーカイブ階層の設定を有効化できます。詳細については、INTELLIGENT_TIERING ストレージの設定 ドキュメンテーションを参照してください。

REST APIの使用

次のAPIによってINTELLIGENT_TIERINGストレージを直接設定できます。

1. まず初めに、REST APIを使用してバケットのINTELLIGENT_TIERINGストレージを有効にします。次のAPIドキュメントをご参照ください。

INTELLIGENT_TIERING を有効化し、低頻度階層の転換日数を設定：

PUT Bucket IntelligentTiering（id=default）

GET Bucket IntelligentTiering（id=default）

アーカイブおよびディープアーカイブ階層ルールの設定、削除：

PUT Bucket IntelligentTiering （idはdefaultでない）

GET Bucket IntelligentTiering（idはdefaultでない）

Delete Bucket IntelligentTiering（idはdefaultでない）

2. バケットのINTELLIGENT_TIERINGストレージを有効にした後、次のAPIドキュメントを参照し、オブジェクトをINTELLIGENT_TIERINGストレージタイプとしてアップロードすることができます。

PUT Object

PUT Object - Copy

POST Object

Initiate Multipart Upload

3. オブジェクトのストレージタイプおよび保存されているストレージ層を照会したい場合は、次のAPIドキュメントをご参照ください。

GET Object

HEAD Object

4. REST APIを直接使用してINTELLIGENT_TIERINGストレージタイプのオブジェクトを削除できます。次のAPIドキュメントのパートをご参照ください。

DELETE Object

DELETE Multiple Objects

PUT Bucket lifecycle

Delete Bucket IntelligentTiering

5. INTELLIGENT_TIERING アーカイブ階層とディープアーカイブ階層のオブジェクトを取得するには、以下の API を参照してください。

PostObjectRestore

SDKの使用

現在COSが公開しているSDKは、すべてINTELLIGENT_TIERINGストレージタイプの使用をサポートしています。具体的な方法は、ファイルをアップロードする際にStorageClassパラメータをINTELLIGENT_TIERINGおよびMAZ_INTELLIGENT_TIERINGに設定し、INTELLIGENT_TIERINGストレージおよびINTELLIGENT_TIERINGストレージ（マルチAZ）への直接転送を実現するものです。オブジェクトのアップロードのSDKドキュメントについては、SDKの概要をご参照ください。

使用制限

INTELLIGENT_TIERINGストレージの利用には次のような制限があります。

設定の制限：最初に設定した後は変更できません。変更が必要な場合は、お問い合わせください。低頻度アクセス層に切り替える日数の選択可能な値は30、60、90です。

初期ストレージ層の制限：INTELLIGENT_TIERINGストレージタイプの新規オブジェクトは、デフォルトでは高頻度アクセス層に保存され、一定期間継続してアクセスがない場合のみ、低頻度アクセス層に切り替わります。

最小ストレージユニットの制限：64KB未満のオブジェクトは永続的に高頻度アクセス層に保存され、高頻度アクセス層と低頻度アクセス層との間での切り替えは行われません。単一のストレージファイルはどのようなサイズであっても、実際のサイズに従って課金されます。

操作の制限：アップロードインターフェースを追加して、オブジェクトをINTELLIGENT_TIERINGストレージタイプとしてアップロードすることはサポートされていません。

ライフサイクルの制限：INTELLIGENT_TIERINGストレージタイプはアーカイブストレージまたはディープアーカイブストレージタイプにのみ切り替えることができます。標準ストレージタイプをINTELLIGENT_TIERINGストレージタイプに移行した場合は、高頻度アクセス層に保存されます。低頻度ストレージタイプをINTELLIGENT_TIERINGストレージタイプに移行した場合は、低頻度アクセス層に保存されます。

バケットコピーの制限：バケットをコピーする際、ターゲットバケットでINTELLIGENT_TIERINGストレージ設定を有効にしていなければ、オブジェクトをINTELLIGENT_TIERINGストレージタイプとしてコピーすることはできません。

よくあるご質問

INTELLIGENT_TIERINGストレージはどのように課金されますか。

INTELLIGENT_TIERINGストレージには**INTELLIGENT_TIERINGストレージ容量料金**と**INTELLIGENT_TIERINGオブジェクト監視料金**が含まれます。その内容は次のとおりです。

1. INTELLIGENT_TIERINGストレージ容量料金は、ファイルの存在するストレージ層によって金額が異なります。

ファイルが高頻度層にある場合は、標準ストレージ容量料金に従って課金されます。

ファイルが低頻度層にある場合は、低頻度ストレージ容量料金に従って課金されます。

説明：

標準ストレージ、低頻度ストレージ容量料金はパブリッククラウドリージョンごとに金額が異なります。具体的な価格については、製品価格をご参照ください。

ファイルのアップロードおよびダウンロードの過程ではさらにリクエスト料金とトラフィック料金が発生します。これらの料金計算の例については、トラフィック料金の課金の例およびリクエスト料金の課金の例をご参照ください。

2. INTELLIGENT_TIERINGオブジェクト監視料金は保存したファイル数に応じて計算され、64KB未満のファイルでは料金は発生しません。具体的な価格については、製品価格をご参照ください。

事例

ある企業に1TBのファイルがあり、各ファイルはいずれも64KBより大きく、ファイル数は計10万ファイルで、データはINTELLIGENT_TIERINGストレージタイプとして北京リージョンに保存され、低頻度アクセス層への切り替え期間を30日に指定しているとします。30日ごとに20%のファイル（すなわち2万個のファイル）が低頻度層に移行すると仮定した場合、**30日あたり**のオブジェクト監視料金およびストレージ料金は下表のとおりとなります。

説明：

下の表内の北京リージョンのオブジェクト監視料金月単価は0.25米ドル/1万オブジェクトであり、「1日単価 = 月単価/30」の換算ロジックに基づき、1日の単価は0.00833333米ドル/1万オブジェクト/日となります。

ストレージ日数	30日あたりのオブジェクト監視料金（米ドル）	30日あたりのINTELLIGENT_TIERINGストレージ料金（米ドル）	30日あたりの標準ストレージ料金（米ドル）
30 x 1	0.25米ドル/1万オブジェクト x 10万	1024 x 0.024 / 30 x 30 = 24.58	1024 x 0.024 / 30 x 30 = 24.58
30 x 2	0.25米ドル/1万オブジェク	819.2 x 0.024 / 30 x 30 + 204.8 x 0.08 /	1024 x 0.024 / 30 x 30

	ト x 10万	30 x 30 = 23.35	= 24.58
30 x 3	0.25米ドル/1万オブジェクト x 10万	655.36 x 0.024 / 30 x 30 + 368.64 x 0.08 / 30 x 30 = 22.36	1024 x 0.024 / 30 x 30 = 24.58
30 x 4	0.25米ドル/1万オブジェクト x 10万	524.288 x 0.024 / 30 x 30 + 499.712 x 0.08 / 30 x 30 = 21.58	1024 x 0.024 / 30 x 30 = 24.58
30 x 5	0.25米ドル/1万オブジェクト x 10万	419.4304 x 0.024 / 30 x 30 + 604.5696 x 0.08 / 30 x 30 = 20.95	1024 x 0.024 / 30 x 30 = 24.58
30 x 6	0.25米ドル/1万オブジェクト x 10万	335.54432 x 0.024 / 30 x 30 + 688.45568 x 0.08 / 30 x 30 = 20.45	1024 x 0.024 / 30 x 30 = 24.58

このように、保存期間が長くなるにつれて、30日ごとに少額の監視料金を支払うだけで、それ以外のコストが明らかに低下することがわかります。

INTELLIGENT_TIERINGストレージはどのような種類のファイルに適しますか。

INTELLIGENT_TIERINGストレージはオーディオビデオ、ログなどの、平均ファイルサイズが比較的大きく、かつアクセスパターンが固定されていないファイルに適しています。平均ファイル容量が大きくなると、ファイル1GBにつき支払う必要がある監視料金は安くなります。業務上、アクセスパターンが比較的固定されている場合は、ライフサイクル設定によって、低頻度ストレージに移行する時間を指定しておけばよく、INTELLIGENT_TIERINGストレージを使用する必要はありません。

ファイルをINTELLIGENT_TIERINGストレージとして保存するにはどうすればよいですか。

次の2つの方法で、ファイルをINTELLIGENT_TIERINGストレージとして保存することができます。
増分ファイル：アップロードの際にストレージタイプをINTELLIGENT_TIERINGストレージに指定するだけで、ファイルをINTELLIGENT_TIERINGストレージとして保存することができます。
既存ファイル：インターフェースをCOPYすることで、ファイルのストレージタイプをINTELLIGENT_TIERINGストレージタイプに変更することができます。あるいはライフサイクル機能により、標準ストレージ、低頻度ストレージタイプをINTELLIGENT_TIERINGストレージタイプに移行することができます。

注意：

INTELLIGENT_TIERING系の64KB未満のファイルは、常に標準層に保存されます。そのため、コストダウンのために、必要に応じて、64KB未満のファイルを標準/低頻度/アーカイブ/ディープアーカイブなどのストレージタイプで直接アップロードすることをお勧めします。

INTELLIGENT_TIERINGストレージ設定を無効化するにはどうすればよいですか。

INTELLIGENT_TIERINGストレージは有効にすると**無効化することはできません**。ファイルをINTELLIGENT_TIERINGストレージとして保存する必要がない場合は、ファイルのアップロード時に、ファイル

ストレージタイプを標準ストレージ、低頻度ストレージ、アーカイブストレージ、ディープアーカイブストレージなどの、INTELLIGENT_TIERINGではないストレージタイプを指定してください。
INTELLIGENT_TIERING アーカイブ階層/ディープアーカイブ階層のルールは削除に対応しているため、対応するルールを削除することで、新しい INT オブジェクトがアーカイブ階層/ディープアーカイブ階層にトランジションことを回避できます。

INTELLIGENT_TIERING アーカイブとディープアーカイブ階層はどのように課金されますか。

ストレージ料金：同じ地域におけるアーカイブとディープアーカイブストレージに基づいて課金されます。
取得料金、取得リクエスト料金：アーカイブ階層高速取得料金を除き、その他の取得料金、取得リクエスト料金はかかりません。
リトリーブレプリカの料金：かかりません。

オブジェクトのアップロード
シンプルアップロード

最終更新日：：2024-06-26 10:57:13

シンプルアップロードとは、ユーザーがPUT Objectインターフェースを使用してオブジェクトをアップロードすることを言います。シンプルアップロード方式は単一のリクエストに適用でき、5GB以下の単一のオブジェクトのアップロードに用いられます。

5GBを超える単一のオブジェクトのアップロードには、次の方式を用いることができます。

コンソールを使用するアップロード：コンソールでは最大512GBまでの単一のオブジェクトのアップロードをサポートしています。詳細については、オブジェクトのアップロードのコンソールガイドドキュメントをご参照ください。

API/SDKを使用するマルチパートアップロード：最大48.82TB（50000GB）までの単一のオブジェクトのアップロードをサポートしています。詳細についてはマルチパートアップロードをご参照ください。

COSCMDツールを使用するアップロードでは最大40TBまでの単一のオブジェクトのアップロードをサポートしています。詳細については、COSCMDツールをご参照ください。

説明：

アップロードリクエストを送信する際、指定のフォルダまたはパスにアップロードしたい場合は、`/`を使用します。例えば、picture.pngをdocフォルダにアップロードする場合は、オブジェクトキーをdoc/picture.pngと設定します。

ユースケース

シンプルアップロード方式は5GB以下のオブジェクトをアップロードするケースに適用できます。

高帯域幅または脆弱なネットワーク環境においては、100MBを超えるような比較的大きなオブジェクト（5GB未満であっても）をアップロードする必要がある場合は、マルチパートアップロード方式を優先的に使用することをお勧めします。マルチパートアップロード方式では複数の分割ファイルの転送を同時実行することが可能なためです。高帯域幅環境ではリソースを有効利用でき、脆弱なネットワーク環境では、単一の分割ファイルのアップロード失敗が他のアップロード済みのファイルに影響を与えることがなく、簡単なリトライで失敗したファイルの再アップロードが可能なため、全体的なアップロード成功率を向上させることができます。モバイル端末での脆弱なネットワーク環境におけるアップロードに関しては、脆弱なネットワークにおけるマルチパートアップロード再開の実践をご参照ください。

利用方法

REST APIの使用

REST APIを直接使用してオブジェクトのシンプルアップロードリクエストを送信できます。 詳細については、PUT ObjectのAPIドキュメントをご参照ください。

SDKの使用

SDKのオブジェクトシンプルアップロードメソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK
C SDK
C++ SDK
.NET SDK
Go SDK
iOS SDK
Java SDK
JavaScript SDK
Node.js SDK
PHP SDK
Python SDK
Mini Program SDK

マルチパートアップロード

最終更新日：：2024-06-26 10:57:13

概要

マルチパートアップロードでは、オブジェクト全体を複数のパートに分割してから、これらのパートをCloud Object Storage (COS)にアップロードすることが可能です。アップロードの際、これらのパートには連続した順序に従って番号が付与され、各パートは単独でアップロードすることも、任意の順序でアップロードすることもできます。最終的にCOSはパートの番号に基づいてそのオブジェクトを再度組み立てます。いずれかのパートが転送に失敗した場合も、そのパートを再転送できるため、他のパートおよびコンテンツの完全性に影響を与えません。一般的に、不安定なネットワーク環境では、単一のオブジェクトが20MBを超えるとマルチパートアップロードを優先的に検討できます。高帯域幅環境では、100MBを超えるオブジェクトについてマルチパートアップロードを実施することができます。

マルチパートアップロードでは、大きなオブジェクトを最大で10000パートにまで分割することができます。分割するパートのサイズは一般的に1MB～5GBであり、最後の1パートは1MB未満とすることが可能です。

説明：

シンプルアップロード方式では最大5GBまでのファイルのアップロードのみサポートしていますが、マルチパートアップロード方式では5GBを超えるファイルのアップロードが可能です。

ユースケース

マルチパートアップロードは不安定なネットワーク、または高帯域幅環境下で大きなオブジェクトをアップロードする場合に適しています。

マルチパートアップロードには次のようなメリットがあります。

不安定なネットワーク環境では、パートを小さくすることで、ネットワークエラーによる中断の影響を低減し、オブジェクトのアップロード継続を実現できます。

高帯域幅環境下では、オブジェクトのパートのアップロードを同時実行することで帯域幅ネットワークを有効活用できます。順不同のアップロードが最終的なオブジェクトの組み立てに影響することもありません。

マルチパートアップロードを使用することで、単一の大きなオブジェクトのアップロードをいつでも一時停止し、再開することができます。終了操作を行った場合を除き、アップロードが未完了のオブジェクトは、いつでもアップロードを再開できます。

マルチパートアップロードはオブジェクト全体のサイズがわからない状態でオブジェクトをアップロードする場合にも適しています。先にアップロードを開始し、後でオブジェクトを組み立てることで完全なサイズを取得できます。

利用方法

REST APIの使用

REST APIを直接使用してマルチパートアップロードリクエストを送信できます。詳細については、次のAPIドキュメントをご参照ください。

Initiate Multipart Upload

Upload Part

Complete Multipart Upload

Abort Multipart Upload

SDKの使用

SDKのパート操作メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

Mini Program SDK

署名付きURLによるアップロード権限承認

最終更新日：：2024-06-26 10:57:13

ユースケース

デフォルトでは、バケットとオブジェクトはすべてプライベートなものです。第三者がオブジェクトをバケットにアップロードできるようにし、なおかつ相手にCAMアカウントまたは一時キーなどの方法を使用させたくない場合は、署名付きURLを使用して署名を第三者に渡し、一時的なアップロード操作を完了させることができます。有効な署名付きURLを受領した人は誰でもオブジェクトをアップロードできます。

URLを署名付きにする際、署名の中にオブジェクトキーを含める設定にすることで、指定されたオブジェクトキーへのアップロードのみを許可することができます。プログラム内で署名付きURLの有効期間を指定することで、タイムアウト後にそのURLが非権限者によって使用されないよう保証することもできます。

説明：

ユーザーには一時キーを使用して署名付きURLを生成し、一時権限承認方式によってさらにアップロード、ダウンロードなどの署名付きリクエストの安全性を向上させることをお勧めします。一時キーを申請する際は、最小権限の原則についてのガイドに従い、目的のバケットまたはオブジェクト以外のリソースが漏洩しないようにしてください。

やむを得ずパーマネントキーを使用して署名付きURLを生成する場合は、リスク回避のため、パーマネントキーの権限の範囲をアップロードまたはダウンロード操作のみに限定することをお勧めします。

利用方法

SDKの使用

SDKの署名付きURL生成メソッドを直接呼び出すことができます。下記の各言語のSDKドキュメントをご参照ください。

Android SDK
C SDK
C++ SDK
.NET SDK
Go SDK
iOS SDK
Java SDK
JavaScript SDK
Node.js SDK
PHP SDK

Python SDK

ミニプログラムSDK

オブジェクトのダウンロード

オブジェクトのシンプルダウンロード

最終更新日：：2024-06-26 10:57:13

ユースケース

リクエストを直接送信してCloud Object Storage（COS）内のオブジェクトをダウンロードすることができます。オブジェクトのダウンロードについては次の機能をサポートしています。

完全な単一のオブジェクトのダウンロード：GETリクエストを直接送信すると、完全なオブジェクトデータをダウンロードできます。

単一のオブジェクトの内容の一部をダウンロード：GETリクエストにRangeリクエストヘッダーを含めることで、ある特定の文字範囲を検索することができます。複数の範囲を検索することはできません。

オブジェクトのメタデータはHTTPレスポンスヘッダーとして、オブジェクトの内容と共に返されます。GETリクエストはURLパラメータを使用する方法で、応答の一部のメタデータ値を上書きします。

Content-Dispositionの応答値を例にとります。変更可能なレスポンスヘッダーには次のものがあります。

Content-Type

Content-Language

Expires

Cache-Control

Content-Disposition

Content-Encoding

利用方法

COSコンソールの使用

COSコンソールを使用してオブジェクトをダウンロードすることができます。詳細については、オブジェクトのダウンロードのコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用してオブジェクトのダウンロードリクエストを送信できます。 詳細については、GET ObjectのAPIドキュメントをご参照ください。

SDKの使用

SDKのオブジェクトダウンロードメソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

ミニプログラムSDK

署名付きURLによるダウンロード権限承認

最終更新日：：2024-06-26 10:57:13

ユースケース

デフォルトでは、バケットとオブジェクトはすべてプライベートなものです。第三者がオブジェクトをダウンロードできるようにし、なおかつ相手にCAMアカウントまたは一時キーなどの方法を使用させたくない場合は、署名付きURLを使用して署名を第三者に渡し、ダウンロード操作を完了させることができます。有効な署名付きURLを受領した人は誰でもオブジェクトをダウンロードできます。

URLを署名付きにする際、署名の中にオブジェクトキーを含める設定にすることで、指定されたオブジェクトのダウンロードのみを許可することができます。プログラム内で署名付きURLの有効期間を指定することで、タイムアウト後にそのURLが非権限者によって使用されないよう保証することもできます。

注意：

CDNドメイン名にアクセスするにはCDNの認証プロセスに従う必要があり、COSの署名は使用できないため、署名付きURLではCDNドメイン名の使用をサポートしていません。

ユーザーには一時キーを使用して署名付きURLを生成し、一時権限承認方式によってさらにアップロード、ダウンロードなどの署名付きリクエストの安全性を向上させることをお勧めします。一時キーを申請する際は、最小権限の原則についてのガイドに従い、目的のバケットまたはオブジェクト以外のリソースが漏洩しないようにしてください。

やむを得ずパーマネントキーを使用して署名付きURLを生成する場合は、リスク回避のため、パーマネントキーの権限の範囲をアップロードまたはダウンロード操作のみに限定することをお勧めします。また、生成した署名の有効期間を、今回のアップロードまたはダウンロード操作に必要な最短の期限までに設定し、指定した署名付きURLの有効期限が過ぎるとリクエストが中断するようにします。失敗したリクエストは新しい署名を申請後に再度実行する必要があります。中断からの再開はサポートしていません。

利用方法

SDKの使用

SDKの署名付きURL生成メソッドを直接呼び出すことができます。詳細については下記の各言語のSDKドキュメントをご参照ください。

Android SDK
C SDK
C++ SDK
.NET SDK
Flutter SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

React Native SDK

ミニプログラムSDK

オブジェクトキーのリストアップ

最終更新日：：2024-06-26 10:57:13

オブジェクトキー（ObjectKey）はオブジェクトのバケット内での固有識別子であり、わかりやすく言えば、ファイルパスであると理解することができます。例えば、オブジェクトキーがdoc/picture.jpgである場合、画像ファイルpicture.jpgがCloud Object Storage（COS）のdocパス（またはフォルダ）下に保存されていることを表します。
オブジェクトキーのリストアップとは、指定されたオブジェクトキーに基づいて特定のオブジェクトを検索することを指します。また、オブジェクトキーのプレフィックスに基づくオブジェクトの検索、すなわちオブジェクトキーの前半の一部（docなど）を指定して、同一のプレフィックスdocを持つすべてのオブジェクトを検索することも可能です。

ユースケース

Tencent Cloud COSではプレフィックスの順序に従ってオブジェクトキーをリストアップすることができます。また、オブジェクトキーの中で記号「/」を使用して、従来のファイルシステムに類似した階層構造を実現することも可能です。COSはまた、区切り文字による階層構造の選択と閲覧もサポートしています。
単一バケット内のすべてのオブジェクトキーをリストアップすることができます。プレフィックスのUTF-8バイナリー順序に従ってリストアップするか、または指定のプレフィックスを選択してオブジェクトキーのリストをフィルタリングすることが可能です。例えば、パラメータ `t` を入力すると `tencent` のオブジェクトがリストアップされますが、 `a` またはその他の文字をプレフィックスとするオブジェクトはスキップされます。
区切り文字 `/` を入力すると、この区切り文字に基づいてオブジェクトキーが再構成されます。プレフィックスと区切り文字を組み合わせることで、フォルダ検索に類似した機能を実現することができます。
Tencent Cloud COSでは1つのバケット内に無限の数のオブジェクトを格納できるため、オブジェクトキーリストが非常に大きくなる可能性があります。管理上の利便性のため、1つのオブジェクトリストアップインターフェースは最大1000個までのオブジェクトのリストを返し、同時にインジケータによって分割の有無の通知を返します。分割が存在する場合は、次のページにもオブジェクトリストが存在することを表します。インジケータおよび区切り文字によってオブジェクトキーリストアップリクエストを複数回送信することで、すべてのオブジェクトキーをリストアップすることや、必要な内容を照会することが可能になります。

利用方法

COSコンソールの使用

コンソールを使用してオブジェクトを検索することができます。詳細については、オブジェクトの検索のコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用してオブジェクトキーリストアップリクエストを送信できます。 詳細については、GET Bucket（List Objects）のAPIドキュメントをご参照ください。

SDKの使用

SDKのオブジェクトリスト照会メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

ミニプログラムSDK

オブジェクトのコピー

シンプルコピー

最終更新日：：2024-06-26 10:57:13

ユースケース

Cloud Object Storage（COS）に保存済みのオブジェクトのシンプルなコピー操作を行うことで、新しいオブジェクトレプリカを作成することができます。1回の操作で最大5GBのオブジェクトをコピーすることができます。オブジェクトが5GBを超える場合は、必ずマルチパートコピーインターフェースを使用してコピーを行わなければなりません。オブジェクトコピーには次の機能があります。

新しいオブジェクトレプリカを作成します。

オブジェクトをコピーして名前を変更し、元のオブジェクトを削除することでリネームを実現します。

オブジェクトのストレージタイプを変更します。コピー時に同一のソースおよびターゲットオブジェクトキーを選択することで、ストレージタイプを変更します。

異なるTencent Cloud COSリージョンにあるオブジェクトをコピーします。

オブジェクトのメタデータを変更します。コピー時に同一のソースおよびターゲットオブジェクトキーを選択し、その中のメタデータを変更します。

オブジェクトをコピーする際は、元のオブジェクトのメタデータがデフォルトで継承されますが、作成日は新しいオブジェクトの時間に基づいて計算されます。

説明：

CASタイプのオブジェクトのコピー＆ペーストはサポートされていません。

マルチAZ設定を有効にしているバケットでは、マルチAZストレージタイプをシングルAZストレージタイプにコピーすることはサポートされていません。

サブアカウントでオブジェクトをコピーするには、PutObject、GetObject、GetObjectACLという3つの権限が必要です。

利用方法

COSコンソールの使用

COSコンソールを使用してオブジェクトをコピーすることができます。詳細については、オブジェクトのコピーのコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用してオブジェクトコピーリクエストを送信できます。 詳細については、PUT Object - CopyのAPIドキュメントをご参照ください。

SDKの使用

SDKのオブジェクトコピー設定メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

ミニプログラムSDK

マルチパートコピー

最終更新日：：2024-06-26 10:57:13

ユースケース

5GBを超えるオブジェクトをコピーしたい場合は、マルチパートコピーのメソッドを選択して実現する必要があります。マルチパートアップロードのAPIを使用して新しいオブジェクトを作成し、Upload Part - Copyの機能を使用して、x-cos-copy-sourceヘッダーを付けてソースオブジェクトを指定します。これには次のフローが含まれます。

1. マルチパートアップロードするオブジェクトを初期化します。
2. ソースオブジェクトのデータをコピーします。x-cos-copy-source-rangeヘッダーを指定できます。1回にコピーできるデータは最大5GBまでです。
3. マルチパートアップロードが完了します。

説明：

Tencent Cloud COSが提供するSDKを使用すると、マルチパートコピー機能を簡単に利用できます。

利用方法

REST APIの使用

REST APIを直接使用してマルチパートコピーリクエストを送信できます。次のAPIドキュメントをご参照ください。

Initiate Multipart Upload

Upload Part - Copy

Complete Multipart Upload

Abort Multipart Upload

SDKの使用

SDKのパートコピーメソッドを直接呼び出すことができます。下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET(C#) SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK
Node.js SDK
PHP SDK
Python SDK
Mini Program SDK

オブジェクトを削除する

単一オブジェクトの削除

最終更新日：：2024-06-26 10:57:13

ユースケース

Tencent Cloud COSは単一または複数のオブジェクトの直接削除をサポートしています。単一のオブジェクトのみを削除したい場合は、オブジェクトキーを提供するだけで、APIリクエストを呼び出して削除することができます。

利用方法

COSコンソールの使用

COSコンソールを使用してオブジェクトを削除することができます。詳細については、オブジェクトの削除のコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用して単一のオブジェクトの削除リクエストを送信できます。 詳細については、DELETE ObjectのAPIドキュメントをご参照ください。

SDKの使用

SDKの単一オブジェクト削除メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK
C SDK
C++ SDK
.NET SDK
Go SDK
iOS SDK
Java SDK
JavaScript SDK
Node.js SDK
PHP SDK
Python SDK

ミニプログラムSDK

複数のオブジェクトの削除

最終更新日：：2024-06-26 10:57:13

ユースケース

Tencent Cloud COSは複数のオブジェクトの一括削除をサポートしています。コンソール、API、SDKなどの複数の方式でオブジェクトの一括削除を行うことができます。
デフォルトでは、削除タスクがすべて正常に完了した時点で、通常は空の内容が返されます。エラーが発生した場合はエラーメッセージが返されます。

注意：

1回のリクエストで最大1000個のオブジェクトを削除することができます。それ以上のオブジェクトを削除したい場合は、リストを分割してからそれぞれリクエストを送信してください。

利用方法

COSコンソールの使用

COSコンソールを使用して複数のオブジェクトの一括削除を行うことができます。詳細については、オブジェクトの削除のコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用して複数のオブジェクトの削除リクエストを送信できます。 詳細については、DELETE Multiple ObjectsのAPIドキュメントをご参照ください。

SDKの使用

SDKの複数オブジェクト削除メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

Mini Program SDK

データ管理

ライフサイクル管理

ライフサイクルの概要

最終更新日：：2024-06-26 10:57:13

概要

Cloud Object Storage（COS）はオブジェクトベースのライフサイクル設定をサポートしています。これはバケットに対して指定の記述言語を送信することで、ルールに該当するオブジェクトに、指定の条件下でいくつかのアクションを自動的に実行させることができるものです。

説明：

各バケットに追加できるライフサイクルルールは1000個までです。

ユースケース

ログの記録

ユーザーがCOSを使用してログデータを保存している場合は、ライフサイクルの設定によって、ログデータを30日後に自動的にアーカイブしたり、2年後に自動的に削除したりすることができます。

コールドデータとホットデータの階層分離

ホットデータはアップロード後、短時間のうちに大量にアクセスされ注目度が高まりますが、一定期間が過ぎると注目度が徐々に低下したり、リアルタイムでのアクセスが必要なくなったりします。ライフサイクルルールによって、30日前のデータを低頻度ストレージに切り替えることや、さらに60日前のデータをアーカイブストレージに切り替えることができます。このプロセスはデータの移行と呼ばれます。

アーカイブ管理

COSを使用してファイルアーカイブを管理する際、しばしば金融、医療などのコンプライアンス要件に基づき、ファイルのすべての過去バージョンを長期保存する必要が生じます。このような場合、ライフサイクル機能を使用することで、過去バージョンのファイルをアーカイブに移行するアクションを実行できます。

設定の要素

ライフサイクルルールを作成する際は、次の要素を設定する必要があります。

リソース

ライフサイクルの実行時にヒットするデータを指定します。カスタムライフサイクルの適用範囲および範囲内でカバーされるデータタイプを指定できます。ライフサイクルの実行時にはユーザーが指定する適用範囲をスキャンし、範囲内のユーザーが設定したデータタイプに対してアクションを実行します。このうち、適用範囲は以下のルールに従って指定することができます。

プレフィックスによる指定：ディレクトリのプレフィックスまたはファイル名のプレフィックスによるマッチングをサポートしています。

タグによる指定：オブジェクトタグによってデータをフィルタリングします。

設定可能なデータタイプには次のものがあります。

現在のバージョンのファイル：バケット内の最新バージョンのオブジェクト。

過去のバージョンのファイル：バージョン管理を有効化した後に保存した過去のバージョンのオブジェクト。バージョン管理に関するその他の情報については、バージョン管理の概要をご参照ください。

削除マーカー：「オブジェクトが削除済みであることを表すマーカー」であり、ライフサイクルでは過去のバージョンがすべて削除された後、このマーカーを自動的に除去します。削除マーカーに関するその他の説明については、削除マーカードキュメントをご参照ください。

フラグメントファイル：マルチパートアップロードタスクが完了していない場合に発生するフラグメント。

操作

オブジェクトにヒットした際に実行する操作を指定します。

データ移行：作成したオブジェクトを、指定の時間以降、低頻度ストレージ、INTELLIGENT_TIERINGストレージ、アーカイブストレージ、ディープアーカイブストレージタイプに移行します。

期限切れの削除：オブジェクトの有効期限を設定し、オブジェクトが期限切れとなった後に自動的に削除します。

時間条件

上記の操作をトリガーする時間条件を指定します。

日数に基づく計算：ルールに対応する動作を、オブジェクトの最終変更日から何日後に実行するかを明確に指定します。

利用説明

説明：

ライフサイクルの使用方法に関しては、ライフサイクルの設定をご参照ください。

ルール時刻の説明

ファイル変更時刻

ライフサイクルはオブジェクトの変更時刻に基づいてルールをトリガーし、実行することができます。ファイルに対して書き込み操作を行う、PUT Object、PUT Object - Copy、POST Object、Complete Multipart Uploadなどのインターフェースだけはオブジェクトの変更時刻を更新しますが、ライフサイクルによるオブジェクトの移行では変更時刻は更新されません。

実行日数の説明

ルールで設定する日数は24時間に準じます。24時間未満は1日として数えません。
例えば、1日の午後3時に、ファイルを変更から1日後に削除するというライフサイクルルールを設定したとします。その場合、ライフサイクルタスクは2日0時から、2日0時以前の時点で最終変更時刻から1日以上過ぎているファイルのスキャンを開始し、削除タスクを実行します。1日当日にアップロードしたファイルは、最終変更時刻から1日経っていないため削除されず、3日0時になってからスキャンと記録が行われ、削除が実行されます。

ルール日数の上限

ライフサイクルの設定は最長3650日までサポートしています。

有効時間

ライフサイクルは日次スキャンと実行という2つの操作によって有効になります。
スキャン：COSは現地時間（GMT+8）毎日0時にライフサイクルルールをプルします。スキャンは適用範囲のすべてのオブジェクトにヒットします。
実行：オブジェクトがルールで指定した日付に達したことがスキャンされた場合は、移行または削除操作を実行します。
例えば、あるユーザーが2023年1月20日にルールAを設定し、test.txtを変更時刻から10日後に削除するよう指定したとします。この場合、2023年1月21日0時から、毎日0時にtest.txtの変更時刻をスキャンします。このファイルの最終変更時刻が2023年1月15日の場合、2023年1月26日0時に行ったスキャンタスクによってこのファイルが削除条件を満たしていると判断されると、スキャン完了後に削除操作を実行します。

注意：

ルールスキャンおよび実行期間中にルールの状態を変更しないでください。変更すると元のルールが終了するため、移行または削除操作の正確な実行が保証されなくなります。

データ移行

単方向性の原則

データ移行は単方向性であり、標準ストレージ > 低頻度ストレージ > INTELLIGENT_TIERINGストレージ > アーカイブストレージ > ディープアーカイブストレージの順序のみ許容されます。階層をスキップしての移行（標準ストレージ > アーカイブストレージ）も可能ですが、逆方向への移行はできません。PUT Object - Copy（アーカイブストレージ/ディープアーカイブストレージタイプ以外）、またはPOST Object restore （アーカイブストレージ/ディープアーカイブストレージタイプにのみ適用）を呼び出すことでのみ、コールドストレージタイプのデータをホットストレージタイプに復元することができます。

最終的な整合性

同一グループのオブジェクトに複数のルールを設定していて、競合が生じている場合（期限切れ削除設定は含まない）、COSは**最もコールドなストレージタイプへの移行**ルールを優先的に実行します。
例えば、ルールAがファイル変更から90日後に**低頻度ストレージに移行**するよう設定し、ルールBがファイル変更から90日後に**アーカイブストレージに移行**するよう設定しており、かつ上記のルールがどちらも同一のオブジェクトtest.txtにヒットしている場合、ルールBが優先的に実行されます。

ルール	リソース	操作	時間条件	実行状況
ルールA	test.txt	低頻度ストレージに移行	ファイル変更時刻から90日	ルール競合により実行失敗
ルールB	test.txt	アーカイブストレージに移行	ファイル変更時刻から90日	実行成功

注意：

Tencent Cloud COSでは、同一グループのオブジェクトに対し、複数の競合条件を含むライフサイクルルールを設定しないよう強く注意喚起しています。競合がある状態で実行すると、料金が変わる可能性があります。
データの移行によってオブジェクトの元のアップロードまたは変更時刻が変更されることはありません。

期限切れの削除

処理ロジック

オブジェクトが指定された期限切れ削除のライフサイクルルールにマッチした場合、Tencent Cloudはオブジェクトを非同期の削除キューに追加します。実際に削除される時間は作成時間から一定程度遅延する場合があります。GETまたはHEAD Object操作によってオブジェクトの現在のステータスを取得することができます。

最終的な整合性

同一グループのオブジェクトに複数のルールを設定していて、競合が生じている場合、COSは最短の有効期限に準じて実行します。また、**期限切れ削除の実行エフェクトはストレージタイプの切り替えより強いものになります**。
例えば、ルールCがファイル変更から180日後に**低頻度ストレージに移行**するよう設定し、ルールDがファイル変更から180日後に**オブジェクトを削除**するよう設定しており、かつ上記のルールがどちらも同一のオブジェクトtest.txtにヒットしている場合、ルールDが優先的に実行されます。

ルール	リソース	操作	時間条件	実行状況
ルールC	test.txt	低頻度ストレージに移行	ファイル変更時刻から180日	ルール競合により実行失敗

ルールD	test.txt	オブジェクトを削除	ファイル変更時刻から180日	実行成功

注意：

Tencent Cloud COSでは、同一グループのオブジェクトに対し、複数の競合条件を含むライフサイクルルールを設定しないよう強く注意喚起しています。競合がある状態で実行すると、料金が変わる可能性があります。

コストの注意点

実行の説明

ライフサイクルが削除操作を実行する際には、バックエンド削除リクエストが発生します。移行操作を実行する際には、バックエンド削除リクエストと書き込みリクエストが発生します。上記の操作で発生するリクエストの回数は、リクエスト料金が発生した請求書に計上されます。例えば、標準ストレージタイプのファイル `test.txt` をライフサイクルで低頻度ストレージに移行すると、標準ストレージデータ削除と低頻度ストレージデータ書き込みの2回のリクエストが発生します。
ライフサイクル実行のエフェクトには予期しない状況や、バケットに大量の既存オブジェクトが含まれる状況は含まれません。他の状況が原因で完了していない場合は、GETまたはHEAD Object操作によってオブジェクトの現在のステータスを取得することができます。
現在Tencent Cloudはライフサイクルの実行エフェクトについて請求書の承諾をご提供していません。すなわち、オブジェクトの料金はライフサイクルの実行が完了した時点で変更されます。

時間に対するあいまいさ

低頻度ストレージとINTELLIGENT_TIERINGストレージタイプは30日以上、アーカイブストレージタイプは90日以上、ディープアーカイブストレージタイプは180日以上の保存期間がそれぞれ必要であることにご注意ください。データ移行または削除を実行する際に追加のストレージ料金は発生しません。Tencent Cloud COSは30/90/180日未満のライフサイクル設定をチェックしないため、正しい設定をお客様ご自身の必要性に応じて実行する必要があります。
例えば、ある低頻度ストレージのオブジェクトについて、保存期間が30日に達する前に移行が実行された場合、これによってその当日にオブジェクトのアーカイブストレージタイプ料金が発生すると同時に、低頻度ストレージの料金も30日目まで引き続きかかります。あるアーカイブストレージオブジェクトについて、保存期間が90日に達する前に削除が実行された場合、そのオブジェクトのアーカイブストレージタイプ料金は90日目まで継続して課金されます。データがディープアーカイブストレージに移行された場合も同様です。

サイズ制限

低頻度ストレージ、アーカイブストレージおよびディープアーカイブストレージにはそれぞれ、オブジェクトの最小占有容量の制限が設けられています。例えば、低頻度ストレージに64KB未満のファイルをアップロードした場合は、64KBとして計算されます。ユーザーコストを抑えるため、ライフサイクルは64KB未満のオブジェクトに対してはストレージタイプ切り替えの操作を実行しません。

説明：

ライフサイクルは64KB未満のオブジェクトに対しては切り替え操作を実行しません。

ライフサイクル設定の要素

最終更新日：：2024-06-26 10:57:13

基本構造

ライフサイクルの設定にはXML記述方法を使用します。1つまたは複数のライフサイクルルールを設定することが可能です。基本構造は次のとおりです。

```
<LifecycleConfiguration>
    <Rule>
        <ID>**your lifecycle name**</ID>
        <Status>Enabled</Status>
        <Filter>
            <And>
                <Prefix>projectA/</Prefix>
                <Tag>
                    <Key>key1</Key>
                    <Value>value1</Value>
                </Tag>
            </And>
        </Filter>
        **transition/expiration actions**
    </Rule>
    <Rule>
        ...
    </Rule>
</LifecycleConfiguration>
```

各ルールには次の内容が含まれます。

ID（オプション）：カスタマイズ可能な記述ルールの内容です。

Status：ルールの有効（Enabled）または無効（Disabled）のステータスを選択できます。

Filter：操作したいオブジェクトのフィルタリング条件の指定に用います。

アクション：上記の記述に該当するオブジェクトに対して実行する必要がある操作です。

時間：最終変更時間に基づいて日数（Days）を指定するか、またはある具体的な日付（Date）までに変更するオブジェクトを指定することができます。

ルールの記述

Filter要素

バケット内の全オブジェクトを対象とする

空のフィルタリング条件を指定すると、バケット内の全オブジェクトに適用されます。

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
        </Filter>
        <Status>Enabled</Status>
        **transition/expiration actions**
    </Rule>
</LifecycleConfiguration>
```

指定したオブジェクトキーのプレフィックスを対象とする

オブジェクトのプレフィックスを指定することで、プレフィックスの記述に該当する一部のオブジェクトグループに対してアクションを実行することができます。例えば、logs/をプレフィックスとするすべてのオブジェクトを設定する場合は次のようになります。

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>logs/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        **transition/expiration actions**
    </Rule>
</LifecycleConfiguration>
```

指定したオブジェクトタグを対象とする

あるオブジェクトキーに該当するkeyおよびvalueをフィルタリング条件として指定し、特定のタグのオブジェクトに対しアクションを実行します。例えば、タグのkey=typeおよびvalue=imageをフィルタリング条件としてオブジェクトを設定する場合は次のようになります。

```
<LifecycleConfiguration>
    <Rule>
```

```
          <Filter>
             <Tag>
                <Key>type</Key>
                <Value>image</Value>
             </Tag>
          </Filter>
          <Status>Enabled</Status>
          **transition/expiration actions**
      </Rule>
  </LifecycleConfiguration>
```

複数のフィルタリング条件を併用する

Tencent CloudのCloud Object Storage（COS）は、ANDロジックによる複数のフィルタリング条件の併用をサポートしています。例えば、logs/をプレフィックスとして設定すると同時に、オブジェクトタグのkey=typeおよびvalue=imageをフィルタリング条件としてオブジェクトを設定する場合は次のようになります。

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
            <And>
                <Prefix>logs/</Prefix>
                <Tag>
                      <Key>type</Key>
                      <Value>image</Value>
                </Tag>
            </And>
        </Filter>
```

```
          <Status>Enabled</Status>
          **transition/expiration actions**
      </Rule>
  </LifecycleConfiguration>
```

アクションの要素

ライフサイクルルールにおいて、条件に該当するオブジェクトのグループに対し、1つまたは複数のアクションを実行することができます。

切り替えアクション

Transitionアクションを指定すると、オブジェクトをあるストレージタイプから別のストレージタイプに切り替えることができます。バケットでバージョン管理を有効にしている場合は、現在のバージョンに対してのみアクションが実行されます。最短のTransition設定時間は0日です。例えば、30日後にアーカイブストレージに移行するよう設定する場合は次のようになります。

```
<Transition>
    <StorageClass>ARCHIVE</StorageClass>
    <Days>30</Days>
</Transition>
```

期限切れの削除

Expirationアクションを指定すると、ルールに該当するオブジェクトに対し期限切れ後に削除するアクションを行うことができます。バケットのバージョン管理を有効にしていない場合は、オブジェクトが完全に削除されます。バケットのバージョン管理を有効にしている場合は、期限切れのオブジェクトに**DeleteMarkerを追加**し、それを

現在のバージョンとして設定します。例えば、30日後にオブジェクトを削除するよう設定する場合は次のようになります。

```
<Expiration>
    <Days>30</Days>
</Expiration>
```

未完了のマルチパートアップロード

注意：

同一のライフサイクルルール内で、指定したオブジェクトタグの設定とアップロードが完了していないファイルフラグメントのクリーンアップを同時に行うことはサポートしていません。

AbortIncompleteMultipartUploadアクションを指定すると、マルチパートアップロードの指定されたUploadIdタスクを一定期間保持した後に削除することを許可し、それについてはアップロード再開または検索可能な特性の提供も行われないようにすることができます。例えば、未完了のマルチパートアップロードタスクを7日後に消去するよう設定する場合は次のようになります。

```
<AbortIncompleteMultipartUpload>
   <DaysAfterInitiation>7</DaysAfterInitiation>
</AbortIncompleteMultipartUpload>
```

現在のバージョンではないオブジェクト

バージョン管理を有効にしているバケットでは、切り替えは最新バージョンに対してのみ実行され、期限切れ操作は削除マーカーの追加だけとなります。そのため、COSでは現在のバージョンではないオブジェクトに対して、次のような操作を提供しています。

NoncurrentVersionTransitionを指定すると、現在のバージョンではないオブジェクトを指定の時間に別のストレージタイプに切り替えることができます。例えば、過去のバージョンを30日後にアーカイブストレージに移行するよう設定する場合は次のようになります。

```
<NoncurrentVersionTransition>
```

```
    <StorageClass>ARCHIVE</StorageClass>
    <Days>30</Days>
</NoncurrentVersionTransition>
```

NoncurrentVersionExpirationを指定すると、現在のバージョンではないオブジェクトを指定の時間に期限切れとして削除することができます。例えば、過去のバージョンを30日後に削除するよう設定する場合は次のようになります。

```
<NoncurrentVersionExpiration>
    <Days>30</Days>
</NoncurrentVersionExpiration>
```

ExpiredObjectDeleteMarkerの指定は余分な削除マーカーを消去するために用います。入力可能な値はtrueまたはfalseです。このオプションの発効は、期限切れとなった過去のバージョンを消去する動作（NoncurrentVersionExpiration）によってトリガーされます。この機能を有効にすると、ライフサイクルによってオブジェクトの最後の過去バージョンが削除された時点で、余分な複数の削除マーカーが自動的に消去されます。具体的には次のようになります。期限切れとなった過去のバージョンを消去する動作（NoncurrentVersionExpiration）が発効した時点で、

余分な削除マーカーが1個しかない場合は、ExpiredObjectDeleteMarkerが有効かどうかにかかわらず、最後の削除マーカーが自動的に消去されます。

余分な削除マーカーが複数ある場合は、ExpiredObjectDeleteMarkerが有効かどうかをCOSがチェックします。その結果がtrueであれば、余分な複数の削除マーカーが自動的に消去されます。falseであれば、余分な複数の削除マーカーは消去されません。

```
<ExpiredObjectDeleteMarker>
    <Days>true|false</Days>
</ExpiredObjectDeleteMarker>
```

時間の要素

日数に基づく計算

Daysを使用する日数の計算は、 オブジェクトの最終変更時間に基づいて行われます。

例えば、あるオブジェクトを0日後にアーカイブストレージタイプに切り替えるよう設定し、オブジェクトを2018-01-01 23:55:00 GMT+8にアップロードした場合、そのオブジェクトは2018-01-02 00:00:00 GMT+8からストレージタイプ切り替えの処理キューに入り、遅くとも2018-01-02 23:59:59 GMT+8までに切り替えが完了します。
例えば、あるオブジェクトを1日後に期限切れとして削除するよう設定し、オブジェクトを2018-01-01 23:55:00 GMT+8にアップロードした場合、そのオブジェクトは2018-01-03 00:00:00 GMT+8から期限切れ削除の処理キューに入り、遅くとも2018-01-03 23:59:59 GMT+8までに削除が完了します。

指定の日付に基づく

Dateを使用して日付を指定すると、特定の日付になった時点で、フィルタリング条件に該当するすべてのオブジェクトに対しこのアクションが実行されます。現時点ではGMT+8タイムゾーンの、0時に設定するISO8601形式の時間のみサポートしています。
例えば、2018年1月1日の場合、2018-01-01T00:00:00+08:00と記述します。

ライフサイクルの設定

最終更新日：：2024-06-26 10:57:13

ユースケース

ライフサイクルルール設定を利用すると、ルールに該当するオブジェクトに対し、指定された条件下でいくつかの操作を自動的に実行することができます。例えば次のようなものがあります。

ストレージタイプの切り替え：作成したオブジェクトを、指定の時間が経過した後に低頻度ストレージ（STANDARD_IA）、INTELLIGENT_TIERINGストレージ（INTELLIGENT_TIERING）、アーカイブストレージタイプ（ARCHIVE）、ディープアーカイブストレージタイプ（DEEP_ARCHIVE）に切り替えます。

期限切れの削除：オブジェクトの有効期限を設定し、オブジェクトが期限切れとなった後に自動的に削除します。

詳細については、ライフサイクルの概要のドキュメントおよびライフサイクル設定の要素のドキュメントをご参照ください。

利用方法

COSコンソールの使用

COSコンソールを使用してライフサイクルを設定することができます。詳細については、ライフサイクルの設定のコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用して、バケット内のオブジェクトのライフサイクルの設定と管理を行うことができます。詳細については次のAPIドキュメントをご参照ください。

PUT Bucket lifecycle

GET Buket lifecycle

DELETE Bucket lifecycle

SDKの使用

SDKのライフサイクルメソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

- iOS SDK
- Java SDK
- JavaScript SDK
- Node.js SDK
- PHP SDK
- Python SDK
- Mini Program SDK

静的ウェブサイトのホスティング

最終更新日：：2024-06-26 10:57:13

基本概念

静的ウェブサイトとは静的コンテンツ（HTMLなど）またはクライアントサイドスクリプトを含むウェブサイトを指します。ユーザーはコンソール上で、カスタムドメイン名をバインド済みのバケットに対して静的ウェブサイト設定を行うことができます。一方、動的ウェブサイトのコンテンツにはPHP、JSP、ASP.NETなどのサーバーサイドスクリプトが含まれ、サーバー側に依存して処理する必要があります。Cloud Object Storage（COS）は静的ウェブサイトのホスティングをサポートしていますが、サーバーサイドスクリプトの作成はサポートしていません。動的ウェブサイトをデプロイしたい場合は、Cloud Virtual Machine（CVM）を使用してサーバー側のコードデプロイを行うことをお勧めします。

事例

ユーザーがexamplebucket-1250000000という名前のバケットを作成し、次のファイルをアップロードしたとします。

静的ウェブサイト

```
index.html
404.html
403.html
test.html
docs/a.html
images/
```

静的ウェブサイト

有効化前：次のようなデフォルトアクセスドメイン名を使用してバケットにアクセスすると、ダウンロードの表示がポップアップされ、`index.html` ファイルをローカルに保存することができます。

```
https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com/index.html
```

有効化後：次のようなアクセスノードを使用してバケットにアクセスすると、ブラウザで直接 `index.html` のページ内容を確認することができます。

```
https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com/index.html
```

強制HTTPS

有効化前：リクエストのソースがHTTPの場合、アクセスノードのURLはHTTPの暗号化されていない転送プロトコルを維持します。

```
http://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com
```

有効化後：リクエストのソースがHTTPとHTTPSのどちらであっても、アクセスノードは常にHTTPSの暗号化された転送プロトコルを維持します。

```
https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com
```

インデックスドキュメント

インデックスドキュメント（静的ウェブサイトのホームページ）は、ユーザーがウェブサイトのルートディレクトリまたは任意のサブディレクトリにリクエストを送信したときに返されるページで、通常このページは `index.html` と命名されます。

ユーザーがバケットのアクセスドメイン名（例： `https://examplebucketbucket-1250000000.cos-`

`website.ap-guangzhou.myqcloud.com`）を使用して静的ウェブサイトにアクセスし、なおかつ特定のページをリクエストしていない場合、Webサーバーからはトップページが返されます。
お客様のユーザーがアクセスするバケットに、ルートディレクトリを含む何らかのディレクトリが含まれる場合、URLアドレスが `/` で終わるものは、このディレクトリ下のインデックスドキュメントに優先的に自動マッチングされます。ルートレベルのURLの `/` はオプションです。次の任意のURLはインデックスドキュメントを返します。

```
http://www.examplebucket.com/
http://www.examplebucket.com
```

注意：

バケット内にフォルダを作成した場合、各フォルダレベルでインデックスドキュメントを追加する必要があります。

エラードキュメント

エラードキュメントを設定する前に次のページにアクセスした場合、ステータスコード404が返され、ページ上にデフォルトのエラーページメッセージが表示されます。

```
https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com/webpage.html
```

エラードキュメントを設定すると、次のページにアクセスした場合、同じようにステータスコード404が返されますが、ページ上には指定したエラーページメッセージが表示されます。

```
https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com/webpage.html
```

リダイレクトルール

説明：

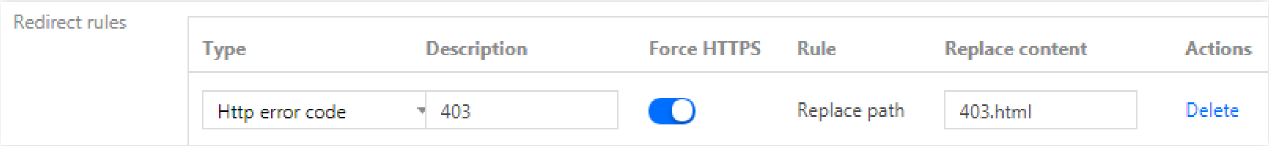
静的ウェブサイトのホスティングにリダイレクトルールを設定する場合、ドキュメントのパスの代わりにバケット内のオブジェクトパスが必要です。

エラーコードのリダイレクト設定

例えばwebpage.htmlというドキュメントに**プライベート読み取り/書き込み**のパブリックアクセス権限を設定した場合、ユーザーがこのファイルにアクセスすると、403エラーが返されます。

エラーコード403を403.htmlにリダイレクトすると、ブラウザは403.htmlの内容を返します。

403.htmlドキュメントを設定していない場合、ブラウザはエラードキュメントまたはデフォルトのエラーメッセージを返します。



Redirect rules

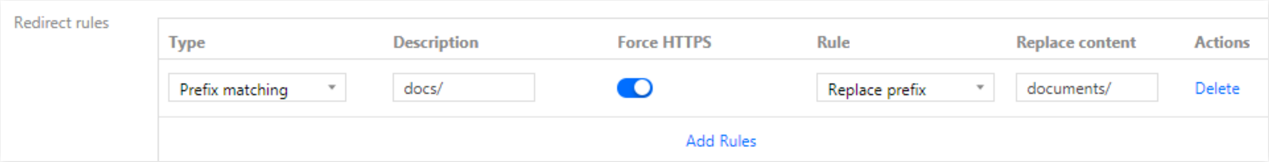
Type	Description	Force HTTPS	Rule	Replace content	Actions
Http error code	403	●	Replace path	403.html	Delete

プレフィックスマッチングの設定

注意：

プレフィックスマッチングはワイルドカードをサポートしていません。プレフィックスを index1/、index2/ という2つのフォルダとしてリダイレクトしたい場合、 `index*/` をマッチングルールとして使用することはできず、対応するマッチングルールをそれぞれ作成する必要があります。

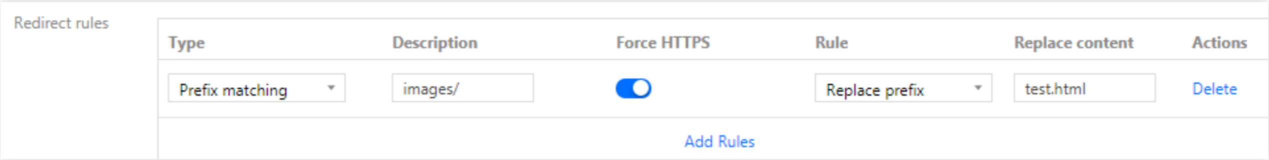
1. フォルダを `docs/` から `documents/` にリネームした場合、ユーザーが `docs/` フォルダにアクセスするとエラーが発生します。そのため、プレフィックスが `docs/` のリクエストを `documents/` にリダイレクトすることができます。



Redirect rules

Type	Description	Force HTTPS	Rule	Replace content	Actions
Prefix matching	docs/	●	Replace prefix	documents/	Delete

2. `images/` フォルダを削除（プレフィックス `images/` を持つすべてのオブジェクトを削除）した場合、リダイレクトルールを追加し、プレフィックスが `images/` のあらゆるオブジェクトのリクエストを、 `test.html` ページにリダイレクトすることができます。



Redirect rules

Type	Description	Force HTTPS	Rule	Replace content	Actions
Prefix matching	images/	●	Replace prefix	test.html	Delete

リスト機能の概要

最終更新日：：2024-06-26 10:57:13

リストとは

リストとは、ユーザーによるバケット内のオブジェクト管理を支援する機能の一種であり、COSの同期的List API操作を計画的に代替することができます。Cloud Object Storage（COS）はユーザーのリストタスク設定に基づき、毎日または毎週一定の時刻にユーザーがバケット内で指定したオブジェクトまたは同一のオブジェクトプレフィックスを持つオブジェクトのスキャンを行い、リストレポートを出力して、CSV形式のファイルをユーザーが指定したバケットに保存することができます。ファイルには保存されているオブジェクトおよびそれに対応するメタデータを出力し、ユーザーの設定情報に基づいて、ユーザーが必要とするオブジェクトのプロパティ情報を記録します。

リスト機能を使用すると、次の基本用途が実現できますが、用途はこれらのみに限定されません。

オブジェクトのコピーおよび暗号化状態を審査し、報告します。

業務のワークフローおよびビッグデータ作業を簡略化し、スピーディーに行えます。

注意：

ユーザーは1つのバケット内に複数のリストタスクを設定することができます。リストタスクは、オブジェクトの内容を直接読み取るのではなく、オブジェクトのメタデータなどのプロパティ情報のスキャンのみを行います。

リストのパラメータ

ユーザーがあるリストタスクを設定すると、COSは設定に応じて、ユーザーがバケット内で指定したオブジェクトを一定の時刻にスキャンし、リストレポートを出力します。リストレポートのファイル形式はCSVをサポートしています。現在、COSのリストレポートには次の情報を記録することができます。

リスト情報	説明
AppID	アカウントのID
Bucket	リストタスクを実行するバケット名
fileFormat	ファイル形式
listObjectCount	リストアップされたオブジェクト数です。料金はこの項目に基づいて課金されます。詳細については、管理機能料金のリスト機能料金の説明をご確認ください
listStorageSize	リストアップされたオブジェクトのサイズ
filterObjectCount	フィルタリングされたオブジェクト数

filterStorageSize	フィルタリングされたオブジェクトのサイズ
Key	バケット内のオブジェクトファイルの名称です。CSVファイル形式を使用する場合、オブジェクトファイル名にはURLエンコード形式が用いられているため、デコードして使用する必要があります
VersionId	オブジェクトのバージョンIDです。バケット上でバージョン管理を有効にすると、COSはバケットに追加されたオブジェクトにバージョン番号を割り当てます。リストがオブジェクトの現在のバージョンのみを対象としている場合は、このフィールドは含まれません
IsLatest	オブジェクトのバージョンが最新の場合はTrueに設定します。リストがオブジェクトの現在のバージョンのみを対象としている場合は、このフィールドは含まれません
IsDeleteMarker	オブジェクトが削除マーカーの場合はTrueに設定します。リストがオブジェクトの現在のバージョンのみを対象としている場合は、このフィールドは含まれません
Size	オブジェクトのサイズ（単位はバイト）
LastModifiedDate	オブジェクトの直近の変更日（日付の遅い方に準じる）
ETag	エンティティタグとはオブジェクトのハッシュです。ETagはオブジェクトの内容への変更のみを反映し、オブジェクトのメタデータへの変更は反映しません。ETagは、オブジェクトデータのMD5ダイジェストである場合も、そうではない場合もあります。どちらであるかは、オブジェクトの作成方法と暗号化方法によって決まります
StorageClass	オブジェクトの保存に用いるストレージクラスです。その他の情報に関しては、ストレージタイプをご参照ください
IsMultipartUploaded	オブジェクトをマルチパートアップロード形式でアップロードする場合は、Trueに設定します。その他の情報に関しては、マルチパートアップロードをご参照ください
Replicationstatus	オブジェクトのコピーにおいてソースファイルとレプリカファイルの状態をマークするために使用します。ソースファイルのマーカーはPENDING（コピー保留中）、COMPLETED（コピー完了）、FAILED（コピー失敗）、レプリカファイルのマーカーはREPLICA（コピー完了、レプリカファイル生成済み）となります。その他の情報に関しては、コピーアクションの説明をご参照ください
Tag	オブジェクトのタグ

リストの設定方法

リストを設定する前に、2つの概念をご理解いただく必要があります。

ソースバケット：リスト機能をアクティブ化したいバケットです。

リストにリストアップされるオブジェクトが含まれます。

リストの設定が含まれます。

ターゲットバケット：リストを保存するバケットです。

リストファイルが含まれます。

リストファイルの位置を記述したManifestファイルが含まれます。

リスト設定の手順は主に次のように分けられます。

ソースバケット内の分析対象のオブジェクト情報を指定する

どのオブジェクト情報を分析したいかをCOSに伝える必要があります。そのため、リスト機能を設定する際に、ソースバケットで次の情報を設定する必要があります。

選択するオブジェクトのバージョン：すべてのオブジェクトのバージョンをリストアップするか、または現在のバージョンのみをリストアップするかを選択します。すべてのオブジェクトのバージョンを選択した場合、COSは同名のオブジェクトのすべての過去バージョンをリストレポートに含めます。現在のバージョンのみを選択した場合、COSは最新バージョンのオブジェクトのみを記録します。

分析したいオブジェクトのプロパティの設定：オブジェクトのプロパティのうち、どの情報をリストレポートに記録させたいかをCOSに伝える必要があります。現在サポートしているオブジェクトのプロパティには、アカウントID、ソースバケット名、オブジェクトファイル名、オブジェクトバージョンID、最新バージョンかどうか、削除マーカーかどうか、オブジェクトサイズ、オブジェクトの最終変更日、ETag、オブジェクトのストレージクラス、地域間コピーマーカー、マルチパートアップロードファイルに該当するかどうかが含まれます。

リストレポートのストレージ情報の設定

どのような頻度でリストレポートをエクスポートするか、リストレポートをどのバケットに保存するかをCOSに伝えるとともに、リストレポートを暗号化するかどうかを決定する必要があります。設定する必要がある情報は次のとおりです。

リストのエクスポート頻度の選択：毎日または毎週から選択できます。この設定によって、どの頻度でリスト機能を実行するかをCOSに伝えることができます。

リストの暗号化の選択：暗号化なしとSSE-COS暗号化から選択できます。SSE-COS暗号化を選択した場合、生成されるリストレポートは暗号化されます。

リストの出力位置の設定：リストレポートを保存したいバケットを設定する必要があります。

注意：

ターゲットバケットは必ずソースバケットと同一のリージョンになければなりません。これらは同一のバケットとすることもできます。

利用方法

コンソール上でのリスト設定

コンソール上でリスト機能の設定を行う方法について詳しくお知りになりたい場合は、リスト機能のアクティブ化コンソールドキュメントをご参照ください。

APIによるリスト設定

APIを使用して、指定のバケットでリスト機能を有効化する場合は、次の手順をご参照ください。

1. COSロールを作成します。
2. COSロールに権限をバインドします。
3. リスト機能を有効化します。

1. COSロールの作成

COSロールを作成します。具体的なインターフェースの情報については、CreateRoleをご参照ください。

このうち、roleNameは必ずCOS_QcsRoleとします。

policyDocumentは次のとおりです。

```
    }]
}
```

```
{
    "version": "2.0",
    "statement": [{
        "action": "name/sts:AssumeRole",
        "effect": "allow",
        "principal":{
            "service": "cos.cloud.tencent.com"
        }
    }]
}
```

2. COSロールへの権限のバインド

ロール権限に権限をバインドします。具体的なインターフェースの情報については、AttachRolePolicyをご参照ください。
このうち、policyNameはQcloudCOSFullAccessとします。roleNameは手順1のCOS_QcsRoleとします。roleNameの作成時に返されたroleIDを使用することもできます。

3. リスト機能の有効化

インターフェースを呼び出してリスト機能を有効化します。具体的なインターフェースの情報については、PUT Bucket inventoryをご参照ください。リストファイルを保存するターゲットバケットはソースバケットと同一のリージョンにあることが必要です。

リストレポートのストレージパス

リストレポートおよび関連するManifest関連ファイルはターゲットバケット内に発行されます。このうちリストレポートは次のパスに発行されます。

```
destination-prefix/appid/source-bucket/config-ID/
```

Manifest関連ファイルはターゲットバケットの次のパスに発行されます。

```
destination-prefix/appid/source-bucket/config-ID/YYYYMMDD/manifest.json
destination-prefix/appid/source-bucket/config-ID/YYYYMMDD/manifest.checksum
```

パスの表す意味は次のとおりです。

desitination-prefix：ユーザーがリストを設定する際に設定する「ターゲットプレフィックス」です。ターゲットバケット内のパブリックな位置にあるすべてのリストレポートのグループ化に用いることができます。

source-bucket：リストレポートに対応するソースバケット名です。このフォルダを追加するのは、複数のソースバケットがそれぞれのリストレポートを同一のターゲットバケットに送信する際に競合が発生しないようにするためです。

config-ID：ユーザーがリストを設定する際に設定する「リスト名」です。同一のソースバケットに複数のリストレポートを設定し、それらを同一のターゲットバケットに送信する際、config-IDを使用して異なるリストレポートを区別することができます。

YYYYMMDD：タイムスタンプです。リストレポートの生成時にバケットスキャンを開始した時間および日付が含まれます。

manifest.json：Manifestファイルを指します。

manifest.checksum：manifest.jsonファイル内容のMD5です。

このうち、Manifest関連ファイルにはmanifest.jsonとmanifest.checksumの2つのファイルが含まれます。

説明：

Mainfestファイルに関する説明は次のとおりです。

manifest.jsonとmanifest.chenksumはどちらもManifestファイルです。manifest.jsonはリストレポートの位置を記述するもので、manifest.checksumはmanifest.jsonのファイル内容のMD5です。新しいリストレポートが発行される際、それには毎回新しいManifestファイルのセットが付属します。

manifest.jsonに含まれる各Manifestにはすべて、リストに関連するメタデータおよびその他の基本情報が記載されています。これらの情報には次が含まれます。

ソースバケット名。

ターゲットバケット名。

リストのバージョン。

タイムスタンプ。リストレポートの生成時にバケットのスキャンを開始した日付および時間が含まれます。

リストファイルの形式とアーキテクチャ。

ターゲットバケット内のリストレポートのオブジェクトキー、サイズ、md5Checksum。

CSV形式のリストのmanifest.jsonファイルにおけるManifestの例は次のとおりです。

```
{
 "sourceAppid": "1250000000",
 "sourceBucket": "example-source-bucket",
 "destinationAppid": "1250000000",
 "destinationBucket": "example-inventory-destination-bucket",
 "fileFormat": "CSV",
 "listObjectCount": "13",
 "listStorageSize": "7212835",
 "filterObjectCount": "13",
 "filterStorageSize": "7212835",
 "fileSchema": "Appid, Bucket, Key, Size, LastModifiedDate, ETag, StorageClass, IsM
 "files": [
  {
   "key": "cos_bucket_inventory/1250000000/examplebucket/inventory01/04d73d9debc73d
   "size": "502",
   "md5Checksum": "7d40288a09c25b302ad6cb5fced54f35"
  }
 ]
}
```

リストの整合性

COSのリストレポートでは、新たなオブジェクトと上書きしたPUTの最終的な整合性、ならびにDELETEの最終的な整合性が示されます。このため、リストレポートには最近追加または削除されたオブジェクトが含まれない可能性があります。例えば、COSがユーザーの設定したリストタスクのプロセスを実行中に、ユーザーがオブジェクトのアップロードまたは削除操作を実行した場合、これらの操作はリストレポートに反映されない可能性があります。

オブジェクトの操作を実行する前にオブジェクトの状態を検証したい場合は、HEAD ObjectAPIを使用してオブジェクトメタデータを検索するか、またはCOSコンソールでオブジェクトのプロパティをチェックすることをお勧めします。

バケットタグの概要

最終更新日：：2024-06-26 10:57:13

概要

バケットタグは1つのキーバリューペア（key = value）であり、タグのキー（key）とタグの値（value）を「=」でつないだ構成となっています（例：group = IT）。バケットを管理する標識として、ユーザーがバケットのグループ化管理を行う際に便利に使用できます。指定したバケットに対しタグの設定、照会および削除操作を行うことができます。

仕様と制限

タグキーの制限

qcs:、project、プロジェクトなどで始まるタグキーはシステム予約済みタグキーです。システム予約済みタグキーの作成は禁止されています。

UTF-8形式で表される文字、スペース、数字および特殊文字 `+ - = ._ : / @` をサポートしています。

タグキーの長さは0～127文字とします（UTF-8形式を使用）。

タグキーはアルファベットの大文字と小文字を区別します。

タグ値の制限

UTF-8形式で表される文字、スペース、数字および特殊文字 `+ - = ._ : / @` をサポートしています。

タグ値の長さは0～255文字とします（UTF-8形式を使用）。

タグ値はアルファベットの大文字と小文字を区別します。

タグ数の制限

バケット次元：バケットタグの数は1つのリソースにつき最大50個までです。

タグ次元：

1ユーザーのkey数は最大で1000個までです 。

1つのkeyにつき、valueは最大で1000個までです。

同一のバケットに同一のkeyを複数設定することはできません。

利用方法

コンソール、API方式でバケットタグを設定できます。

COSコンソールの使用

COSコンソールを使用してバケットタグを設定したい場合は、バケットタグの設定のコンソールガイドドキュメントをご参照ください。

REST APIの使用

次のAPIによってバケットタグを直接管理できます。

PUT Bucket tagging

GET Bucket tagging

DELETE Bucket tagging

オブジェクトタグの概要

最終更新日：：2024-06-26 10:57:13

機能の説明

オブジェクトタグ機能は、オブジェクトにキーバリューペア形式の標識を追加することで、ユーザーがバケット内のオブジェクトをグループ化管理しやすくするものです。オブジェクトタグはタグのキー（`tagKey`）とタグの値（`tagValue`）を`=`でつないだ構成となっています（例：`group = IT`）。ユーザーは指定したオブジェクトに対しタグの設定、照会、削除操作を行うことができます。

注意：

オブジェクトタグ機能は課金項目です。価格の詳細については、製品価格のドキュメントをご参照ください。

使用方法

COSコンソールの使用

COS(Cloud Object Storage)コンソールから、オブジェクトタグを管理することができます。詳細については、オブジェクトタグの設定をご参照ください。

REST APIの使用

次のAPIによってオブジェクトタグを管理できます。

PUT Object tagging

GET Object tagging

DELETE Object tagging

仕様と制限

タグキーの制限

UTF-8形式で表される文字、スペース、数字[0-9]および特殊文字`+ - = ._ : / @`をサポートしています。

タグキーの長さの範囲は1～127文字とします（UTF-8形式を使用）。

タグキーはアルファベットの大文字と小文字を区別します。

タグ値の制限

UTF-8形式で表される文字、スペース、数字[0-9]および特殊文字`+ - = ._ : / @`をサポートしています。

タグ値の長さの範囲は1～255文字とします（UTF-8形式を使用）。

タグ値はアルファベットの大文字と小文字を区別します。

タグ数の制限

オブジェクト次元：オブジェクトタグの数は1つのオブジェクトにつき最大10個までです。
タグ次元：制限がありません。

イベント通知

最終更新日：：2024-06-26 10:57:13

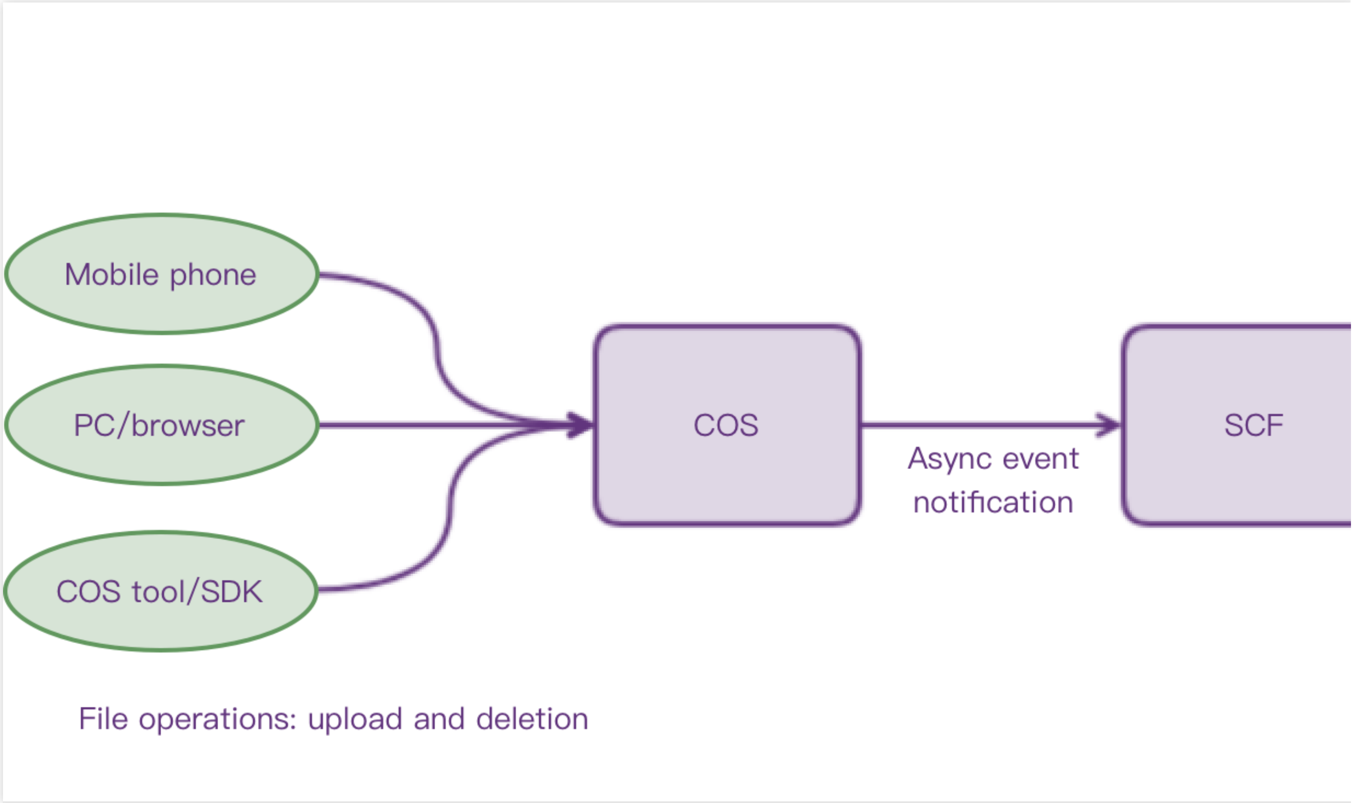
概要

COSのリソースに変動（新規ファイルのアップロード、ファイルの削除など）があった場合、ユーザーは速やかにメッセージ通知を受信できます。イベント通知はSCF（Serverless Cloud Function）と組み合わせることで、より豊富なユースケースを実現できます。

製品間連携：例えば、新たなファイルがCOSにアップロードされると、CDNキャッシュの自動更新を行います。新たなファイルがCOSにアップロードされると、データベースを自動更新します。

システム統合：COS上のファイルに変更（新規作成、削除、上書き）が発生した場合、自身のサービスインターフェースを自動的に呼び出します。UGC（User Generated Content）のケースでは、イベント通知機能をベースにしてモバイル端末とサーバーの連携を実現することができます。

データ処理：COS上のファイルに対し、自動解凍、AI認識などの自動処理を行います。


Mobile phone
PC/browser
COS tool/SDK
COS
Async event notification
SCF
File operations: upload and deletion

COSイベント通知には次のような特徴があります。

非同期処理：通知の送信は正常なCOS操作に影響を与えません。

通知先：通知は同リージョンのSCF関数への送信のみサポートされます。

現在は次のCOSイベントをサポートしています。

イベントタイプ	説明
cos:ObjectCreated:*	次に挙げるアップロードイベントはすべてSCFをトリガーします
cos:ObjectCreated:Put	PUT Objectインターフェースを使用してファイルを作成した際にSCFをトリガーします
cos:ObjectCreated:Post	POST Objectインターフェースを使用してファイルを作成した際にSCFをトリガーします
cos:ObjectCreated:Copy	PUT Object - Copyインターフェースを使用してファイルを作成した際にSCFをトリガーします
cos:ObjectCreated:CompleteMultipartUpload	Complete Multipart Uploadインターフェースを使用してファイルを作成した際にSCFをトリガーします
cos:ObjectCreated:Origin	イメージのback-to-originが発生した際にSCFをトリガーします
cos:ObjectCreated:Replication	地域間コピーによってオブジェクトを作成した際にSCFをトリガーします
cos:ObjectRemove:*	次に挙げる削除イベントはすべてSCFをトリガーします
cos:ObjectRemove:Delete	バージョン管理を有効にしていないバケットで、DELETE Objectインターフェースを使用してオブジェクトを削除するか、またはversionidを使用して指定のバージョンのオブジェクトを削除した際にSCFをトリガーします
cos:ObjectRemove:DeleteMarkerCreated	バージョン管理を有効化または一時停止しているバケットで、DELETE Objectインターフェースを使用してオブジェクトを削除した際にSCFをトリガーします
cos:ObjectRestore:Post	アーカイブ復元タスクを作成した際にSCFをトリガーします
cos:ObjectRestore:Completed	アーカイブ復元タスクを完了した際にSCFをトリガーします

COSイベント通知の利用方法

COSイベント通知の利用には次の手順が含まれます。

1. SCF関数の作成

SCFコンソールまたはCLIによって関数を作成することができます。関数作成の過程では、実行環境の選択（その後の関数作成に使用する言語に基づいて選択します）、関数コードの送信（オンラインでの編集またはローカルでのコードパッケージのアップロードをサポートしています）が必要です。

SCFのプリセットテンプレートによる簡略化した作成フローを使用することもできます。詳細については、関数の作成をご参照ください。関数の書き方はプログラミング言語によって異なります。詳細については、SCFのドキュメントをご参照ください。

2. 関数のテスト

関数の作成完了後に、テストテンプレート機能を使用して一次テストを行うことができます。テストテンプレートはCOSイベントを再現し、関数の実行をトリガーします。詳細については、関数のテストをご参照ください。

3. トリガーの追加

一次テストの完了後、COSトリガーを作成してSCF関数をバケットにバインドすることができます。トリガーの追加はコンソールまたはコマンドラインによって行うことができます。詳細については、トリガーの作成のドキュメントをご参照ください。

4. 実際の検証

上記の手順が完了すると、COS内のバケットの操作および、フロー全体が正常かどうかの検証が行えるようになります。例えば、コンソール、COS Browserなどのツールによってファイルをアップロード、削除できるほか、**SCFコンソール** > **関数の詳細** > **対応する関数名** > **実行ログ**に進み、正常に動作しているかを検証することができます。

SCF COSトリガーに関するその他の詳細については、 COSトリガーのドキュメントをご参照ください。

データ検索

Selectの概要

最終更新日：：2024-06-26 10:57:13

COS Select機能は、COS上に保存されたオブジェクトを構造化問い合わせ言語（SQL）によってフィルタリングすることで、オブジェクトを検索してユーザーが必要とするデータを取得するために役立てるものです。COS Select機能によってオブジェクトデータをフィルタリングすることで、COSの転送データ量を削減することができ、そのデータの検索に必要なコストと遅延を減少させることができます。

COS Select機能は現在CSV、JSON、Parquet形式によるストレージオブジェクトの検索、GZIPまたはBZIP2で圧縮されたオブジェクト（CSV、JSON形式のオブジェクトのみ）の検索をサポートしています。また、COS Select機能は結果の形式をCSVまたはJSONに指定することができるほか、結果の記録の区切り形式を決定することができます。

リクエストの中でSQL式をCOSに伝達することができます。COS Selectは現時点では一部のSQL式のみサポートしています。COS SelectがサポートするSQL式についての詳しい情報は、SQL関数のドキュメントをご参照ください。

COSコンソール、API、SDK、COSCMDなどの方法でSQL照会を実行することができます。COSコンソールを使用したファイル検索には一定の制限があり、検索できるファイルは最大128Mまでで、返されるデータ量は40MBに限られる点に注意が必要です。より多くのデータを検索したい場合は、その他の方法を使用して行ってください。

説明：

COS Selectがサポートするデータタイプおよび現在の予約フィールドについて詳しくお知りになりたい場合は、データタイプおよび予約フィールドをご参照ください。

現在検索機能は中国大陸のパブリッククラウドリージョンのみサポートしており、他のリージョンでは現時点ではこの機能をサポートしていません。

使用制限

COS Selectを使用する際は次の制限があります。

照会するオブジェクトの `cos:GetObject` 権限を有している必要があります。ルートアカウントはデフォルトでこの権限を有しています。

標準ストレージタイプのオブジェクトの検索のみサポートします。

SQL式の長さは最大256KBまでです。

検索結果の中の1つのレコードの長さは最大1MBまでです。

COS Select機能は現在次のSQL句をサポートしています。

SELECTステートメント

FROM句

WHERE句

LIMIT句

説明：

SQL句に関する詳細情報については、Selectコマンドをご参照ください。

COS Selectが現在サポートしている関数は次のとおりです。

集計関数：AVG関数、COUNT関数、MAX関数、MIN関数、SUM関数など。

条件関数：COALESCE関数、NULLIF関数など。

変換関数：CAST関数など。

日付関数：DATE_ADD関数、DATE_DIFF関数、EXTRACT関数、TO_STRING関数、TO_TIMESTAMP関数、UTCNOW関数など。

文字列関数：CHAR_LENGTH関数、CHARACTER_LENGTH関数、LOWER関数、SUBSTRING関数、TRIM関数、UPPER関数など。

説明：

SQL関数に関する詳細情報については、SQL関数をご参照ください。

COS Selectは現在次の演算子をサポートしています。

論理演算子： `AND, NOT, OR`

比較演算子： `<, >, <=, >=, =, <>, !=, BETWEEN, IN`

パターンマッチ演算子： `LIKE`

算術演算子： `+, -, *, %`

説明：

演算子に関する詳細情報については、演算子をご参照ください。

検索リクエストの発行

コンソール、API、SDKなどの複数の方式を使用して検索リクエストを発行することができます。

コンソールを使用する場合は、データ検索のドキュメントを参照して操作を行うことができます。

APIを使用する場合は、SELECT Object ContentAPIドキュメントをご参照ください。

SDKを使用する場合は、SDKの概要で、必要なSDKインターフェースを選択することができます。

よくあるご質問

照会の実行を試した際に問題が発生した場合、COS Selectはエラーコードおよび関連のエラーメッセージを返します。

Selectコマンド

最終更新日：：2024-06-26 10:57:13

概要

COS Select機能はSELECT SQL照会コマンドのみをサポートし、必要なデータの一部を検索できるようにすることで、転送データ量を削減することができます。こうすることでコストが削減でき、リクエストの遅延も低減できます。SELECT照会でサポートする標準句は次のとおりです。

SELECTステートメント

WHERE句

LIMIT句

注意：

COS Selectは現時点では句の照会またはjoinsをサポートしていません。

SELECTステートメント

SELECTステートメントはCOSオブジェクトの中からお客様が見たいデータを検索できるようにするもので、列の名称、関数または式などの次元で照会を行い、リスト形式で照会結果が返されます。SELECTステートメントの呼び出し形式は次のとおりです。

```
SELECT *
SELECT projection [ AS column_alias | column_alias ] [, ...]
```

最初のSELECTステートメントには `*` （アスタリスク）が付いており、COSオブジェクトのすべての列が返されます。2番目のSELECTステートメントはユーザー定義の出力スカラー式、**projection**を各列に使用した、カスタム名の出力リストを作成します。

WHERE句

WHERE句は次の構文を使用します。

```
WHERE condition
```

WHERE句は**condition**によってフィルタリングされます。**condition**はブール型の結果を返すことができる式であり、戻り値がTRUEの行のみ結果として出力されます。

LIMIT句

LIMIT句は次の構文を使用します。

```
LIMIT number
```

LIMIT句は毎回の照会で返されるレコード数を制限します。**number**パラメータによってこの制限を指定することができます。

属性アクセス

SELECTおよびWHERE句は、次の任意の方法で照会するフィールドを選択することができます。ファイル形式がCSVかJSONかによって選択できます。

CSV

列番号： `_N` によって、N番目の列のデータの照会を指定できます。任意のCSVファイルの場合、列番号は1から順に増加します。1列目の番号が `_1` であれば、2列目の番号は `_2` となります。SELECTおよびWHERE句では、 `_N` または `alias._N` によって、照会を行いたい列をすべて正しい方法で指定できます。
列ヘッダー：照会対象のCSVファイルにヘッダーが含まれる場合、SELECTおよびWHERE句ではこのヘッダーによって照会を行いたい列を指定することができます。SQLステートメントでは、SELECTおよびWHERE句で `alias.column_name` または `column_name` の方法によって指定できます。

JSON

ドキュメント（Document）： `alias.name` の方法によってJSONドキュメントにアクセスできます。ネスト配列には `alias.name1.name2.name3` の方法でアクセスできます。
リスト（List）：インデックスによってリスト内の要素にアクセスできます。インデックス番号は0から開始し、オペレーター `[]` を使用します。例えば、 `alias[1]` によってJSONリスト内の2番目の要素にアクセスできます。ネスト配列にアクセスしたい場合は、 `alias.name1.name2[1].name3` のような方法によってアクセスすることもできます。
事例
この事例のデータサンプルを次に示します。

```
}
```

```
{
    "name": "Leon",
    "org": "Tencent",
    "projects":
        [
         {"project_name":"project1", "completed":true},
         {"project_name":"project2", "completed":false}
        ]
}
```

事例1：データサンプルにおいてnameのSQLステートメントを照会した結果：

```
Select s.name from COSObject s
```

```
{"name":"Leon"}
```

事例2：データサンプルにおいてproject_nameのSQLステートメントを照会した結果：

```
Select s.projects[0].project_name from COSObject s
```

```
{"project_name":"project1"}
```

ヘッダーおよび属性名の大文字と小文字の区別

二重引用符を使用して、CSVファイルのヘッダーおよびJSONファイルの属性名が大文字と小文字を区別するかどうかを表示できます。二重引用符を追加しない場合、ヘッダー/属性名の大文字と小文字は区別されないことを表します。設定が不明確な場合、COS Selectがエラーをスローする場合があります。

事例1：照会したヘッダー/属性名の中に「NAME」のオブジェクトがある場合
次のSQLの例では二重引用符を使用していないため、大文字と小文字は区別されないことを表します。テーブルの中にこのヘッダーがあるため、最終的に正しく値が返されます。

```
SELECT s.name from COSObject s
```

次のSQLの例では二重引用符を使用しているため、大文字と小文字が区別されることを表します。テーブルの中に実際にこのヘッダーが含まれないため、最終的にエラーコード400の `SQLParsingError` が返されます。

```
SELECT s."name" from COSObject s
```

事例2：照会したヘッダー/属性名の中に"NAME"と"name"のオブジェクトがある場合

次のSQLの例では二重引用符を使用していないため、大文字と小文字は区別されないことを表します。テーブルの中に「NAME」と「name」という2つのヘッダーがあるため、照会コマンドの設定が不明確であり、エラーAmbiguousFieldNameがスローされます。

```
SELECT s.name from COSObject s
```

次のSQLの例では二重引用符を使用しているため、大文字と小文字は区別されることを表します。テーブルの中に「NAME」ヘッダーが存在するため、照会結果が正しく返されます。

```
SELECT s."NAME" from COSObject s
```

予約フィールドをユーザーカスタムフィールドとして使用する

COS SelectのSQL式には、関数名、データタイプ、オペレーターなどを含むいくつかの予約フィールドがあります。状況によっては、ユーザーがこれらの予約フィールドをCSVファイルの列ヘッダーまたはJSONファイルの属性名として使用する可能性がありますが、その場合、予約フィールドとの競合が発生する可能性があります。この

ような場合は、二重引用符を使用することで、カスタムフィールドを使用中であることを明らかにすることができます。これを行わない場合、COSは `400 parse error` を返します。

完全な予約フィールドリストをご覧になりたい場合は、予約フィールドをご参照ください。

事例：照会対象のオブジェクトのヘッダー/属性名に予約フィールド「CAST」が含まれる場合

次のSQLの例では二重引用符を使用し、CASTがユーザーカスタムフィールドであることを明らかにしていますので、照会結果が正しく返されます。

```
SELECT s."CAST" from COSObject s
```

次のSQLの例では、二重引用符を使用せず、CASTがユーザーカスタムフィールドであることを明らかにしていませんので、COSは予約フィールドとして処理し、 `400 parse error` が返されます。

```
SELECT s.CAST from COSObject s
```

スカラー式

SELECTステートメントとWHERE句では、SQLスカラー式（値を返す式）を使用することができます。現在COS Selectでは次の形式をサポートしています。

literal：SQLのテキスト。

column_reference：column_nameまたはalias.column_name。

unary_opexpression： SQLの単項演算子。

expressionbinary_opexpression：SQLの二項演算子。

func_name：呼び出された値関数の名称。

expression [NOT] BETWEEN **expression** AND **expression**

expression LIKE **expression** [ESCAPE **expression**]

SQL関数

最終更新日：：2024-06-26 10:57:13

集計関数

COS Selectは次の集計関数をサポートしています。

関数名	パラメータタイプ	リターンタイプ
AVG(expression)	INT、 FLOAT、 DECIMAL	入力パラメータが整数型の場合はDECIMALを返し、入力パラメータが浮動小数点型の場合はFLOATを返し、その他の場合の戻り値は入力パラメータと一致
COUNT	-	INT
MAX(expression)	INT、 DECIMAL	戻り値は入力パラメータと一致
MIN(expression)	INT、 DECIMAL	戻り値は入力パラメータと一致
SUM(expression)	INT、 FLOAT、 DOUBLE、 DECIMAL	入力パラメータが整数型の場合はINTを返し、入力パラメータが浮動小数点型の場合はFLOATを返し、その他の場合の戻り値は入力パラメータと一致

条件関数

COS Selectは次の条件関数をサポートしています。

COALESCE

COALESCE関数は入力されたパラメータを順序に従って判断し、空ではない最初のパラメータ値を返します。入力パラメータに空ではないパラメータが含まれない場合は、関数がnull値を返します。

構文

```
COALESCE ( expression, expression, ... )
```

説明：

expressionパラメータはINT、String、Floatタイプの数値、配列またはネスト関数を渡すことができます。

事例

```
COALESCE(1)                 -- 1
COALESCE(1, null)           -- 1
COALESCE(null, null, 1)     -- 1
COALESCE(missing, 1)        -- 1
COALESCE(null, 'string')    -- 'string'
COALESCE(null)              -- null
COALESCE(null, null)        -- null
COALESCE(missing)           -- null
COALESCE(missing, missing)  -- null
```

NULLIF

NULLIF関数は渡された2つのパラメータ間の差異を判断し、2つの入力パラメータが同一の値であればNULLを返し、異なっていれば最初の入力パラメータの値を返します。

構文

```
NULLIF ( expression1, expression2 )
```

説明：

expressionパラメータはINT、String、Floatタイプの数値、配列またはネスト関数を渡すことができます。

事例

```
NULLIF(1, 2)             -- 1
NULLIF(1, '1')           -- 1
NULLIF(1, NULL)          -- 1
NULLIF(1, 1)             -- null
NULLIF(1.0, 1)           -- null
NULLIF(missing, null)    -- null
NULLIF(missing, missing) -- null
NULLIF([1], [1])         -- null
NULLIF(NULL, 1)          -- null
NULLIF(null, null)        -- null
```

変換関数

COS Selectは次の変換関数をサポートしています。

CAST

CAST関数はある種類のインスタンスを別の種類のインスタンスに変換することができます。このインスタンスは数値でも、計算によってある確定した数値が出せる関数でも構いません。

構文

```
CAST ( expression AS data_type )
```

説明：

expressionパラメータは数値、配列、演算子または計算によって、ある確定した数値が出せるSQL関数とすることができます。

data_typeパラメータは例えばINTタイプのような、変換後のデータタイプです。現在COS Selectがサポートしているデータタイプについては、データタイプをご参照ください。

事例

```
CAST('2007-04-05T14:30Z' AS TIMESTAMP)
CAST(0.456 AS FLOAT)
```

日付関数

COS Selectは次の日付関数をサポートしています。

DATE_ADD

DATE_ADD関数は指定のタイムスタンプのある部分（年、月、日、時、分、秒）に指定された時間間隔を加え、新しいタイムスタンプを返すことができます。

構文

```
DATE_ADD( date_part, quantity, timestamp )
```

説明：

date_partパラメータはタイムスタンプの中の変更したい部分を指定します。年、月、日、時、分、秒とすることができます。

quantityパラメータは増分したい数値を表します。正の整数でなければなりません。

timestampパラメータは変更したいタイムスタンプです。

事例

```
DATE_ADD(year, 5, `2010-01-01T`)                -- 2015-01-01
DATE_ADD(month, 1, `2010T`)                     -- 2010-02T
DATE_ADD(month, 13, `2010T`)                    -- 2011-02T
DATE_ADD(hour, 1, `2017T`)                      -- 2017-01-01T01:00-00:00
DATE_ADD(hour, 1, `2017-01-02T03:04Z`)          -- 2017-01-02T04:04Z
DATE_ADD(minute, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:05:05.006Z
DATE_ADD(second, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:04:06.006Z
```

DATE_DIFF

DATE_DIFF関数は2つの有効なタイムスタンプを比較し、2つのタイムスタンプの差を返します。この差は指定された時間単位で表示することができます。 timestamp1のdate_part値がtimestamp2より大きい場合は、戻り値は正の値となります。逆の場合は負の値を返します。

構文

```
DATE_DIFF( date_part, timestamp1, timestamp2 )
```

説明：

date_partパラメータは2つのタイムスタンプを比較する時間単位を指定します。年、月、日、時、分、秒とすることができます。

timestamp1パラメータは最初に入力されたタイムスタンプです。

timestamp2パラメータは2番目に入力されたタイムスタンプです。

事例

```
DATE_DIFF(year, `2010-01-01T`, `2011-01-01T`)            -- 1
DATE_DIFF(year, `2010T`, `2010-05T`)                     -- 4
DATE_DIFF(month, `2010T`, `2011T`)                       -- 12
DATE_DIFF(month, `2011T`, `2010T`)                       -- -12
DATE_DIFF(day, `2010-01-01T23:00T`, `2010-01-02T01:00T`) -- 0
```

EXTRACT

EXTRACT関数は、指定されたタイムスタンプから指定された時間単位の数値を抽出します。

構文

```
EXTRACT( date_part FROM timestamp )
```

説明：

date_partパラメータは抽出したい時間単位を指定します。年、月、日、時、分、秒とすることができます。

timestampパラメータは入力されたタイムスタンプです。

事例

```
EXTRACT(YEAR FROM `2010-01-01T`)                           -- 2010
EXTRACT(MONTH FROM `2010T`)                                -- 1
EXTRACT(MONTH FROM `2010-10T`)                             -- 10
EXTRACT(HOUR FROM `2017-01-02T03:04:05+07:08`)             -- 3
EXTRACT(MINUTE FROM `2017-01-02T03:04:05+07:08`)           -- 4
EXTRACT(TIMEZONE_HOUR FROM `2017-01-02T03:04:05+07:08`)    -- 7
EXTRACT(TIMEZONE_MINUTE FROM `2017-01-02T03:04:05+07:08`)  -- 8
```

TO_STRING

TO_STRING関数は、タイムスタンプを指定されたフォーマットの時間文字列に変換することができます。

構文

```
TO_STRING ( timestamp time_format_pattern )
```

説明：

timestampパラメータは変換したいタイムスタンプを指定します。

time_format_patternパラメータは変換する時間のフォーマットを指定します。

フォーマット	説明	例

yy	2桁の年	98
y	4桁の年	1998
yyyy	固定された4桁の年。桁数に満たない場合はゼロ埋め	0199
M	月	1
MM	固定された2桁の月。桁数に満たない場合はゼロ埋め	01
MMM	月の英略称	Jan
MMMM	月の英文正式名称	January
MMMMM	月の最初のアルファベットの略称	J（to_timestamp関数には適用しない）
d	1か月のうちのある1日（1～31）	1
dd	固定された2桁の日（1～31）	01
a	午前午後の指標（AM/PM）	AM
h	時間。12時間制	1
hh	固定された2桁の時刻。12時間制	01
H	時間。24時間制	1
HH	固定された2桁の時刻。24時間制	01
m	分（00～59）	1
mm	固定された2桁の分。24時間制	01
s	秒（00～59）	1
ss	固定された2桁の時刻。24時間制	01
S	秒の小数部分（精度0.1、範囲0.0～0.9）	0
SS	秒の小数部分（精度0.01、範囲0.00～0.99）	6
SSS	秒の小数部分（精度0.001、範囲0.000～0.999）	60
...	...	...
SSSSSSSSS	秒の小数部分（精度0.000000001，範囲0.000000000～0.999999999）	60000000

n	ナノ秒	60000000
X	時間単位のオフセット。オフセットが0の場合は「Z」	+01またはZ
XXまたはXXXX	時間または分単位のオフセット。オフセットが0の場合は「Z」	+0100またはZ
xxxまたはxxxxx	時間または分単位のオフセット。オフセットが0の場合は「Z」	+01:00またはZ
x	時間単位のオフセット	1
xxまたはxxxx	時間または分単位のオフセット	0100
xxxまたはxxxxx	時間または分単位のオフセット	01:00

事例

```
TO_STRING(`1998-07-20T20:18Z`,  'MMMM d, y')                    -- "July 20, 1998"
TO_STRING(`1998-07-20T20:18Z`, 'MMM d, yyyy')                   -- "Jul 20, 1998"
TO_STRING(`1998-07-20T20:18Z`, 'M-d-yy')                        -- "7-20-69"
TO_STRING(`1998-07-20T20:18Z`, 'MM-d-y')                        -- "07-20-1998"
TO_STRING(`1998-07-20T20:18Z`, 'MMMM d, y h:m a')               -- "July 20, 1998 8
TO_STRING(`1998-07-20T20:18Z`, 'y-MM-dd''T''H:m:ssX')           -- "1998-07-20T20:1
TO_STRING(`1998-07-20T20:18+08:00Z`, 'y-MM-dd''T''H:m:ssX')     -- "1998-07-20T20:1
TO_STRING(`1998-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXX')   -- "1998-07-20T20:1
TO_STRING(`1998-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXXX')  -- "1998-07-20T20:1
```

TO_TIMESTAMP

TO_TIMESTAMP関数は時間文字列をタイムスタンプに変換することができます。

構文

```
TO_TIMESTAMP ( string )
```

説明：

stringパラメータは入力された時間文字列です。

事例

```
TO_TIMESTAMP('2007T')                         -- `2007T`
TO_TIMESTAMP('2007-02-23T12:14:33.079-08:00') -- `2007-02-23T12:14:33.079-08:00`
```

UTCNOW

UTCNOW関数はUTCの現在時刻をタイムスタンプとして返すことができます。

構文

```
UTCNOW()
```

事例

```
UTCNOW() -- 2019-01-01T14:23:12.123Z
```

文字列関数

COS Selectは次の文字列関数をサポートしています。

CHAR_LENGTH, CHARACTER_LENGTH

CHAR_LENGTHおよびCHARACTER_LENGTHはいずれも文字列内の文字数を計算することができます。この2つの関数は同一のものです。

構文

```
CHAR_LENGTH ( string )
```

説明：

stringパラメータは文字数を計算したい文字列を指定します。

事例

構文

```
CHAR_LENGTH('null')       -- 4
CHAR_LENGTH('tencent')    -- 7
```

LOWER

LOWER関数は指定された文字列内のすべての大文字アルファベットを小文字アルファベットに変換することができます。大文字アルファベット以外はすべてそのまま維持されます。

構文

```
LOWER ( string )
```

説明：

stringパラメータは大文字アルファベットを小文字アルファベットに変換したい文字列を指定します。

事例

説明：

```
LOWER('TENcent') -- 'tencent'
```

SUBSTRING

SUBSTRING関数はある文字列の部分文字列を返すことができます。あるインデックスを指定すると、SUBSTRING関数はそのインデックスから開始し、指定された部分文字列の長さに従って元の文字列から残りの内容を切り取り、結果を返します。

説明：

入力された文字列が1文字のみであり、かつインデックスの設定が1より大きい場合、SUBSTRING関数は自動的に1に切り替えます。

構文

```
SUBSTRING( string FROM start [ FOR length ] )
```

説明：

stringパラメータは部分文字列を切り取りたい文字列を指定します。

startパラメータは文字列のあるインデックス値であり、切り取る位置を指定します。

lengthパラメータは部分文字列の長さを指定します。特に指定されない場合は、文字列の残りのすべての内容を切り取ります。

事例

```
SUBSTRING("123456789", 0)      -- "123456789"
SUBSTRING("123456789", 1)      -- "123456789"
SUBSTRING("123456789", 2)      -- "23456789"
SUBSTRING("123456789", -4)     -- "123456789"
SUBSTRING("123456789", 0, 999) -- "123456789"
SUBSTRING("123456789", 1, 5)   -- "12345"
```

TRIM

TRIM関数は指定された文字列の先頭の文字の前または末尾の文字の後の全ての文字を削除することができます。デフォルトの削除文字は「 」です。

構文

```
TRIM ( [[LEADING | TRAILING | BOTH remove_chars] FROM] string )
```

説明：

stringパラメータは操作したい文字列を指定します。

`LEADING | TRAILING | BOTH` パラメータは文字列の前（LEADING）または文字列の後（TRAILING）またはその両方（BOTH）の削除したい余分な文字を指定します。

remove_charsパラメータはどの種類の余分な文字を削除したいかを指定します。remove_charsパラメータは長さが1文字より長い文字列とすることができます。TRIM関数はstringパラメータの前後でこれにマッチする余分な文字列を発見すると、そのすべてに削除処理を行います。

事例

```
TRIM('       foobar         ')               -- 'foobar'
TRIM('      \\tfoobar\\t         ')            -- '\\tfoobar\\t'
TRIM(LEADING FROM '       foobar         ')  -- 'foobar'
TRIM(TRAILING FROM '       foobar         ') -- 'foobar'
TRIM(BOTH FROM '       foobar         ')     -- 'foobar'
TRIM(BOTH '12' FROM '1112211foobar22211122') -- 'foobar'
```

UPPER

UPPER関数はすべての小文字を大文字に変換することができます。小文字以外の文字は変更されません。

構文

```
UPPER ( string )
```

説明：

stringパラメータは大文字に変換したい文字列を指定します。

事例

```
UPPER('tenCENT') -- 'TENCENT'
```

予約フィールド

最終更新日：：2024-06-26 10:57:13

機能の正常な使用とその後の拡張を保証するため、COS Select機能では次の予約フィールドを制定しています。これには関数名、データタイプ、演算子などのフィールドが含まれます。SQLを使用する際、これらのフィールドを使用することで照会に役立つ可能性があります。

番号	フィールド	番号	フィールド	番号	フィールド	番号	フィールド	番号
1	absolute	51	create	101	goto	151	octet_length	201
2	action	52	cross	102	grant	152	of	202
3	add	53	current	103	group	153	on	203
4	all	54	current_date	104	having	154	only	204
5	allocate	55	current_time	105	hour	155	open	205
6	alter	56	current_timestamp	106	identity	156	option	206
7	and	57	current_user	107	immediate	157	or	207
8	any	58	cursor	108	in	158	order	208
9	are	59	date	109	indicator	159	outer	209
10	as	60	day	110	initially	160	output	210
11	asc	61	deallocate	111	inner	161	overlaps	211
12	assertion	62	dec	112	input	162	pad	212
13	at	63	decimal	113	insensitive	163	partial	213
14	authorization	64	declare	114	insert	164	pivot	214
15	avg	65	default	115	int	165	position	215
16	bag	66	deferrable	116	integer	166	precision	216
17	begin	67	deferred	117	intersect	167	prepare	217
18	between	68	delete	118	interval	168	preserve	218
19	bit	69	desc	119	into	169	primary	219

20	bit_length	70	describe	120	is	170	prior	220
21	blob	71	descriptor	121	isolation	171	privileges	221
22	bool	72	diagnostics	122	join	172	procedure	222
23	boolean	73	disconnect	123	key	173	public	223
24	both	74	distinct	124	language	174	read	224
25	by	75	domain	125	last	175	real	225
26	cascade	76	double	126	leading	176	references	226
27	cascaded	77	drop	127	left	177	relative	227
28	case	78	else	128	level	178	restrict	228
29	cast	79	end	129	like	179	revoke	229
30	catalog	80	end-exec	130	limit	180	right	230
31	char	81	escape	131	list	181	rollback	231
32	char_length	82	except	132	local	182	rows	232
33	character	83	exception	133	lower	183	schema	233
34	character_length	84	exec	134	match	184	scroll	234
35	check	85	execute	135	max	185	second	235
36	clob	86	exists	136	min	186	section	236
37	close	87	external	137	minute	187	select	237
38	coalesce	88	extract	138	missing	188	session	238
39	collate	89	false	139	module	189	session_user	239
40	collation	90	fetch	140	month	190	set	240
41	column	91	first	141	names	191	sexp	241
42	commit	92	float	142	national	192	size	242
43	connect	93	for	143	natural	193	smallint	-
44	connection	94	foreign	144	nchar	194	some	-

45	constraint	95	found	145	next	195	space	-	
46	constraints	96	from	146	no	196	sql	-	
47	continue	97	full	147	not	197	sqlcode	-	
48	convert	98	get	148	null	198	sqlerror	-	
49	corresponding	99	global	149	nullif	199	sqlstate	-	
50	count	100	go	150	numeric	200	string	-	

データタイプ

最終更新日：：2024-06-26 10:57:13

概要

COS Selectは複数のプリミティブデータ型をサポートしています。

説明：

コンパイラによって直接サポートされるデータタイプを**プリミティブデータ型**と呼びます。

データタイプの変換

COS Selectは入力されたデータのデータタイプをCAST関数によって決定します。一般的に、お客様がCAST関数によるデータタイプの指定を行っていない場合、COS Selectは入力されたデータのタイプをstring型と判断します。

CAST関数に関する情報についてさらに詳しくお知りになりたい場合は、SQL関数のドキュメントのCASTの章をご参照ください。

サポートするデータタイプ

COS Selectは次のプリミティブデータ型をサポートしています。

名称	説明	例
bool	TRUE/FALSE	FALSE
int，integer	8バイトの符号付整数 範囲は-9,223,372,036,854,775,808 - 9,223,372,036,854,775,807	100000
string	UTF-8エンコーディングの文字列です。文字の長さの範囲は1 - 2,147,483,647	'xyz'
float	8バイトの浮動小数点数	CAST(0.456 AS FLOAT)
decimal, numeric	10進数の数値であり、最大精度は小数38桁です。数値の範囲は$-2^{31}$ - $2^{31}-1$	123.456
timestamp	タイムスタンプはある決まった時刻を表すもので、任意の精度をサポートします。テキスト形式のタイムスタンプはW3Cルールに従いますが、	CAST('2007-04-05T14:30Z'

	「T」で終わる必要があります（「日」を記録単位にしている場合を除く）。 小数秒を使用する場合は、少なくとも小数点以下1桁の精度まで維持する必要があり、小数点以下は任意の桁数を維持できます。 ローカルタイムオフセットはUTCからの時間差を用いて表すか、または「Z」を使用してUTCからのローカル時間との差を表すことができます。タイムオフセットは、日付だけを記録する場合には表示する必要がありません。	AS TIMESTAMP)

演算子

最終更新日：：2024-06-26 10:57:13

COS Selectは次の演算子をサポートしています。

演算子の種類	演算子
論理演算子	AND，NOT，OR
比較演算子	<，>，<=，>=，=，<>，!=，BETWEEN，IN。例えば`IN ('a', 'b', 'c')`など
パターンマッチ演算子	LIKE
算術演算子	+，-，*，%

演算子の優先順位

次の表に、演算子の操作順序を降順で表示します。

演算子/要素	関連性	ユースケース
-	右	単項マイナス
*, /, %	左	剰、除、モジュロ
+, -	左	加、減
IN	-	メンバー資格の設定
BETWEEN	-	（）の範囲内にある
LIKE	-	文字列パターンマッチ
<>	-	小なり、大なり
=	右	等価、割り当て
NOT	右	論理否定
AND	左	論理積
OR	左	論理和

ログ管理

ログ管理の概要

最終更新日：：2024-06-26 10:57:13

概要

ログ管理機能は、指定したソースバケットへの詳細なアクセス情報を記録し、これらの情報をログファイル形式で指定のバケットに保存することで、バケットのより適切な管理を実現するものです。
ターゲットバケット内のログレコードのパスは次のとおりです。

```
ターゲットバケット/パスプレフィックス{YYYY}/{MM}/{DD}/{time}_{random}_{index}
```

ログは5分に1回生成され、1件のレコードにつき1行となります。各レコードに複数のフィールドが含まれ、フィールドの間はスペースで区切られます。単一のログファイルは最大256MBまでであり、この5分間に生成されたログの量が256MBを超えた時点で、ログが複数のログファイルに分割される点に注意が必要です。現在サポートされているログフィールドは次のとおりです。

フィールド番号	名称	意味	例

1	eventVersion	レコードのバージョン	1.0
2	bucketName	バケット名	examplebucket-1250000000
3	qcsRegion	リクエストリージョン	ap-beijing
4	eventTime	イベント時間（リクエスト終了時間、UTC 0時 タイムスタンプ）	2018-12-01T11:02:33Z
5	eventSource	ユーザーがアクセスしたドメイン名	examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com
6	eventName	イベント名	UploadPart
7	remoteIp	ソースIP	192.168.0.1
8	userSecretKeyId	ユーザーアクセスKeyId	AKIDNYVCdoJQyGJ5b1234
9	reservedFiled	予約フィールド	予約フィールドは `-` と表示されます。
10	reqBytesSent	リクエストバイト数（Bytes）	83886080
11	deltaDataSize	ストレージ量変更のリクエスト（Bytes）	808
12	reqPath	リクエストのファイルパス	/folder/text.txt
13	reqMethod	リクエストメソッド	put
14	userAgent	ユーザーUA	cos-go-sdk-v5.2.9
15	resHttpCode	HTTP戻りコード	404

16	resErrorCode	エラーコード	NoSuchKey
17	resErrorMsg	エラーメッセージ	The specified key does not exist.
18	resBytesSent	戻りバイト数（Bytes）	197
19	resTotalTime	リクエスト総消費時間（ミリ秒、応答のTTLB-リクエストのTTFBと同じ）	4295
20	logSourceType	ログソースタイプ	USER（ユーザーアクセスリクエスト）、CDN（CDN back-to-originリクエスト）
21	storageClass	ストレージタイプ	STANDARD，STANDARD_IA，ARCHIVE
22	accountId	バケット所有者のID	100000000001
23	resTurnAroundTime	リクエストサーバー消費時間（ミリ秒、応答のTTFB-リクエストのTTLBと同じ）	4295
24	requester	アクセス者のアカウント。アクセス者がルートアカウントの場合、アカウント形式は `ルートアカウントUIN:ルートアカウントUIN` となります。アクセス者がサブアカウントの場合、ア	100000000001:100000000001

		カウント形式は `ルートアカウントUIN:サブアカウントUIN` となります。 アクセス者がサービスアカウントの場合、アカウント形式は `権限を承認されたルートアカウントUIN:ロールID` となります。 匿名アクセスの場合は、`-` と表示されます	
25	requestId	リクエストID	NWQ1ZjY4MTBfMjZiMjU4NjRfOWI1N180NDBiYTY=
26	objectSize	オブジェクトサイズ（Bytes）	808。マルチパートアップロードを使用した場合、objectSizeフィールドはアップロード完了後にしか表示されません。各パートのアップロード中、このフィールドには `-` と表示されます
27	versionId	オブジェクトのバージョンID	ランダムな文字列
28	targetStorageClass	ターゲットバケットのタイプ。コピー操作のリクエストを送信すると、このフィールドが記録されます	STANDARD，STANDARD_IA，ARCHIVE
29	referer	リクエストのHTTP referer	`*.example.com` または111.111.111.1
30	requestUri	リクエストのURI	"GET /fdgfdgsf%20/%E6%B5%AE%E7%82%B9%E6%95%B0

			HTTP/1.1"
31	vpcId	VPCリクエストのID	"0"(VPCではない)/"12345"(VPC。「0」以外のstringとします)

注意：

現在、COSのログ管理機能でサポートされているリージョンには、北京、上海、広州、南京、重慶、成都、中国香港、シンガポール、ソウル、トロント、シリコンバレー、ムンバイがあります。

ログ管理機能を使用するには、ソースバケットとターゲットバケットが同一のリージョンにあることが必要です。

ログを保存するターゲットバケットはソースバケットと同じにすることもできますが、推奨されません。

現時点では、XML APIおよびXML APIをベースにして実装したSDK、ツールなどによってバケットへのアクセスリクエストを行った場合にのみ、ログが記録されます。JSON APIおよびJSON APIをベースにして実装したSDK、ツールなどによるアクセスでは、ログは記録されません。

ユーザーのニーズおよび業務の発展に応じて、COSはアクセスログに新しいフィールドを追加する場合があります。ログ解析の際に必ず適切に処理してください。

ログ管理の有効化

コンソールの使用

ログ管理機能はコンソールでスピーディーに有効化することができます。操作ガイドについては、ログ管理の設定のコンソールガイドをご参照ください。

APIの使用

APIを使用して、指定のバケットでログ管理機能を有効化する場合は、次の手順をご参照ください。

1. ログロールを作成します。
2. ログロールに権限をバインドします。
3. ログ管理を有効化します。

1. ロググロールの作成

ロググロールを作成します。具体的なインターフェースの情報については、CreateRoleをご参照ください。

このうち、roleNameは必ずCLS_QcsRoleとします。

policyDocumentは次のとおりです。

```
    }
}
```

```
{
    "version": "2.0",
    "statement": [{
        "action": "name/sts:AssumeRole",
        "effect": "allow",
        "principal":{
            "service": "cls.cloud.tencent.com"
        }
    }]
}
```

2. ログロールへの権限のバインド

ロール権限に権限をバインドします。具体的なインターフェースの情報については、AttachRolePolicyをご参照ください。
このうち、policyNameはQcloudCOSAccessForCLSRoleとします。roleNameは手順1のCLS_QcsRoleとします。roleNameの作成時に返されたroleIDを使用することもできます。

3. ログ管理の有効化

インターフェースを呼び出してログ管理機能を有効化します。具体的なインターフェースの情報については、PUT Bucket loggingをご参照ください。ログを保存するターゲットバケットはソースバケットと同一のリージョンにあることが必要です。

ログ管理の制限

最終更新日：：2024-06-26 10:57:13

ログ管理機能は、指定したソースバケットについて、その詳細なアクセス情報を記録し、これらの情報をログファイル形式で指定のバケットに保存することで、バケットのより適切な管理を実現するものです。

現在、アクセスログ管理機能には次のような使用上の制限があります。

配信頻度制限：ログは5分に1回生成されます。

配信ログファイルのサイズ制限：1回に配信できるログファイルは最大256MBまでであり、この制限を超えた時点で新しいファイルとして配信されます。

ログ配信形式：1件のレコードにつき1行となり、各レコードに複数のフィールドが含まれます。フィールドの間はスペースで区切ります。

配信フィールドの制限：配信フィールドの制限に関しては、ログ管理の概要の説明をご参照ください。

無効なフィールドの説明：ログの中に `-` という記号が存在する場合、そのフィールドは無効またはデフォルトのレコードであることを表します。

ログに記録される内容

ユーザーが送信したオブジェクトのアップロード、ダウンロード、ならびにバケットの作成・削除、バケット設定の変更などのリクエスト。

ユーザーがCDNを通じてコンテンツを配信し、CDNからCOSオリジンサーバーに戻ってデータをプルした際に発生したリクエスト。

ログに記録されない内容

オフラインback-to-originリクエスト：ユーザーがback-to-originを設定した後、COS内にオブジェクトがなく、ユーザーが指定したオリジンサーバーからデータをダウンロードした場合、そのダウンロード操作をオフラインback-to-originリクエストといいます。これはログには記録されません。

静的ウェブサイト内のリダイレクト操作：ユーザーが静的ウェブサイト機能でリダイレクトを設定している場合、index.htmlにアクセスすると他のページにリダイレクトされる場合があります。このようなリダイレクト操作はログには記録されません。

ライフサイクルによる移行、削除などの操作：ユーザーがライフサイクル機能を設定し、期限切れとなったオブジェクトを移行または削除する場合、これらの移行および削除操作はCOSバックエンドで実現されるため、ログには記録されません。

オブジェクトのリストアップおよびリストレポートのアップロード操作：リスト機能はユーザーに代わり、定期的にバケット内の全部または指定のオブジェクトをリストアップし、生成したリストレポートをユーザーのバ

ケットに配信するものです。オブジェクトのリストアップおよびリストレポート配信の操作はログには記録されません。

オブジェクトの地域間コピー操作：地域間コピー機能ではソースバケットからオブジェクトを取得し、そのオブジェクトをターゲットバケットにアップロードする必要があります。これらの操作はCOSバックエンドで実現されるため、ログには記録されません。

COS Select機能におけるオブジェクトダウンロード操作：COS Select機能はユーザーのオブジェクト検索をアシストするもので、検索を行うには先にストレージデバイスからデータを取得する必要があります。オブジェクトのダウンロード操作はCOSバックエンドで実現されるため、ログには記録されません。

COSを使用したクラウド製品ログのストレージ

最終更新日：：2024-06-26 10:57:13

概要

Tencent Cloud製品を使用する際には大量のログが生成されます。これらのログはお客様の業務の状況を記録するもので、業務状況の分析に役立ち、業務の発展と方針決定を支援します。COSのストレージ機能を利用することで、クラウド製品ログの永続ストレージを実現することができるほか、API、SDKまたはツールなどの方法で、COSから便利かつスピーディーにログを取得し、分析を行うことができます。

COSを使用したクラウド製品ログのストレージは、次のような問題の解決に役立ちます。

永続ストレージ：COSは安定した永続的なストレージサービスをご提供します。ログをCOSに保存することで、永続ストレージを極めて低コストで実現することができます。業務上、ログに基づいた分析または方針決定が必要な場合は、COSによっていつでもどこでも任意の時間帯のログを取得することが可能です。

データ検索：COSはSelect機能をご提供しています。この機能を使用することで、COSに保存されたログを簡単に検索および抽出することができます。ログのフィールドを結合し、必要な情報の検索とデータダウンロードトラフィックの削減に役立てることも可能です。

ログ配信方式

次の2つの方法でTencent Cloud製品のログをCOS上に保存することができます。

クラウド製品に付属しているログ配信機能を使用：例えばCloud Object Storage（COS）、Cloud Load Balancer（CLB）、CloudAudit（CA）などの製品では、ログを直接COSに配信することができます。

Cloud Log Service（CLS）の配信機能を使用：CLSに配信されるクラウド製品のログは、CLSによってCOS上に配信されて永続ストレージとなります。

現在のTencent Cloud製品の、これらの2つの方法に対するサポート状況は次のとおりです。

クラウド製品名	COSへの直接配信をサポートしているか	CLSへの配信をサポートしているか
CloudAudit（CA）	はい	いいえ
Cloud Load Balancer（CLB）	いいえ	はい
Cloud Kafka	はい	いいえ
API Gateway	いいえ	はい

Serverless Cloud Function（SCF）	いいえ	はい
Tencent Kubernetes Engine（TKE）	いいえ	はい
CSS	いいえ	はい
クラウド開発TCB	いいえ	はい、ただしCLSによるCOSへの配信はサポートなし
COS（Cloud Object Storage）	はい	ホワイトリストアクティブ化申請のサポートあり。テクニカルサポートWeChatグループに加入し、ホワイトリストアクティブ化についてお問い合わせください

COSへのログ直接配信

次のTencent Cloud製品はCOSへのログ直接配信機能を備えています。対応する製品ドキュメントガイドをご参照の上、ログ配信ルールを設定し、ログをCOSに配信することができます。

クラウド製品名	ログ配信ドキュメント	ログ配信間隔	ログ配信パス
CA（CloudAudit）	ここをクリックして確認	10～15分	cloudaudit/customprefix/timestamp
Cloud Kafka	ここをクリックして確認	5分～60分 配信間隔を指定可能	instance id/topic id/timestamp
Cloud Object Storage（COS）	ここをクリックして確認	5分	パスのプレフィックスはご自身で指定できます。例えばcos_bucketname_access_log/timestampのように、識別可能なパスを設定することをお勧めします

説明：

Cloud Kafkaではその製品上で生成されたメッセージデータの配信をサポートしています。CKafkaインスタンスの作成などの行動ログを取得したい場合は、CloudAudit製品のログ配信を選択することができます。

Cloud Log Service（CLS）によるCOSへのログ配信

次のTencent Cloud製品は、ユーザーの検索と分析に役立つよう、Cloud Log Service（CLS）へのログ配信をサポートしています。CLSはまたCOSへの配信を行う製品機能をご提供し、ログストレージの永続化に役立てるこ

とができます。CLSへのログ配信をサポートする製品では、CLS側でCOSへのログ配信を有効化する方式により、データを永続ストレージにしてストレージコストを引き下げ、さらなるオフライン分析に役立てることも可能です。現在、CLSへのログ直接配信をサポートしている製品は次のとおりです。

クラウド製品名	ログ配信ドキュメント
API Gateway	ここをクリックして確認
Tencent Kubernetes Engine（TKE）	ここをクリックして確認
CSS	ここをクリックして確認

CLSによるCOSへの配信には3つの方式がサポートされています。

区切り文字形式による配信：データを区切り文字形式でCOSに配信できます。詳細については、区切り文字形式による配信をご参照ください。

JSON形式による配信：データをJSON形式でCOSに配信できます。詳細については、JSON形式による配信をご参照ください。

データをオリジナルテキスト形式で配信することができます。シングルライン全文、マルチライン全文の配信をサポートし、CSV形式の配信も一部サポートしています。詳細については、オリジナルテキスト形式による配信をご参照ください。

CLSによるCOSへのログ配信では、次の操作を実行する必要があります。

1. 業務上のニーズに応じて対応する製品を選択し、上記でご提供した製品ログ配信についてのドキュメントリンクガイドに従ってログセットとログトピックを設定し、業務で発生したデータをCLSに結合します。
2. その後、業務上の必要に応じ、適切な形式を選択してデータをCOS上に配信します。ログをCOSに配信する際は、複数の製品のログを区別するため、製品名をパスのプレフィックスとすることをお勧めします。例えばTKEのログの場合、 `TKE_tkeid_log/timestamp` と命名することができます。
3. 配信ルールを設定した後、さらにSCF製品にファイルアップロードイベント通知を追加で設定し、ログデータがCOSに配信された後、イベント通知に基づいて次の操作を実行できるようにすることも可能です。詳細については、イベントの通知をご参照ください。

ログに対する分析

ログをローカルにダウンロードしてオフライン分析を実施

ログデータをローカルにダウンロードしたい場合は、コンソール、SDK、APIまたはツールなどの複数の方式で行うことができます。各ダウンロード方式で使用するドキュメントの説明は次のとおりです。下記のドキュメントを参照し、コードの中のファイルパスに関わる部分をログのストレージパスに置き換えることで、ログデータをローカルにダウンロードすることができます。

ダウンロード方式	使用説明

コンソール	ここをクリックして確認
cosbrowser	ここをクリックして確認
coscmd	ここをクリックして確認
Android SDK	ここをクリックして確認
C SDK	ここをクリックして確認
C++ SDK	ここをクリックして確認
.NET SDK	ここをクリックして確認
Go SDK	ここをクリックして確認
iOS SDK	ここをクリックして確認
Java SDK	ここをクリックして確認
JavaScript SDK	ここをクリックして確認
Node.js SDK	ここをクリックして確認
PHP SDK	ここをクリックして確認
Python SDK	ここをクリックして確認
ミニプログラムSDK	ここをクリックして確認
API	ここをクリックして確認

COS Selectを使用したログ分析の実施

COS Select機能を使用して、COS上に保存されたログファイルを直接検索および分析することもできます。前提として、ログファイルがCSVまたはJSON形式で保存されていることが必要です。COS Select機能によって必要なログフィールドをフィルタリングできるため、COSの転送ログデータ量を大幅に削減することができ、使用コストを引き下げると同時にデータ取得効率を向上させることが可能です。COS Select機能について詳しくお知りになりたい場合は、Selectの概要をご参照ください。

現在、COS Select機能はコンソールまたはAPI方式でご利用いただけます。

使用方式	使用説明
コンソール	ここをクリックして確認
API	ここをクリックして確認

データ災害復帰

バージョン管理

バージョン管理の概要

最終更新日：：2024-06-26 10:57:13

概要

バージョン管理は同一のバケット内に同一のオブジェクトの複数のバージョンを保存する場合に使用します。例えば、1つのバケット内に、オブジェクトキーは同じpicture.jpgでも、バージョンIDが100000、100101、120002のように異なる複数のオブジェクトを保存することができます。ユーザーがあるバケットでバージョン管理機能を有効にすると、バケット内に保存したオブジェクトをバージョンIDに基づいて照会、削除または復元できるようになります。ユーザーが誤って削除したデータや、アプリケーションプログラムの障害によって消失したデータの回復に役立ちます。例えば、ユーザーがバージョン管理されたオブジェクトの削除操作を行う場合は、次のようになります。

オブジェクトを削除したい（完全削除ではない）場合、COSは削除されるオブジェクトに削除タグを挿入し、このタグが現在のオブジェクトのバージョンとなります。この**削除タグ**に基づいて以前のバージョンを復元することができます。

オブジェクトを置き換えたい場合、COSは新たにアップロードされたオブジェクトに新しいバージョンIDを挿入します。バージョンIDに基づいて、置き換える前のオブジェクトを復元することも引き続き可能です。

バージョン管理の状態

バケットには3種類のバージョン管理状態があります。バージョン管理を有効にしていない状態、バージョン管理を有効にしている状態、バージョン管理の一時停止状態です。

バージョン管理を有効にしていない状態：バケットのデフォルトの初期状態であり、このときバージョン管理機能はオフになっています。

バージョン管理を有効にしている状態：バケットのバージョン管理機能をオンにすることを指し、このときバージョン管理は有効な状態になっています。バージョン管理状態はこのバケット内の**すべてのオブジェクト**に適用されます。バケットのバージョン管理を最初に有効にした時点から、このバケットに新たにアップロードされるオブジェクトには一意のバージョンIDが付与されます。

バージョン管理の一時停止状態：バケットのバージョン管理をオンにした状態から一時停止状態に変更することを指します（バージョン管理を有効にしていない状態に戻すことはできません）。この後にバケットにアップロードされるオブジェクトについては、バージョン管理されたオブジェクトが保存されなくなります。

注意：

1. 一度バージョン管理を有効にしたバケットは、バージョン管理を有効にしていない状態（初期状態）に戻すことはできません。ただし、このバケットのバージョン管理を一時停止することはでき、その後に新たにアップロードされるオブジェクトでは複数のバージョンが生成されなくなります。
2. バージョン管理を有効にする前にバケット内に保存されたオブジェクトのバージョンIDは、すべてnullとなっています。
3. バージョン管理を有効にする、または一時停止すると、COSがこれらのオブジェクトを処理する際のリクエスト方式が変更されます。オブジェクト自体は変更されません。
4. バケットのバージョン管理を一時停止できるのはルートアカウントと権限を持つサブアカウントのみです。

バージョン管理状態下のオブジェクトの管理

バケットがどのバージョン管理状態にある場合でも、その状態のバケット内のオブジェクトに対し、アップロード、照会および削除操作を行うことができます。バージョン管理を有効にしていない状態を除き、バージョン管理を有効にしている状態およびバージョン管理の一時停止状態であれば、バケット内のオブジェクトの照会および削除操作は、バージョンIDを指定してもしなくても行うことができます。

バージョン管理を有効にしていない状態：オブジェクトのアップロード、照会および削除などの操作方法に変更はありません。詳細については、オブジェクト管理ディレクトリ下のドキュメントをご参照ください。

バージョン管理を有効にしている状態およびバージョン管理の一時停止状態：オブジェクトのアップロード、照会および削除などの操作方法が従来の方法と異なる点は、バージョンIDが導入されている点です。オブジェクトの削除操作の実行には「削除タグ」の概念も加わっています。

バージョン管理を有効にしている状態のオブジェクトの管理

バケットバージョン管理を有効にする前にすでにバケット内に保存されていたオブジェクトについては、そのバージョンIDはnullとなります。バージョン管理を有効にした後も、バケット内の既存のオブジェクトは変更されず、既存オブジェクトに対するCOSの処理方式（リクエスト方式など）だけが変更されます。この時点で、新たにアップロードされた同名のオブジェクトは異なるバージョンとして同一のバケット内に存在することになります。バージョン管理を有効にしたバケット内でオブジェクトがどのように管理されるかについて、次にご説明します。

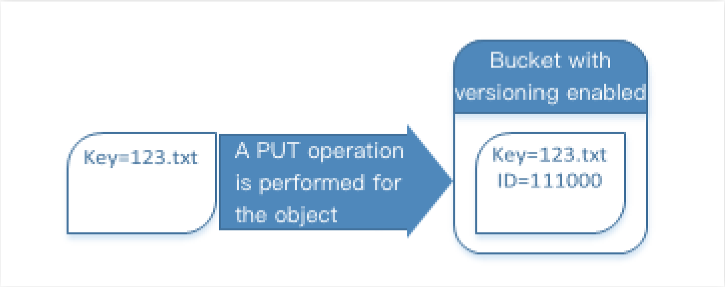
注意：

バージョン管理を有効にしていないバケットと、有効にしているバケットでは、ユーザーのオブジェクトのアップロード方法は同じですが、バージョンIDが異なります。バージョン管理を有効にしている場合、COSはオブジェクトに特定のバージョンIDを割り当てますが、バージョン管理を有効にしていない場合、バージョンIDは常にnullとなります。

オブジェクトのアップロード

バケットのバージョン管理を有効にすると、ユーザーがPUT、POSTまたはCOPY操作を実行した際、COSがこのバケットに保存するオブジェクトには一意のバージョンIDが自動的に追加されます。

下図のように、バージョン管理を有効にしたバケットにオブジェクトをアップロードすると、COSはそのオブジェクトに一意のバージョンIDを追加します。


Key=123.txt
A PUT operation is performed for the object
Bucket with versioning enabled
Key=123.txt
ID=111000

バージョン管理オブジェクトのリストアップ

COSはバケットにバインドしたversionsパラメータにオブジェクトのバージョン情報を保存します。COSは保存時刻の順序に従ってオブジェクトのバージョンを返します。その際、直近のバージョンが最初に返されます。

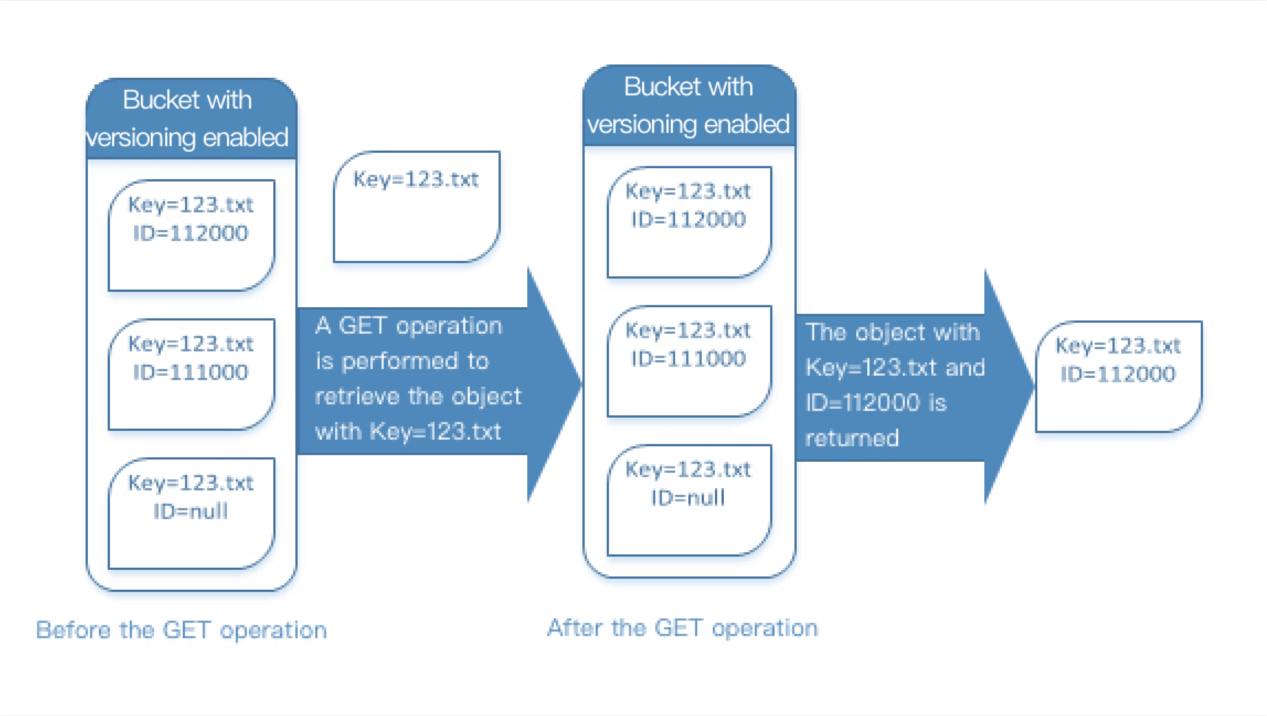
特定のオブジェクトの全バージョン照会

次のプロセスによって、versionsパラメータおよびprefixリクエストパラメータを使用して、あるオブジェクトの全バージョンを照会できます。prefixに関するその他の情報については、GET Bucket Object versionsのドキュメントをご参照ください。
あるオブジェクトの全バージョンを照会する場合の、リクエストの例は次のとおりです。

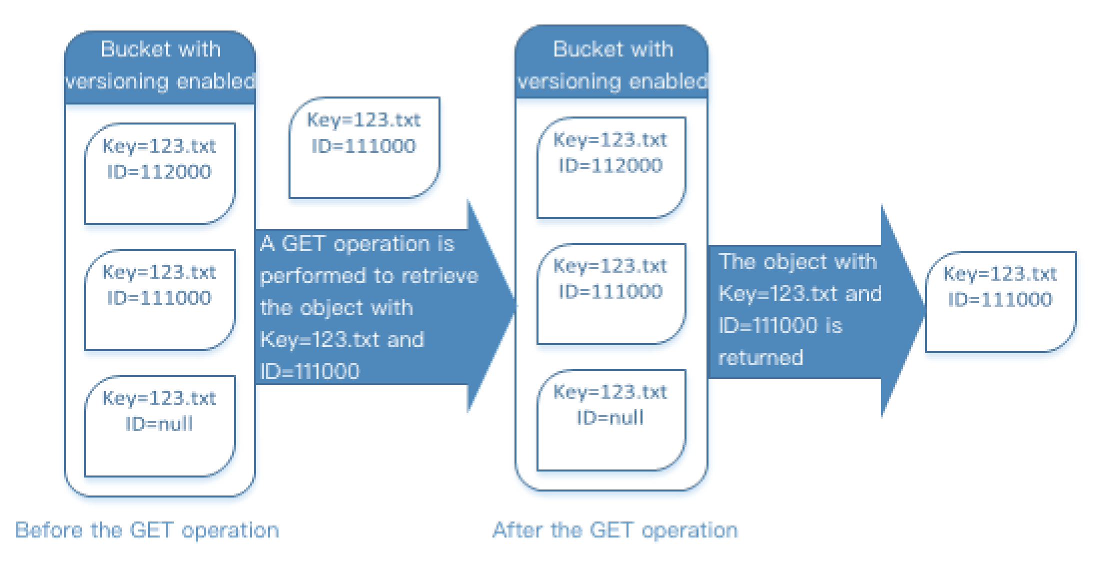
```
GET /?versions&prefix=ObjectKey HTTP/1.1
```

データのメタバージョンの照会

ユーザーがGETリクエストを使用する際にバージョンIDを指定しなければ、オブジェクトの現在のバージョンを照会します。下図のように、GETリクエストには123.txtオブジェクトの現在のバージョン（直近のバージョン）が返されます。


Bucket with versioning enabled
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt ID=null
Key=123.txt
A GET operation is performed to retrieve the object with Key=123.txt
Bucket with versioning enabled
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt ID=null
The object with Key=123.txt and ID=112000 is returned
Key=123.txt ID=112000
Before the GET operation
After the GET operation

ユーザーがGETリクエストを使用する際にバージョンIDを指定した場合は、指定したバージョンIDのオブジェクトを照会します。下図のように、GET versionIdリクエストによって指定したバージョン（現在のバージョンでも可）のオブジェクトを照会します。


Bucket with versioning enabled
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt ID=null
Key=123.txt ID=111000
A GET operation is performed to retrieve the object with Key=123.txt and ID=111000
Bucket with versioning enabled
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt ID=null
The object with Key=123.txt and ID=111000 is returned
Key=123.txt ID=111000
Before the GET operation
After the GET operation

オブジェクトバージョンのメタデータの照会

オブジェクトの内容ではなく、メタデータのみを照会したい場合は、HEAD操作を使用することができます。デフォルトでは最新バージョンのメタデータを取得します。指定したオブジェクトのバージョンのメタデータを照会したい場合は、リクエストの送信時にバージョンIDを指定します。

指定したバージョンのオブジェクトのメタデータを照会する手順は次のとおりです。

versionIdを照会するオブジェクトメタデータのバージョンIDに設定します。

指定したversionIdのHEAD操作リクエストを送信します。

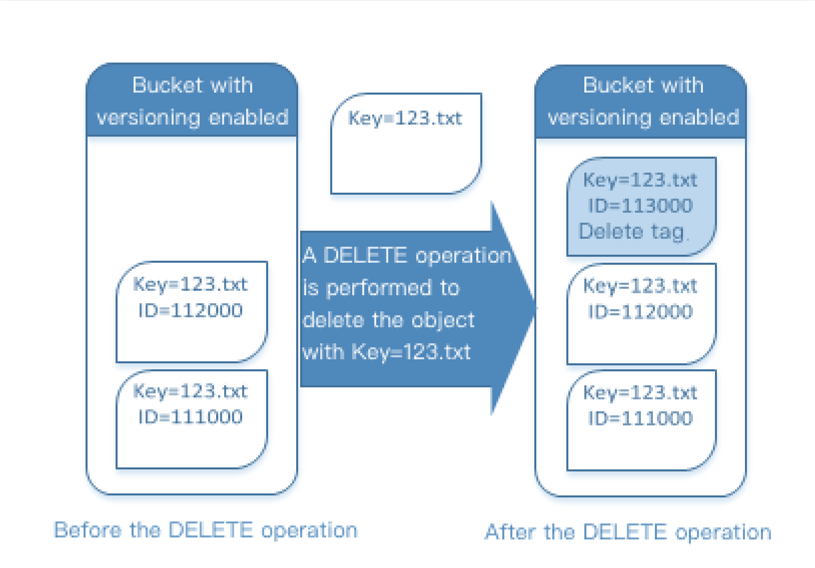
オブジェクトの削除

必要に応じて、不要なオブジェクトのバージョンを随時削除することができます。ユーザーがバージョン管理を有効にしている状態でDELETEリクエストを使用するケースには次の2つがあります。

1. ユーザーがバージョンIDを指定せず、通常のDELETE操作を実行する場合。

この操作のケースは削除されたオブジェクトを「ごみ箱」に入れる場合に似ていますが、オブジェクトを完全に消去してはおらず、その後ユーザーが必要とする場合はデータを復元できます。

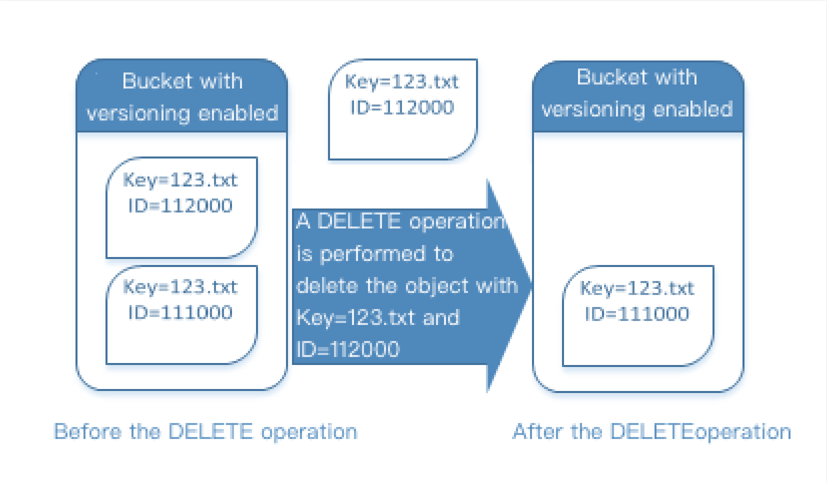
下図のように、ユーザーがDELETE操作の際にバージョンIDを指定しなかった場合、実際にはKey=123.txtのオブジェクトは削除されず、新しい削除タグが挿入され、新しいバージョンIDが追加されます。


Bucket with versioning enabled
Key=123.txt
Bucket with versioning enabled
Key=123.txt
ID=113000
Delete tag
A DELETE operation is performed to delete the object with Key=123.txt
Key=123.txt
ID=112000
Key=123.txt
ID=112000
Key=123.txt
ID=111000
Key=123.txt
ID=111000
Before the DELETE operation
After the DELETE operation

注意：

COSはバケット内で、削除されるオブジェクトに新しいバージョンIDを持つ削除タグを挿入し、この削除タグが削除されるオブジェクトの現在のバージョンとなります。この削除タグのあるオブジェクトに対しGET操作の実行を試すと、COSはこのオブジェクトが存在しないと認識し、404エラーを返します。

2. ユーザーがバージョンIDを指定して、オブジェクトバージョンの削除操作を実行した場合、このケースではバージョン管理されたオブジェクトを永久に削除することができます。


Bucket with versioning enabled
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt ID=112000
A DELETE operation is performed to delete the object with Key=123.txt and ID=112000
Bucket with versioning enabled
Key=123.txt ID=111000
Before the DELETE operation
After the DELETEoperation

初期バージョンの復元

バージョン管理によってオブジェクトの初期バージョンを復元することができます。この操作を実行するには2つの方法があります。

1. オブジェクトの初期バージョンを同一のバケットにコピーする

コピーしたオブジェクトはそのオブジェクトの現在のバージョンとなり、なおかつオブジェクトのすべてのバージョンが維持されます。

2. オブジェクトの現在のバージョンを永久に削除する

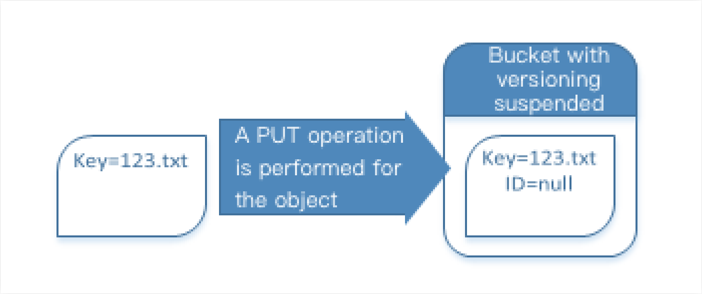
オブジェクトの現在のバージョンを削除すると、実際には1つ前のバージョンがオブジェクトの現在のバージョンに置き換わります。

バージョン管理が一時停止状態にあるオブジェクトの管理

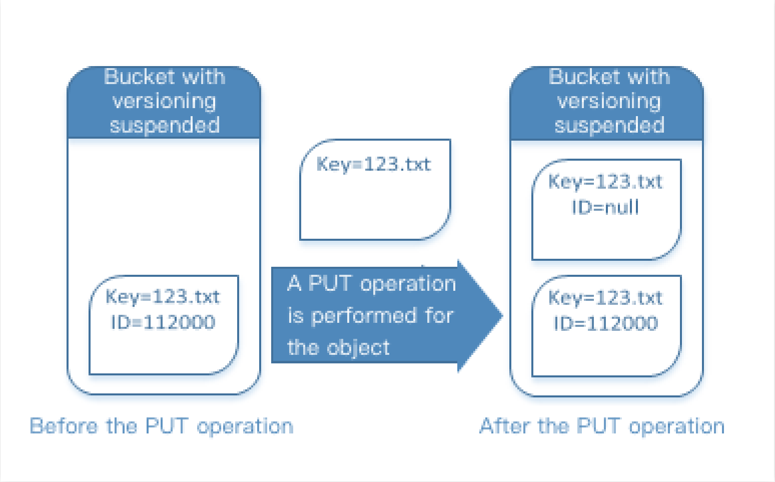
バージョン管理を一時停止した場合、バケット内の既存のオブジェクトは変更されません。変更されるのはCOSがそれ以降のリクエストにおいてオブジェクトを処理する方式です。バージョン管理を一時停止しているバケット内でオブジェクトがどのように管理されるかについて、次にご説明します。

オブジェクトのアップロード

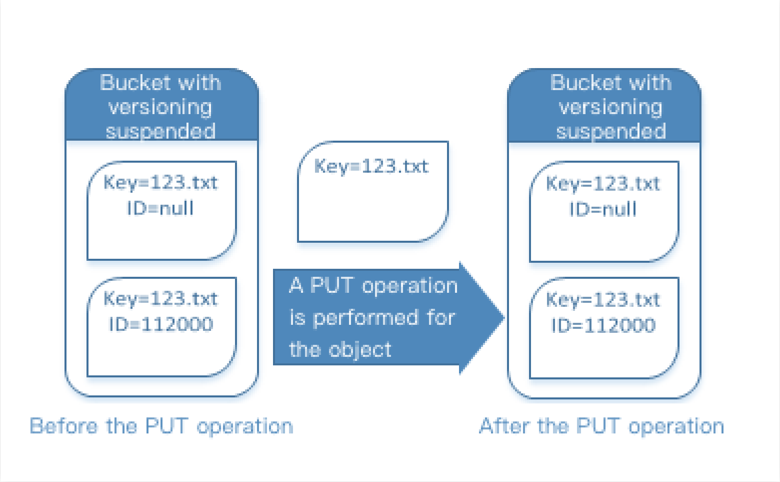
バケット上でバージョン管理を一時停止すると、ユーザーがPUT、POSTまたはCOPY操作を実行した際、COSはこのバケット内に保存されるオブジェクトに、バージョンIDをnullとして自動的に追加します。下図に示します。


Key=123.txt
A PUT operation is performed for the object
Bucket with versioning suspended
Key=123.txt
ID=null

バケット内にバージョン管理を行っているオブジェクトが存在する場合、バケットにアップロードされたオブジェクトが現在のバージョンとなり、バージョンIDはnullとなります。下図に示します。


Bucket with versioning suspended
Key=123.txt
ID=112000
Before the PUT operation
Key=123.txt
A PUT operation is performed for the object
Bucket with versioning suspended
Key=123.txt
ID=null
Key=123.txt
ID=112000
After the PUT operation

バケット内に空のバージョンがすでに存在する場合、この空のバージョンは上書きされ、それまでのオブジェクトの内容もそれに応じて置き換えられます。下図に示します。


Bucket with versioning suspended
Key=123.txt
ID=null
Key=123.txt
ID=112000
Before the PUT operation
Key=123.txt
A PUT operation is performed for the object
Bucket with versioning suspended
Key=123.txt
ID=null
Key=123.txt
ID=112000
After the PUT operation

データのメタバージョンの照会

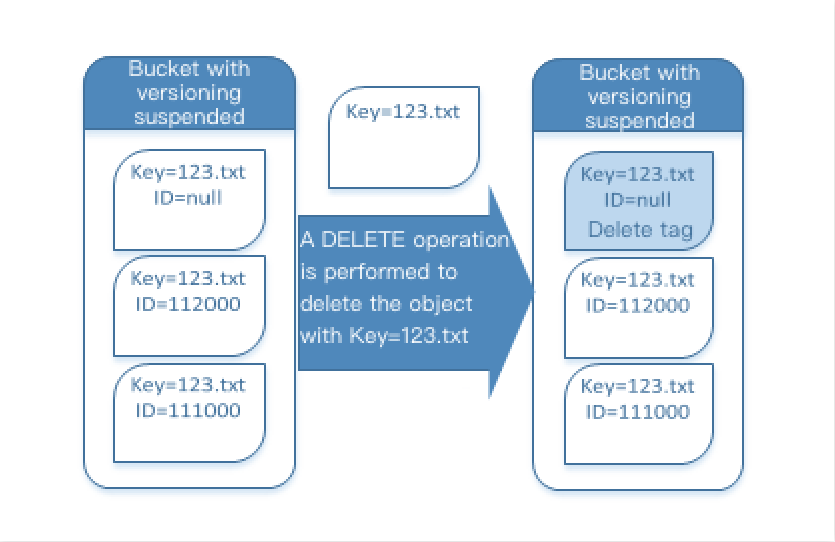
バージョン管理を一時停止したバケットで、ユーザーがGET Objectリクエストを送信すると、オブジェクトの現在のバージョンが返されます。

オブジェクトの削除

バージョン管理を一時停止した状態でDELETEリクエストを実行すると、次のようになります。

バケット内に空のバージョンのオブジェクトが存在する場合は、バージョンIDがnullのオブジェクトを削除します。

下図のように、ユーザーが通常のDELETE操作を実行した際、COSは空のバージョンのオブジェクトに削除タグを挿入します。

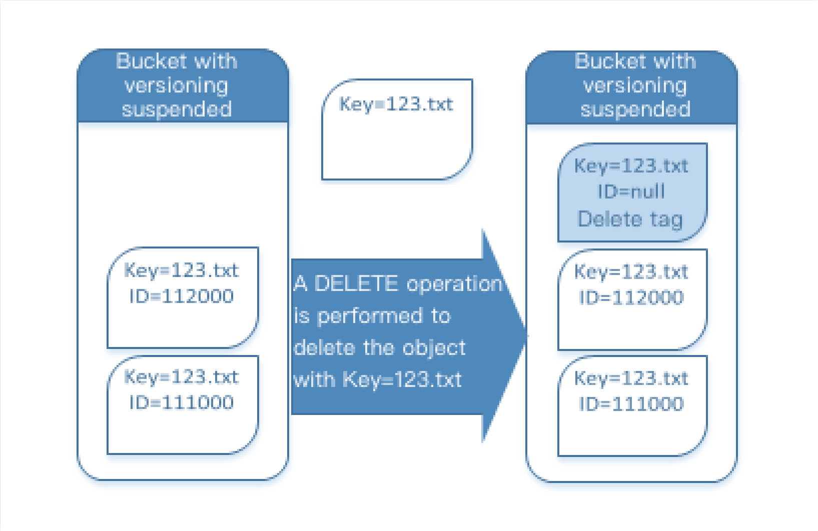

Bucket with versioning suspended
Key=123.txt ID=null
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt
A DELETE operation is performed to delete the object with Key=123.txt
Bucket with versioning suspended
Key=123.txt ID=null Delete tag
Key=123.txt ID=112000
Key=123.txt ID=111000

注意：

削除タグには内容が存在しません。削除タグが空のバージョンに置き換わった時点で、空のバージョンの元の内容は失われます。

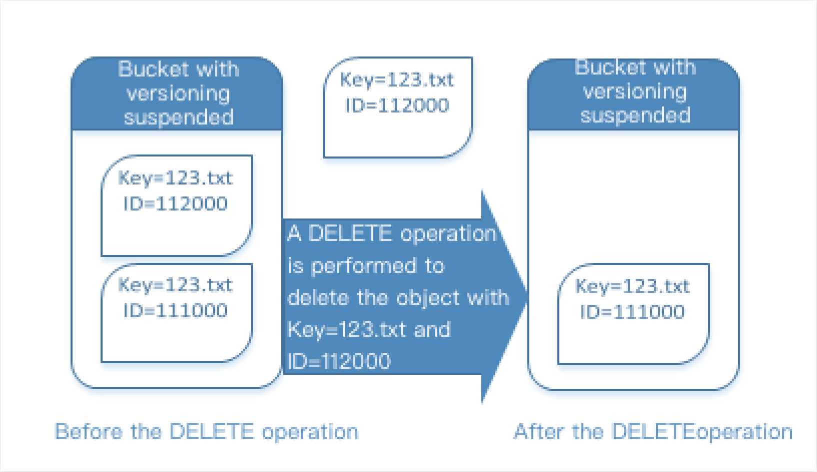
バケット内に空のバージョンのオブジェクトが存在しない場合は、バケット内に新たに削除タグが追加されます。

下図のように、バケットに空のバージョンが存在しない場合、ユーザーがDELETE操作を実行してもいかなる内容も削除されず、COSが削除タグを挿入するだけとなります。


Bucket with versioning suspended
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt
A DELETE operation is performed to delete the object with Key=123.txt
Bucket with versioning suspended
Key=123.txt ID=null Delete tag
Key=123.txt ID=112000
Key=123.txt ID=111000

バージョン管理を一時停止しているバケットであっても、ルートアカウントであれば指定したバージョンを永久に削除することができます。

下図のように、指定したオブジェクトのバージョンを削除すると、そのオブジェクトは永久に削除されます。


Bucket with versioning suspended
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt ID=112000
A DELETE operation is performed to delete the object with Key=123.txt and ID=112000
Bucket with versioning suspended
Key=123.txt ID=111000
Before the DELETE operation
After the DELETEoperation

注意：

指定したオブジェクトのバージョンを削除できるのは、ルートアカウントまたはルートアカウントから権限を付与されたアカウントのみです。

タグの削除

最終更新日：：2024-06-26 10:57:13

概要

削除マーカーはバージョン管理されているオブジェクトに用いられます。削除マーカーはCOS内で「オブジェクトがすでに削除されている」ことを示すマーカーとみなすことができます。削除マーカーにはオブジェクトと同様に、オブジェクトキー（Key）とバージョンIDがあります。その違いは次のいくつかの点にあります。

削除マーカーは内容が空です。

削除マーカーにはACL値が存在しません。

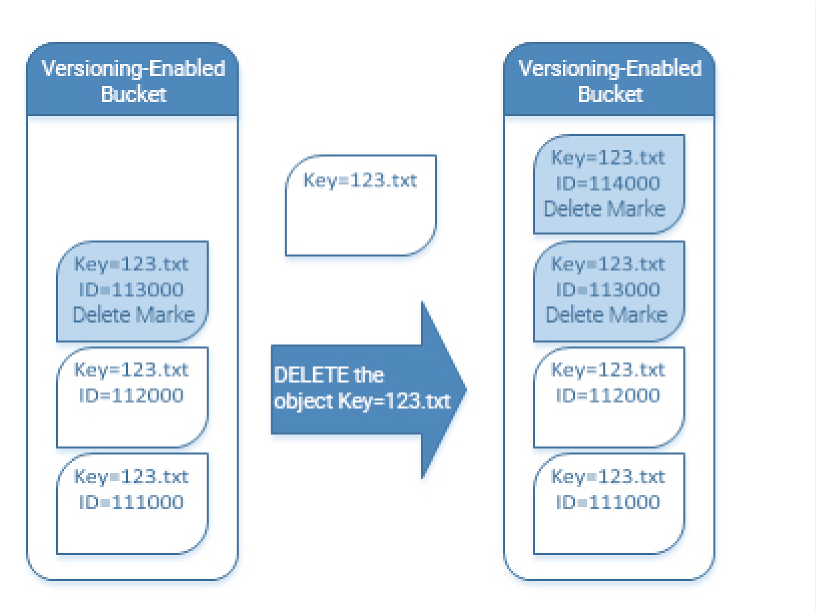
削除マーカーがGETリクエストを実行するとエラーコード404が返されます。

削除マーカーはDELETE操作のみサポートされます（ルートアカウントによるリクエスト送信が必要）。

「削除マーカー」の削除

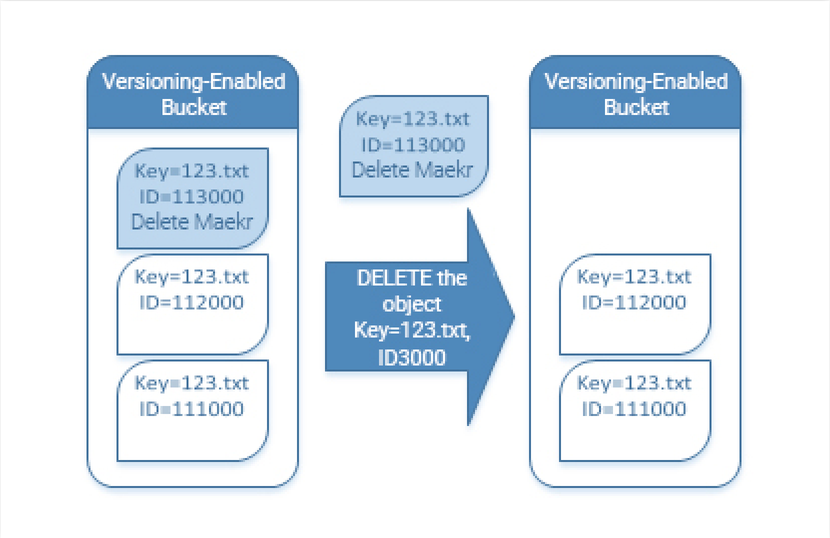
ユーザーが「削除マーカー」を削除したい場合は、DELETE Object versionIdリクエストでそのバージョンIDを指定することで、「削除マーカー」を完全に削除することができます。削除マーカーのバージョンIDを指定せず、削除マーカーに対しDELETEリクエストを送信した場合、COSはその削除マーカーを削除せず、新たな削除マーカーを挿入します。

下図のように、削除マーカーに対し通常のDELETEリクエストを実行しても、いかなる内容も削除されず、バケット内には新たな削除マーカーが追加されます。


Versioning-Enabled Bucket
Key=123.txt ID=113000 Delete Marke
Key=123.txt ID=112000
Key=123.txt ID=111000
Key=123.txt
DELETE the object Key=123.txt
Versioning-Enabled Bucket
Key=123.txt ID=114000 Delete Marke
Key=123.txt ID=113000 Delete Marke
Key=123.txt ID=112000
Key=123.txt ID=111000

バージョン管理を有効にしているバケットでは、新たに追加される削除マーカーは一意のバージョンIDを有しています。このため、1つのバケット内にある同一のオブジェクトが複数の削除マーカーを持つ可能性があります。「削除マーカー」を完全に削除したい場合は、DELETE Object versionIdリクエストに必ずそのバージョンIDを含めなければなりません。

下図のように、DELETE Object versionIdリクエストを実行して「削除マーカー」を完全に削除します。


Versioning-Enabled Bucket
Key=123.txt
ID=113000
Delete Maekr
Key=123.txt
ID=112000
Key=123.txt
ID=111000
Key=123.txt
ID=113000
Delete Maekr
DELETE the object Key=123.txt, ID3000
Versioning-Enabled Bucket
Key=123.txt
ID=112000
Key=123.txt
ID=111000

说明：

「削除マーカー」を削除できるのは、ルートアカウントによる権限承認を受けて `DeleteObject` 操作を行った場合のみです。

「削除マーカー」を完全に削除する手順は次のとおりです。

1. versionIdを削除マーカーのバージョンIDに設定します。
2. DELETE Object versionIdリクエストを送信します。

バージョン管理の使用

バージョン管理の設定

最終更新日：：2024-06-26 10:57:13

ユースケース

バージョン管理機能を利用すると、バケット内にオブジェクトの複数のバージョンを保存することができ、指定のバージョンのオブジェクトを検索、削除、復元することが可能になります。
バージョン管理についてより詳しい情報をお知りになりたい場合は、バージョン管理の概要のドキュメントをご参照ください。

注意：

バケットのバージョン管理状態を設定できるのはルートアカウントと権限を付与されたサブアカウントのみです。

使用方法

COSコンソールの使用

COSコンソールを使用してバージョン管理機能を有効化することができます。詳細については、バージョン管理の設定のコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用して、バケットのバージョン管理の設定およびバージョン管理状態にあるバケット内のオブジェクトの管理を行うことができます。次のAPIドキュメントのパートをご参照ください。

PUT Bucket versioning
GET Buket versioning
GET Bucket Object versions
PUT Object
GET Object
DELETE Object
DELETE Multiple Objects

SDKの使用

SDKのバージョン管理メソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

- C SDK
- C++ SDK
- .NET SDK
- Go SDK
- iOS SDK
- Java SDK
- JavaScript SDK
- Node.js SDK
- PHP SDK
- Python SDK
- Mini Program SDK

バケットコピー

バケットコピーの概要

最終更新日：：2024-06-26 10:57:13

概要

バケットコピーはオブジェクトのある設定を対象とし、バケットコピールールを設定することで、異なるバケット間で自動的、非同期的に**増分オブジェクト**のコピーを行うことができるものです。バケットコピーを有効にすると、COS（Cloud Object Storage）はソースバケット内のオブジェクトの内容（オブジェクトメタデータやバージョンIDなど）をターゲットバケットに正確にコピーし、コピーされたオブジェクトレプリカは完全に一致する属性情報を有します。また、ソースバケット内でオブジェクトに対して行う操作（オブジェクトの追加、削除などの操作）もターゲットバケットにコピーされます。

注意：

バケットのコピー機能を有効にするには、同時にソースバケットとターゲットバケットの両方でバージョン管理機能を有効にしておく必要があります。

バケットコピーを有効にする際は、データのコピー時にオブジェクトレプリカのストレージタイプを明確に指定した場合を除き、オブジェクトレプリカはソースオブジェクトと同じストレージタイプとなります。

COSはコピー時にソースバケットのアクセス制御リスト（ACL）をコピーします。現時点でCOSでは、異なる2つのアカウントのバケットをソースバケットとターゲットバケットとすることはできません。

ユースケース

リモートディザスタリカバリ：COSはオブジェクトデータに対し99.9999999999%の可用性をご提供していますが、戦争、自然災害などの様々な不可抗力要素によるデータ消失の可能性は存在します。データ消失に伴う損失が許容できないものであり、異なるバケットでデータレプリカを明示的に保全したい場合は、バケットコピーによってデータのリモートディザスタリカバリを実現できます。あるデータセンターが不可抗力要素によって破壊された場合でも、もう1つのバケットのデータセンターが使用可能なレプリカデータを提供することができます。

コンプライアンス要件：COSはデフォルトでは物理ディスク内で、マルチレプリカおよびイレージャーコーディング方式でデータの可用性を保障していますが、業界によってはコンプライアンス要件が存在し、異なるバケット間でデータレプリカを保存するよう規定されている場合があります。このため、バケットコピーを有効にすることで、異なるバケットのデータコピーを実現し、これらのコンプライアンス要件を満たすことができます。

アクセス遅延の減少：お客様の顧客が異なる地理的位置からオブジェクトにアクセスする際、バケットコピーによって、顧客の地理的位置に最も近いバケット内でオブジェクトレプリカを保全することで、顧客のアクセス遅延を最大限に短縮でき、製品体験の向上に役立てることができます。

操作上の理由：2つの異なるバケットのどちらにもコンピューティングクラスターがあり、かつこれらのコンピューティングクラスターが同じデータセットを処理する必要がある場合は、バケットコピーによってこれらの異なるバケット内でオブジェクトレプリカを保全することができます。
データマイグレーションおよびバックアップ：業務の発展による必要性に応じて、業務データをあるバケットから別のバケットにコピーすることで、データマイグレーションおよびデータバックアップを実現できます。

注意事項

コピー時間の制限

COSがオブジェクトのコピーに要する時間は、オブジェクトのサイズ、バケットのリージョン間の距離、オブジェクトのアップロード方式などの要素に左右されます。同期時間は上記の要素により異なり、数分間から数時間の幅があります。
オブジェクトのサイズ。大きなオブジェクトのコピーにはより多くの時間が必要です。大きなオブジェクトについてはマルチパートアップロード方式を利用することで、オブジェクトのアップロードおよび同期時間を短縮することをお勧めします。
バケットのリージョン間の距離。リージョン間の距離が遠いほど、同期の際にデータ転送時間がより長くかかります。
オブジェクトのアップロード方式。シンプルアップロード方式は同時実行ができず、1本の接続上でデータをシリアルにアップロードまたはダウンロードするだけですが、マルチパートアップロード方式では同時実行が可能なため、大容量ファイルのアップロードの際はマルチパートアップロードを使用することで、アップロードおよびバケットコピーの速度を速めることができます。オブジェクトのアップロード方式に関する詳細な説明については、シンプルアップロードおよびマルチパートアップロードのドキュメントをご参照ください。

ライフサイクル関連

バケットコピーにあたってはユーザーがバージョン管理機能を有効にしておく必要があります。バージョン管理機能は、バケット内にオブジェクトの複数の過去のバージョンを存在させるもので、ストレージを比較的多く消費します。COSバケットコピーの過程ではデータリクエスト料金、ダウンストリームトラフィックコストおよびデータストレージコストを消費します。このうちデータストレージコストはターゲットバケットの所在リージョンのストレージコスト価格に従います。バケットコピーおよびバージョン管理によるコストを削減したい場合、またはデータの保存方法をカスタマイズしたい場合は、業務の背景を踏まえて、ライフサイクルの管理によってストレージコスト抑制またはデータ保存方法のカスタマイズを実現することができます。
ターゲットバケット内のオブジェクトレプリカがソースバケット内と同様のライフサイクルルールに従うようにしたい場合は、ターゲットバケットにソースバケットと同一のライフサイクルルールを追加してください。
ターゲットバケットにライフサイクルルールを設定する場合、バケットコピー後のオブジェクトレプリカの作成時間は、それがターゲットバケット内に生成された時間ではなく、ソースバケット内での作成時間に対応することに注意する必要があります。

ソースバケットにライフサイクルルールを設定し、あるオブジェクトがバケットコピーを実行中に同時にライフサイクルルールによって削除される必要がある場合も、オブジェクトのバケットコピーは完了でき、ターゲットバケット内のオブジェクトレプリカも引き続き保存されます。

バージョン管理関連

バケットコピーの設定には、ユーザーがソースバケットとターゲットバケットの両方でバージョン管理機能を設定しておく必要があります。バージョン管理機能の詳細な内容については、バージョン管理の概要をご参照ください。バージョン管理をオンにした場合は、バージョン管理をオフにすることでバケットコピー機能に生じる影響に注意する必要があります。

バケットコピー機能を有効にしているバケット内でバージョン管理を無効にすることを試してみると、COSはエラーを返し、「先にバケットコピールールを削除してからバージョン管理を無効にする必要があります」とのメッセージを表示します。

あるターゲットバケット内でバージョン管理を無効にすることを試してみると、COSは、「バージョン管理をオフにするとバケットコピー機能に影響があります。バージョン管理を引き続きオフにする場合、このバケットをターゲットバケットとするCOSのバケットコピールールは失効します」とのメッセージを表示します。

コピーアクションの説明

最終更新日：：2024-06-26 10:57:13

このドキュメントでは主に、ユーザーがバケットのバケットコピー機能を有効にした場合に、COSがコピーするコンテンツとコピーしないコンテンツについてご説明します。

コピーするコンテンツ

バケットのコピー機能を有効にしたソースバケットにおいて、COSは次のコンテンツをコピーします。

バケットコピールールを追加した後に、ユーザーがソースバケットに新たにアップロードしたあらゆるオブジェクト。

オブジェクトのメタデータおよびバージョンIDなどのオブジェクトの属性情報。

オブジェクトの操作に関する情報。新たに追加された同名のオブジェクト（新規追加オブジェクト）、削除されたオブジェクトなど。

説明：

ソースバケット内で特定のオブジェクトのバージョンを削除するよう指定（バージョンIDを指定）した場合、その操作はコピーされません。

ソースバケットに、例えばライフサイクルルールのような、バケットレベルの設定を追加している場合、これらの設定によって発生したオブジェクトの操作もターゲットバケットにはコピーされません。

バケットコピーにおける削除操作

ソースバケットからオブジェクトを削除する場合、バケットのコピーアクションは次のとおりとなります。

オブジェクトのバージョンIDを指定せずにDELETEリクエストを実行した場合、COSはソースバケットに削除タグを追加します。同期削除タグを選択した場合、バケットコピーはこのタグをターゲットバケットにコピーします。非同期削除タグを選択した場合、ターゲットバケットは削除タグを新たに追加しません。どちらの状況でも、ターゲットバケットは対応するファイルを削除せず、ユーザーはバージョンIDを指定してオブジェクトの過去バージョンにアクセスすることができます。バージョン管理と削除タグの詳細情報については、バージョン管理の概要のドキュメントをご参照ください。

オブジェクトのバージョンIDを指定してDELETEリクエストを実行した場合、COSはソースバケット内の指定されたオブジェクトのバージョンを削除しますが、ターゲットバケットではこの削除操作をコピーしません。すなわち、COSがターゲットバケット内で指定されたオブジェクトのバージョンを削除することはありません。これにより悪意あるデータ削除を防止することができます。

コピーしないコンテンツ

ソースバケットがバケットコピー機能を有効にしている場合、COSは次のコンテンツをコピーしません。

バケットコピー機能を有効にする前にすでに存在したオブジェクトの内容、すなわち既存データ。

暗号化されたオブジェクトの暗号化情報。暗号化されたオブジェクトがコピーされると、暗号化情報は失われます。

ソースバケット内に新たに追加されたデータが、他のバケットからコピーされたオブジェクトデータの場合。

バケットレベルの設定更新アクション。

ライフサイクルの設定実行後の結果。

説明：

オブジェクトデータのバケット間でのバケットコピーは伝達性を有しません。例えば、バケットAをソースバケット、バケットBをターゲットバケットとするものと、バケットBをソースバケット、バケットCをターゲットバケットとする、2つのバケットコピールールを同時に設定したとします。この場合、バケットAに新たに追加されたオブジェクトデータはバケットBにのみコピーされます。そこからさらにバケットCにコピーされることはありません。

例えばライフサイクル設定の場合、ソースバケットのライフサイクル設定を更新しても、COSがこのライフサイクル設定をターゲットバケットに同期的に適用することはありません。

ソースバケットに対してのみライフサイクルルールを設定している場合、COSは期限切れのオブジェクトに削除タグを追加しますが、ターゲットバケットがこれらのタグをコピーすることはありません。ターゲットバケットで期限切れのオブジェクトを削除できるようにしたい場合は、ターゲットバケットに対し単独で、ソースバケットと同一のライフサイクルルールを設定する必要があります。

バケットコピーの設定

最終更新日：：2024-06-26 10:57:13

適用ケース

バケットコピールールを設定することで、ユーザーはオブジェクトデータをソースバケットから別の指定するターゲットバケットにコピーすることができます。バケットコピー機能は、リモート障害復旧、業界のコンプライアンス要件への適合、データマイグレーションおよびバックアップ、顧客アクセス遅延の低減、異なるリージョン間のクラスターのデータアクセス利便性向上などのケースに適しています。

特殊なケース：

マルチリージョンバックアップ：ソースバケットに複数のレプリケーションルールを設定し、オブジェクトを異なるリージョンのバケットにコピーすることで、マルチリージョナルなバックアップと障害復旧を実現できます。

双方向レプリケーション：ソースバケットとターゲットバケットでそれぞれレプリケーションルールを作成することにより、2つのバケット間での双方向レプリケーションを実現し、バケットデータの同期を実現することができます。

注意：

バージョン管理機能を有効にしている場合、新たにアップロードするオブジェクトには複数のバージョンが生成され、ストレージスペースを占有します。このためこれらのバージョンのオブジェクトも同様にストレージ料金の課金対象となります。

使用方法

COSコンソールの使用

COSコンソールでバケットコピールールを設定することができます。バケットコピーの設定のコンソールガイドドキュメントをご参照ください。

REST APIの使用

REST APIを直接使用して、バケットのバケットコピールールの設定と管理を行うことができます。具体的には次のAPIドキュメントをご参照ください。

PUT Bucket replication

GET Bucket replication

DELETE Bucket replication

SDKの使用

SDKのバケットコピーメソッドを直接呼び出すことができます。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK

C SDK

C++ SDK

.NET SDK

Go SDK

iOS SDK

Java SDK

JavaScript SDK

Node.js SDK

PHP SDK

Python SDK

マルチAZ特性の概要

最終更新日：：2024-06-26 10:57:13

マルチAZ（Available Zone）とはTencent Cloud COSがリリースしたマルチAZストレージアーキテクチャです。このストレージアーキテクチャによってユーザーデータにデータセンターレベルの障害復旧機能を提供することが可能です。

お客様のデータは都市の複数の異なるデータセンターに分散して保存されます。あるデータセンターに自然災害、停電などの極端な状況による全面的な障害が発生した場合でも、マルチAZストレージアーキテクチャによって安定した信頼性の高いストレージサービスを引き続き提供することができます。

マルチAZ特性により、設計上のデータ信頼性は99.9999999999%（トゥエルブナイン）に、設計上のサービス可用性は99.995%に達します。データをCOSにアップロードする際にオブジェクトのストレージタイプを指定するだけで、オブジェクトをマルチAZのリージョンに保存することができます。

説明：

COSのマルチAZ特性は現時点では北京、広州、上海、中国香港、シンガポール。その他のパブリッククラウドリージョンでも順次サポート予定です。

COSのマルチAZ特性を使用する場合、ストレージ容量料金が相対的に高くなります。詳細については、製品価格をご参照ください。

マルチAZのメリット

データをマルチAZリージョンに保存すると、データがいくつかのパートに分割され、同時にイレイジャーコーディングアルゴリズムに従って対応するチェックコードパートが算出されます。オリジナルのデータパートとチェックコードパートは分割されてそのリージョンの異なるデータセンターに均等に保存され、同一都市内障害復旧が可能になります。あるデータセンターが利用できなくなった場合も、別のデータセンターのデータは正常に読み取りと書き込みができるため、お客様のデータが消失することなく永続的に保存され、お客様の業務におけるデータ連続性と高可用性が維持できます。COSのマルチAZを使用すると次のようなメリットがあります。

同一都市内障害復旧：異なるデータセンター間での障害復旧が可能です。マルチAZストレージアーキテクチャでは、オブジェクトデータは同一リージョンの異なるデータセンターの異なるデバイス内に保存されます。データセンターのうち1か所に障害が発生した際、冗長データセンターは引き続き利用できるため、ユーザーの業務は影響を受けず、データも消失しません。

安定性・永続性：イレイジャーコーディングによる冗長ストレージ方式を採用し、99.9999999999%の設計上のデータ信頼性を実現しています。また、データの分割ストレージ、読み取り/書き込み同時実行により、設計上のサービス可用性は99.995%に達しています。

使いやすさ：オブジェクトのストレージタイプによって、データをどのストレージアーキテクチャに保存するかを指定します。バケット内の任意のオブジェクトを指定してマルチAZアーキテクチャに保存することができ、より簡単に使用できます。

マルチAZストレージと非マルチAZストレージの仕様面での制限の比較は次のとおりです。

比較項目	マルチAZストレージ	非マルチAZストレージ
設計上のデータ永続性	99.9999999999% （トゥエルブナイン）	99.999999999%（イレブンナイン）
設計上のサービス可用性	99.995%	99.99%
サポートするリージョン	ストレージタイプの概要のドキュメントをご参照ください	
サポートするストレージタイプ	標準ストレージ（マルチAZ）（MAZ_STANDARD）低頻度ストレージ（マルチAZ）（MAZ_STANDARD_IA）INTELLIGENT_TIERINGストレージ（マルチAZ）（MAZ_INTELLIGENT_TIERING）	標準ストレージ（STANDARD）低頻度ストレージ（STANDARD_IA）アーカイブストレージ（ARCHIVE）ディープアーカイブストレージ（DEEP_ARCHIVE）INTELLIGENT_TIERINGストレージ（INTELLIGENT_TIERING）

利用方法

ユーザーはバケットのマルチAZ設定を有効化することができます。マルチAZ設定を有効化したバケットにオブジェクトをアップロードすると、オブジェクトのストレージタイプをマルチAZに設定できます。ユーザーがオブジェクトをアップロードする際にオブジェクトのストレージタイプを指定すると、オブジェクトをマルチAZストレージアーキテクチャ内に保存することができます。

簡潔に言えば、次の2つの手順を実行するだけで、ファイルをマルチAZアーキテクチャ内に保存することができます。

1. バケットを作成し、**作成の際に**マルチAZ設定を有効化します。バケットの作成ガイドについては、バケットの作成のドキュメントをご参照ください。
2. ファイルをアップロードし、その際にファイルのストレージタイプを指定します。ファイルのアップロードガイドについては、オブジェクトのアップロードのドキュメントをご参照ください。

説明：

バケットのマルチAZ設定は有効化すると変更できませんので、慎重に設定してください。作成済みのバケットはデフォルトでマルチAZ設定が有効化されていません。有効化できるのは新しく作成したバケットのみとなります。

マルチAZ設定を有効化したバケットには、標準ストレージ（マルチAZ）、低頻度ストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ（マルチAZ）タイプをアップロードすることができます。このうち、INTELLIGENT_TIERINGストレージ（マルチAZ）タイプをアップロードする場合は、バケットで同時にマルチAZ設定およびINTELLIGENT_TIERING設定を有効化する必要があります。

既存のデータをマルチAZバケットに保存したい場合は、マルチAZ設定を有効化したバケットを新規作成し、同時にCOS Batchの一括コピー機能を使用し、既存のバケット内のファイルを新規バケットに一括コピーすることができます。COS Batchの使用ガイドについては、バッチ処理の操作ドキュメントをご参照ください。

使用制限

現在COSは、標準ストレージ（マルチAZ）、低頻度ストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ（マルチAZ）タイプのアップロードをサポートしています。このため、ストレージタイプの変更を伴う関連の機能には同様に制限が存在します。関連機能の制限についての説明は次のとおりです。

ストレージタイプの制限：現在は標準ストレージ（マルチAZ）、低頻度ストレージ（マルチAZ）、INTELLIGENT_TIERINGストレージ（マルチAZ）タイプのアップロードのみをサポートしています。このうち、INTELLIGENT_TIERINGストレージ（マルチAZ）タイプをアップロードする場合は、バケットで同時にマルチAZ設定およびINTELLIGENT_TIERING設定を有効化する必要があります。

操作の制限：現在はオブジェクトのアップロード、ダウンロードおよび削除操作のみサポートしています。オブジェクトはマルチAZのバケットへのコピーのみをサポートしており、シングルAZのバケットへのコピーはサポートしていません。

ライフサイクルの制限：現在は期限切れとなったオブジェクトの削除のみをサポートしています。マルチAZストレージタイプからシングルAZストレージタイプへの移行はサポートしていません。

クロスリージョンレプリケーションの制限：マルチAZストレージタイプをシングルAZストレージタイプにコピーすることはサポートされていません。

Data Security

サーバー側の暗号化の概要

最終更新日：：2024-06-26 10:57:13

概要

Cloud Object Storage（COS）はデータをデータセンター内のディスクに書き込む前に、オブジェクトレベルでデータの暗号化を適用する保護ポリシーをサポートしています。データはアクセス時に自動的に復号されます。暗号化と復号の操作プロセスはサーバー側で完了します。このサーバー側での暗号化機能によって静的データを有効に保護することができます。

注意：

暗号化されたオブジェクトと暗号化されていないオブジェクトへのアクセスに体験上の違いはありません。ただしユーザーがオブジェクトへのアクセス権限を持っていることが前提です。

サーバー側での暗号化はオブジェクトデータのみを暗号化するものであり、オブジェクトメタデータの暗号化は行いません。またサーバー側で暗号化したオブジェクトには有効な署名を使用してアクセスする必要があり、匿名ユーザーはアクセスできなくなります。

ユースケース

プライベートデータストレージのケース：プライベートデータのストレージについては、サーバー側での暗号化は、保存されているデータに対して行うことができます。ユーザーのプライバシーは保証され、ユーザーがアクセスした際は自動的に復号されます。

プライベートデータ転送のケース：プライベートデータの転送については、COSはHTTPSを使用してデプロイしたSSL証明書を提供して暗号化機能を実現します。転送リンクレイヤー上に暗号化レイヤーを設け、データの転送過程でのハッキングや改ざんを確実に防止します。

暗号化方式

COSがサポートしているサーバー側の暗号化方式はSSE-COS、SSE-KMS、SSE-Cです。ユーザーはご自身に合った暗号化方式を選択し、COSに保存したデータの暗号化を行うことができます。

SSE-COSの暗号化

SSE-COS暗号化は、COSホストキーによるサーバー側の暗号化です。Tencent Cloud COSはマスターキーをホストし、データを管理します。ユーザーはCOSによってデータの管理と暗号化を直接行います。SSE-COSは多要素

による強力な暗号化を採用し、一意のキーを使用して各オブジェクトを暗号化し、同時に256ビットの高度な暗号化（AES-256）を用いてデータを暗号化するほか、マスターキーの定期的なローテーションによってキー自体の暗号化を行うことができます。

注意：

`POST` 操作を使用してオブジェクトのアップロードを行う際は、フォームのフィールドで、 `x-cos-server-side-encryption` ヘッダーではなく、同一の情報を提供する必要があります。詳細については、POST Objectをご参照ください。

署名付きURLを使用してアップロードしたオブジェクトについては、SSE-COS暗号化を使用できません。COSコンソールまたはHTTPリクエストヘッダーを使用してサーバー側での暗号化を指定することのみ可能です。

COSコンソールの使用

コンソール上でオブジェクトのSSE-COS暗号化を行う方法について詳しくお知りになりたい場合は、オブジェクト暗号化の設定コンソールドキュメントをご参照ください。

REST APIの使用

注意：

バケット内のオブジェクトをリストアップする際、オブジェクトが暗号化されているかどうかにかかわらず、すべてのオブジェクトのリストが返されます。

POSTを使用してオブジェクトのアップロード操作を行う際は、フォームのフィールドで、このリクエストヘッダーではなく、同一の情報を提供してください。詳細については、POST Objectをご参照ください。

ユーザーが次のインターフェースをリクエストする場合は、 `x-cos-server-side-encryption` ヘッダーを提供することでサーバー側の暗号化を適用することができます。詳細については、パブリックリクエストヘッダー - SSE-COSをご参照ください。

PUT Object

Initiate Multipart Upload

PUT Object - Copy

POST Object

SSE-KMSの暗号化

SSE-KMS暗号化は、KMSホストキーによるサーバー側の暗号化です。KMSはTencent Cloudが推進するセキュリティ管理系サービスの1つであり、サードパーティ認証を経たハードウェアセキュリティモジュールHSM（Hardware Security Module）を使用してキーの生成と保護を行うものです。ユーザーがキーの作成と管理を手軽に行えるよう支援し、複数のアプリケーションや複数の業務でのキー管理のニーズを満たすとともに、規制とコンプライアンスの要件にも適合します。

SSE-KMS暗号化を初めて使用する際は、KMSサービスのアクティブ化を行う必要があります。KMSサービスをアクティブ化すると、システムは自動的にデフォルトのマスターキー（CMK）1個を作成します。またはKMSコンソールで自らキーを作成し、キーのポリシーと使用方法を定義することもできます。KMSでは、ユーザーがキー

素材のソースを**KMS**または**外部**から自ら選択することができます。その他の情報については、キーの作成および外部キーのインポートをご参照ください。

注意：

SSE-KMSはオブジェクトデータのみを暗号化し、オブジェクトメタデータは一切暗号化しません。

SSE-KMSは現在、北京、上海、中国香港、広州リージョンのみサポートしています。

SSE-KMS暗号化の使用には別途料金が発生し、KMSによって課金されます。詳細については、KMS課金概要をご参照ください。

SSE-KMS暗号化を使用したオブジェクトには有効な署名を使用してアクセスする必要があり、匿名ユーザーはアクセスできなくなります。

COSコンソールの使用

コンソール上でオブジェクトのSSE-KMS暗号化を行う方法について詳しくお知りになりたい場合は、オブジェクト暗号化の設定コンソールドキュメントをご参照ください。

REST APIの使用

注意：

バケット内のオブジェクトをリストアップする際、オブジェクトが暗号化されているかどうかにかかわらず、すべてのオブジェクトのリストが返されます。

POSTを使用してオブジェクトのアップロード操作を行う際は、フォームのフィールドで、このリクエストヘッダーではなく、同一の情報を提供してください。詳細については、POST Objectをご参照ください。

ユーザーが次のインターフェースをリクエストする場合は、`x-cos-server-side-encryption` ヘッダーを提供することでサーバー側の暗号化を適用することができます。詳細については、パブリックリクエストヘッダー - SSE-KMSをご参照ください。

PUT Object

Initiate Multipart Upload

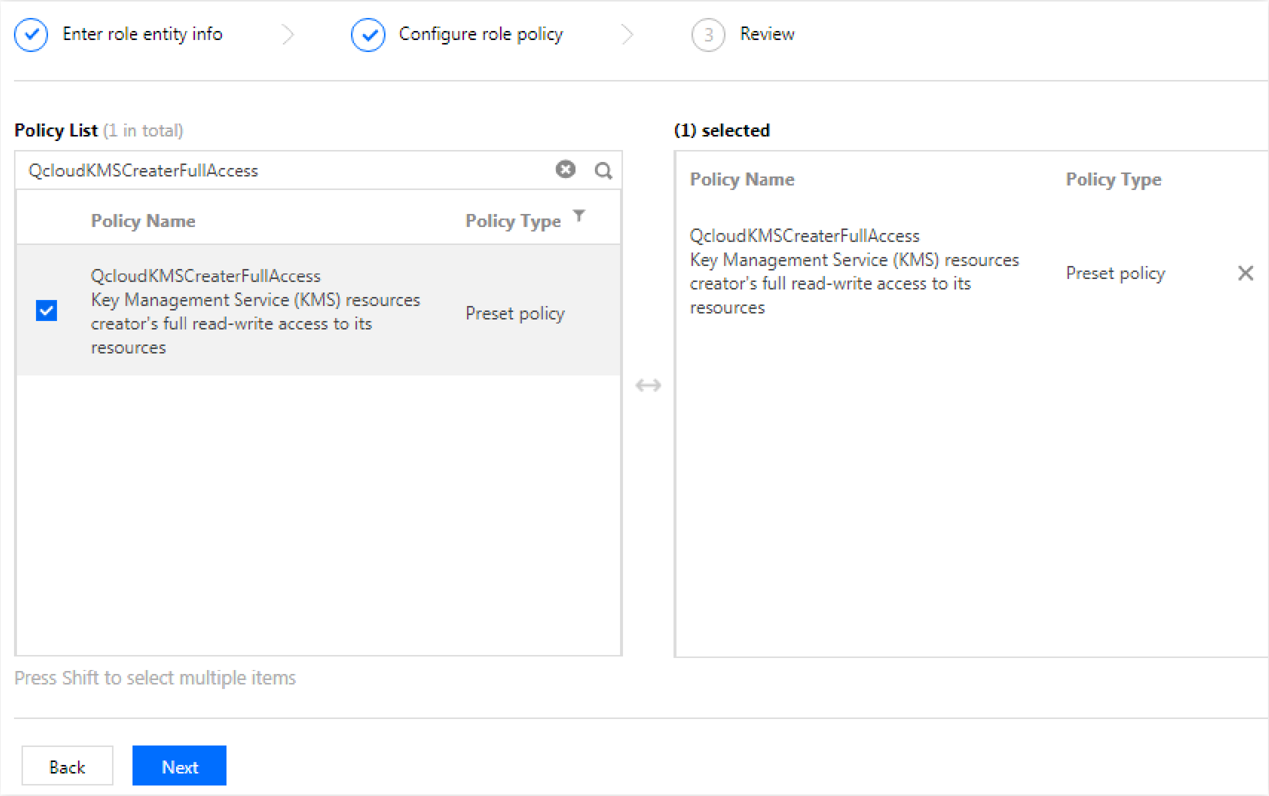
PUT Object - Copy

POST Object

注意事项

COSコンソールを使用してSSE-KMS暗号化を行ったことがなく、**API**方式のみを使用してSSE-KMS暗号化を行っている場合は、まずCAMロールを作成する必要があります。具体的な作成手順は次のとおりです。

1. CAMコンソールにログインし、ロールリストページに進みます。
2. **ロールの新規作成**をクリックし、**Tencent Cloud製品サービス**をロールエンティティとして選択します。
3. ロールをサポートするサービスは**COS**を選択し、**次のステップ**をクリックします。
4. ロールポリシーを設定します。**QcloudKMSAccessForCOSRole**を検索してチェックを入れ、**次のステップ**をクリックします。



Policy List (1 in total)

QcloudKMSCreaterFullAccess

	Policy Name	Policy Type
☑	QcloudKMSCreaterFullAccess Key Management Service (KMS) resources creator's full read-write access to its resources	Preset policy

(1) selected

Policy Name	Policy Type	
QcloudKMSCreaterFullAccess Key Management Service (KMS) resources creator's full read-write access to its resources	Preset policy	×

5. ロールのタグキーとタグ値をマークし、**次のステップ**をクリックします。
6. 指定のロール名：COS_QcsRoleを入力します。
7. 最後に**完了**をクリックすれば作成は終了です。

SSE-Cの暗号化

SSE-C暗号化は、ユーザー定義キーによるサーバー側の暗号化です。暗号化鍵はユーザーが提供し、COSはユーザーがオブジェクトをアップロードする際に、ユーザーが提供した暗号化鍵のキーペアを使用して、ユーザーのデータをAES-256で暗号化します。

注意：

COSはユーザーが提供した暗号化鍵を保存せず、暗号化鍵にランダムデータを追加したHMAC値を保存します。この値はオブジェクトにアクセスするためのユーザーのリクエストの検証に使用されます。COSは、ランダムデータのHMAC値を使用して暗号化鍵の値を推測したり、暗号化されたオブジェクトの内容を復号したりすることはできません。そのため、ユーザーが暗号化鍵を紛失した場合、このオブジェクトを再取得できなくなります。

POST操作を使用してオブジェクトのアップロードを行う際は、フォームのフィールドで、`x-cos-server-side-encryption-*` ヘッダーではなく、同一の情報を提供する必要があります。詳細については、POST Objectをご参照ください。

SSE-CはAPIでしか使用できず、コンソールでの操作はサポートされていません。

REST APIの使用

ユーザーが次のインターフェースをリクエストする場合は、PUTおよびPOSTリクエストについては `x-cos-server-side-encryption-*` ヘッダーを提供することでサーバー側の暗号化を適用することができます。GETおよびHEADでSSE-C暗号化を使用したオブジェクトをリクエストする場合は、 `x-cos-server-side-encryption-*` ヘッダーを提供して指定のオブジェクトの復号を行う必要があります。詳細については、パブリックリクエストヘッダー - SSE-Cをご参照ください。このヘッダーをサポートしている操作は次のとおりです。

GET Object
HEAD Object
PUT Object
Initiate Multipart Upload
Upload Part
POST Object
PUT Object - Copy

バケット暗号化の概要

最終更新日：：2024-06-26 10:57:13

概要

バケット暗号化はバケットに対する設定の1つであり、バケット暗号化を設定することで、新たにバケットにアップロードされたすべてのオブジェクトに対し、デフォルトで、指定された暗号化方式で暗号化を行うことができます。

現在はSSE-COS暗号化をサポートしています。これはCloud Object Storage（COS）ホストキーを使用したサーバー側の暗号化です。

サーバー側の暗号化に関するその他の情報については、サーバー側の暗号化の概要をご参照ください。

使用方法

COSコンソールの使用

COSコンソールを使用してバケットの暗号化を設定することができます。詳細については、バケット暗号化の設定のコンソールガイドドキュメントをご参照ください。

REST APIの使用

次のAPIによってバケットの暗号化を設定できます。

PUT Bucket encryption

GET Bucket encryption

DELETE Bucket encryption

注意事項

暗号化されたバケットへのオブジェクトのアップロード

バケット暗号化機能を設定したいバケットについては、次の数点に注意が必要です。

バケット暗号化では、バケット内にすでに存在するオブジェクトに対する暗号化操作は行いません。

バケット暗号化を設定した後、このバケットにアップロードされるオブジェクトは次のようになります。

PUTリクエストに暗号化情報が含まれていない場合、アップロードされるオブジェクトはバケットの暗号化設定によって暗号化されます。

PUTリクエストに暗号化情報が含まれている場合、アップロードされるオブジェクトはPUTリクエスト内の暗号化情報によって暗号化されます。

バケット暗号化を設定した後、このバケットに送信されるリストレポートは次のようになります。

リスト自体に暗号化が設定されていない場合、送信されるリストはバケットの暗号化設定によって暗号化されます。

リスト自体に暗号化が設定されている場合、送信されるリストはリストの暗号化設定によって暗号化されます。

バケット暗号化を設定した後、このバケットにback-to-originされるデータは、デフォルトでバケットの暗号化設定によって暗号化されます。

クロスリージョンレプリケーションルールが設定されているバケットに対する暗号化

クロスリージョンレプリケーションルールが設定されているターゲットバケットに対し、さらにバケット暗号化を設定する場合は、次の数点に注意が必要です。

ソースバケット内のオブジェクトが暗号化されていない場合、ターゲットバケット内のレプリカオブジェクトに対してはデフォルトで暗号化が設定されます。

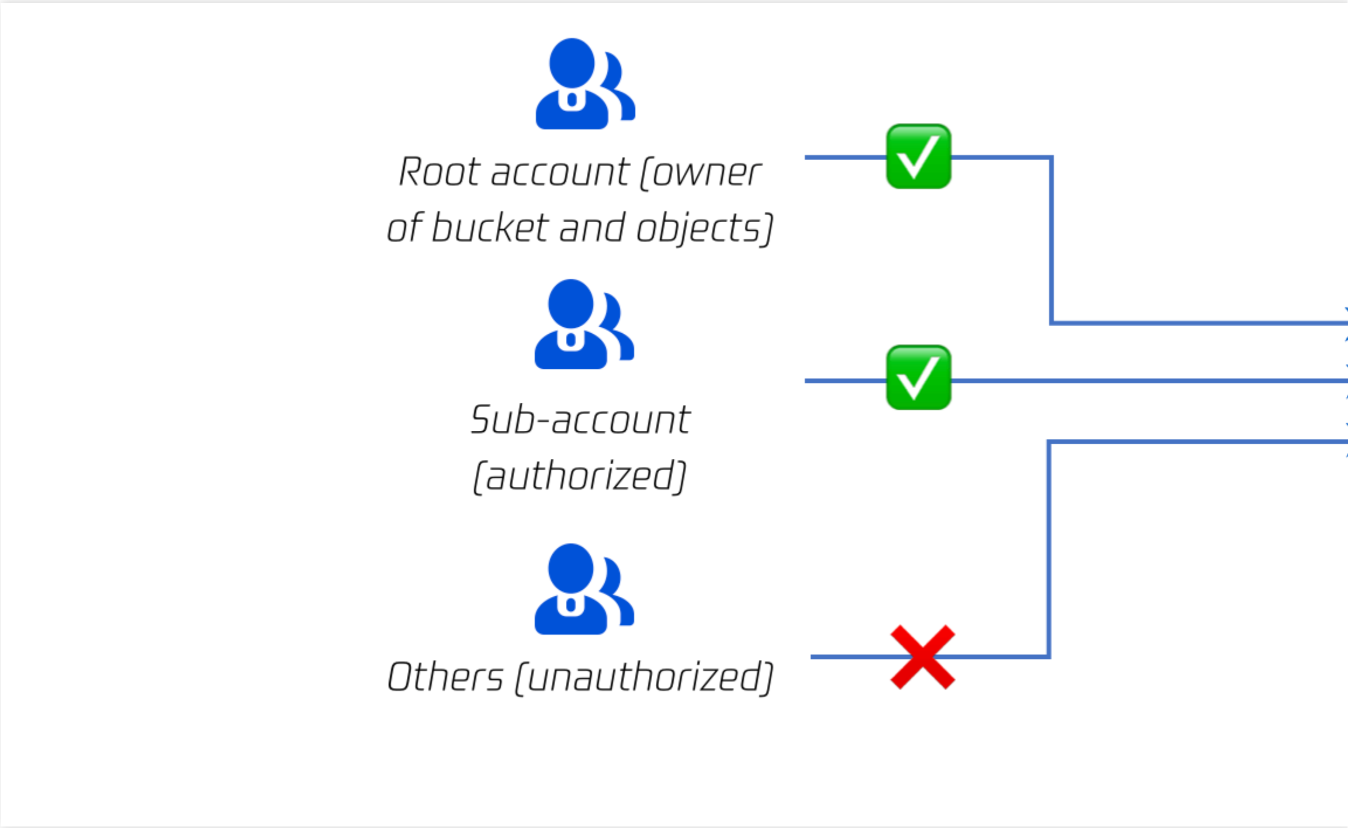
ソースバケット内のオブジェクトが暗号化されている場合、ターゲットバケット内のレプリカオブジェクトはソースバケットの暗号化を継承するため、バケット暗号化設定は実行されません。

クラウドアクセスマネジメント
アクセス権限設定の説明
アクセス管理の概要
アクセス制御の基本概念

最終更新日：：2024-06-26 11:09:29

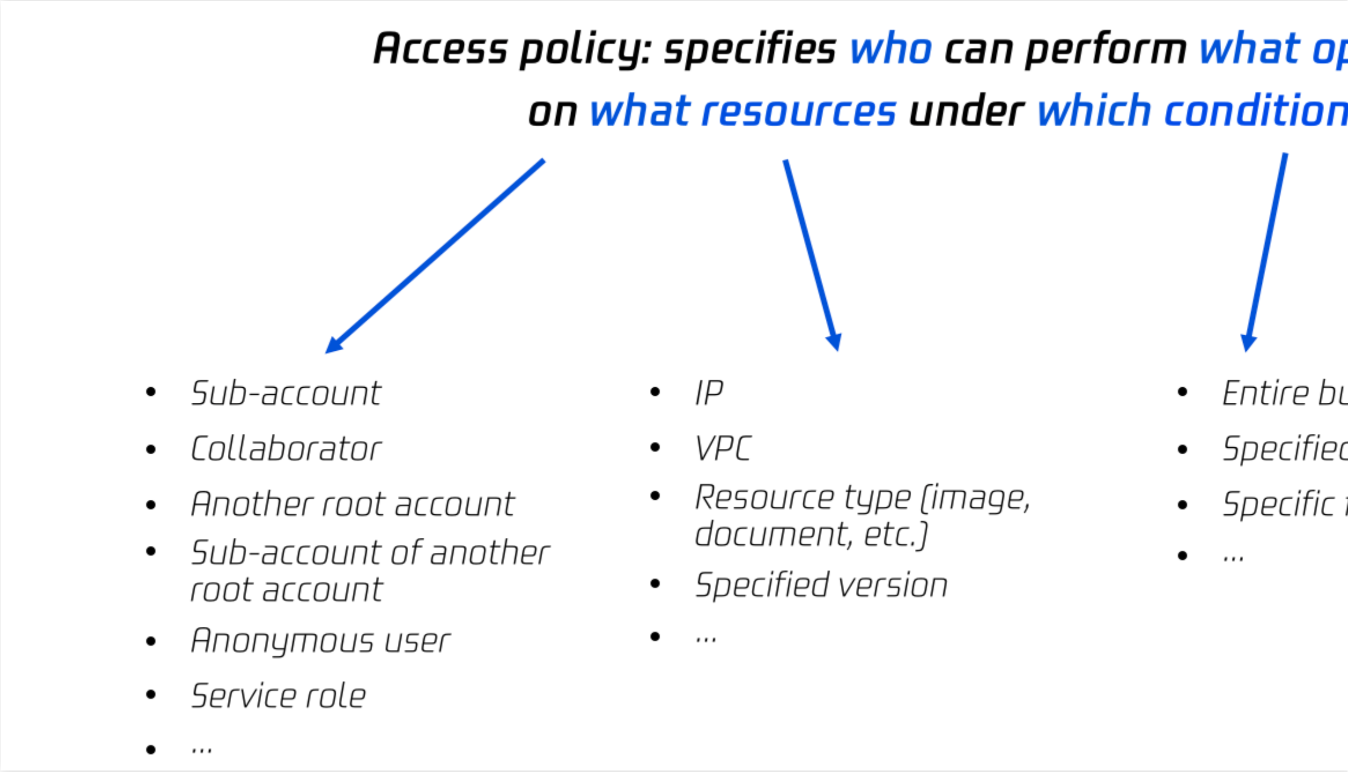
デフォルトでは、Cloud Object Storage（COS）のリソース（バケットおよびオブジェクト）はすべてプライベートです。Tencent Cloudルートアカウント（リソース所有者）でなければバケットおよびオブジェクトへのアクセス、変更を行うことはできず、他のユーザー（サブアカウント、匿名ユーザーなど）はいずれも権限承認がなければURLから直接オブジェクトにアクセスすることはできません。

Tencent Cloudサブアカウントを作成すると、アクセスポリシーによってサブアカウントに権限を付与することができます。非Tencent Cloudユーザーにリソースを開放したい場合は、リソース（バケット、オブジェクト、ディレクトリ）のパブリック権限（パブリック読み取り）を設定することで実現できます。


Root account (owner of bucket and objects)
✓
Sub-account (authorized)
✓
Others (unauthorized)
✗

アクセス制御の要素

アクセス権限の付与とは、どの人が、どのような条件下で、どのリソースに対して、具体的な操作を行うかという制御機能の組み合わせをユーザーが決定できることを指します。このため、1つのアクセス権限行為の記述には、通常**プリンシパル、リソース、操作、条件（オプション）**の4つの要素が含まれます。


Access policy: specifies who can perform what op
on what resources under which condition
Sub-account
Collaborator
Another root account
Sub-account of another root account
Anonymous user
Service role
...
IP
VPC
Resource type (image, document, etc.)
Specified version
...
Entire bu
Specifie
Specific
...

アクセス権限の要素

Tencent Cloudのプリンシパル（Principal）

ユーザーがTencent Cloudアカウントを申請する際、システムはTencent Cloudサービスへのログインに用いるルートアカウントのIDを作成します。Tencent Cloudルートアカウントはユーザー管理機能によって、異なる職責を持つユーザーを分類し管理します。ユーザーのタイプには**コラボレーター、メッセージ受信者、サブユーザーおよびロール**などがあります。具体的な定義についてはCAMのユーザーのタイプおよび用語集のドキュメントをご参照ください。

説明：

社内のある同僚への権限承認を行いたい場合は、まずCAMコンソールでサブユーザーを作成し、バケットポリシー、ACLまたはユーザーポリシーのいずれかまたは複数の手段を選択し、サブユーザーの具体的な権限を設定する必要があります。

COSのリソース（Resource）

BucketおよびObjectはCOSの基本リソースです。そのうちフォルダは特殊なオブジェクトであり、フォルダを通じてフォルダ下のオブジェクトの権限承認を行うことができます。詳細については、フォルダの権限の設定をご参照ください。

また、バケットとオブジェクトにはどちらもそれらに関連するサブリソースが存在します。

バケットのサブリソースには次のものが含まれます。

aclおよびpolicy：バケットのアクセス制御情報です。

website：バケットの静的ウェブサイトホスティング設定です。

tagging：バケットのタグ情報です。

cors：バケットのクロスドメイン設定情報です。

lifecycle：バケットのライフサイクル設定情報です。

オブジェクトのサブリソースには次のものが含まれます。

acl：オブジェクトのアクセス制御情報です。

restore：アーカイブタイプのオブジェクトの復元設定です。

COSのアクション（Action）

COSではリソースに対する一連のAPI操作をご提供しています。詳細については、操作リストのドキュメントをご参照ください。

COSの条件（Condition、オプション）

オプションです。vpc、vipなどの権限発効の条件についての詳細は、CAMの発効条件をご参照ください。

プライベートの原則

説明：

デフォルトでは、Tencent Cloud COS内のリソースはすべてプライベートです。

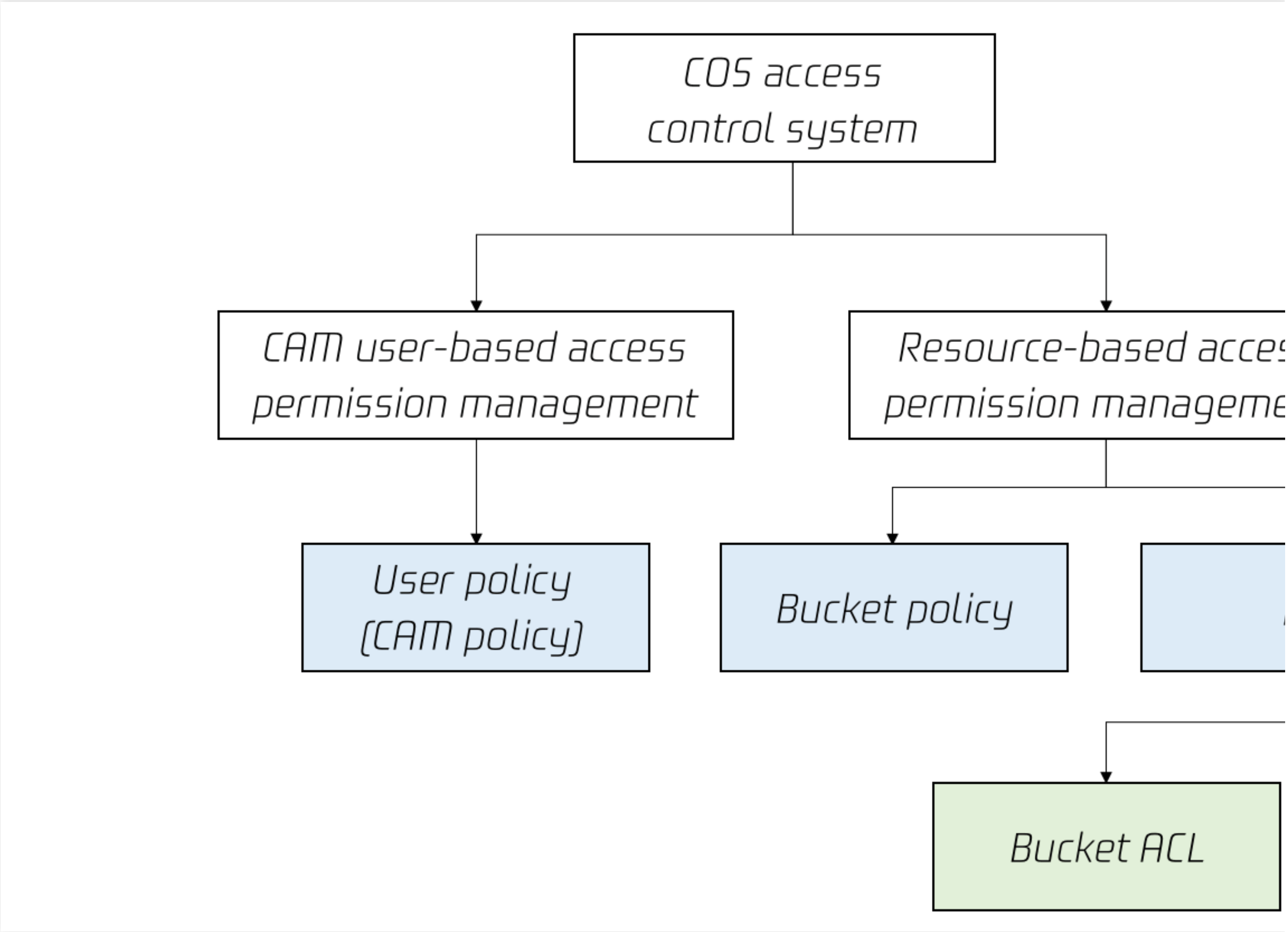
リソース所有者（バケットのリソースを作成したTencent Cloudルートアカウント）はこのリソースに対する最高権限を有します。リソース所有者はアクセスポリシーを編集および変更することができ、第三者または匿名ユーザーに対しアクセス権限を与えることができます。

Tencent CloudのCAM（Cloud Access Management）アカウントを使用してバケット作成またはオブジェクトのアップロードを行う場合、親アカウントにあたるルートアカウントがリソース所有者となります。

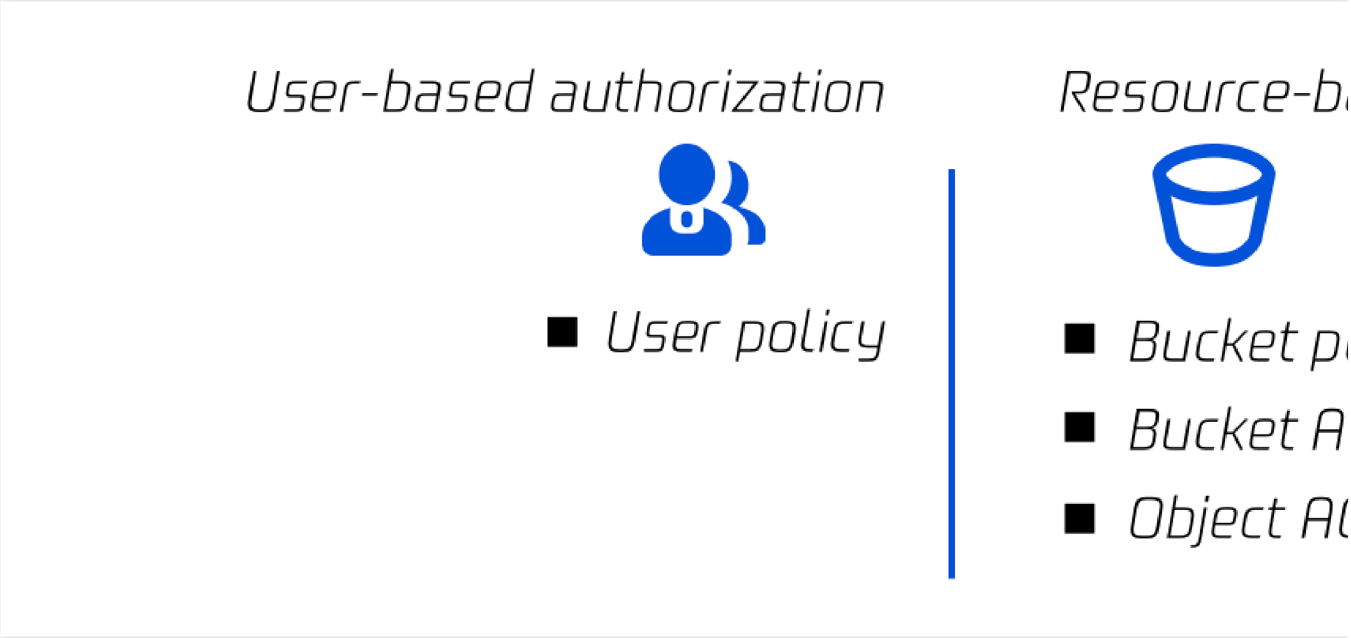
バケット所有者のルートアカウントは他のTencent Cloudルートアカウントに対し、オブジェクトのアップロード権限を付与することができます（クロスアカウントアップロード）。この場合、オブジェクトの所有者は引き続きバケット所有者のルートアカウントとなります。

アクセス制御の複数の手段

COSは複数の権限設定方式をご提供してアクセス制御を実現しています。これにはバケットポリシー、ユーザーポリシー（CAMポリシー）、バケットACLおよびオブジェクトACLが含まれます。
これらはポリシー設定の出発点に基づき、リソースベースとユーザーベースの2種類の方式に分けられます。また権限承認の方式に基づき、ポリシーとACLの2種類の方式に分けられます。


COS access control system
CAM user-based access permission management
Resource-based acce
permission manageme
User policy (CAM policy)
Bucket policy
Bucket ACL

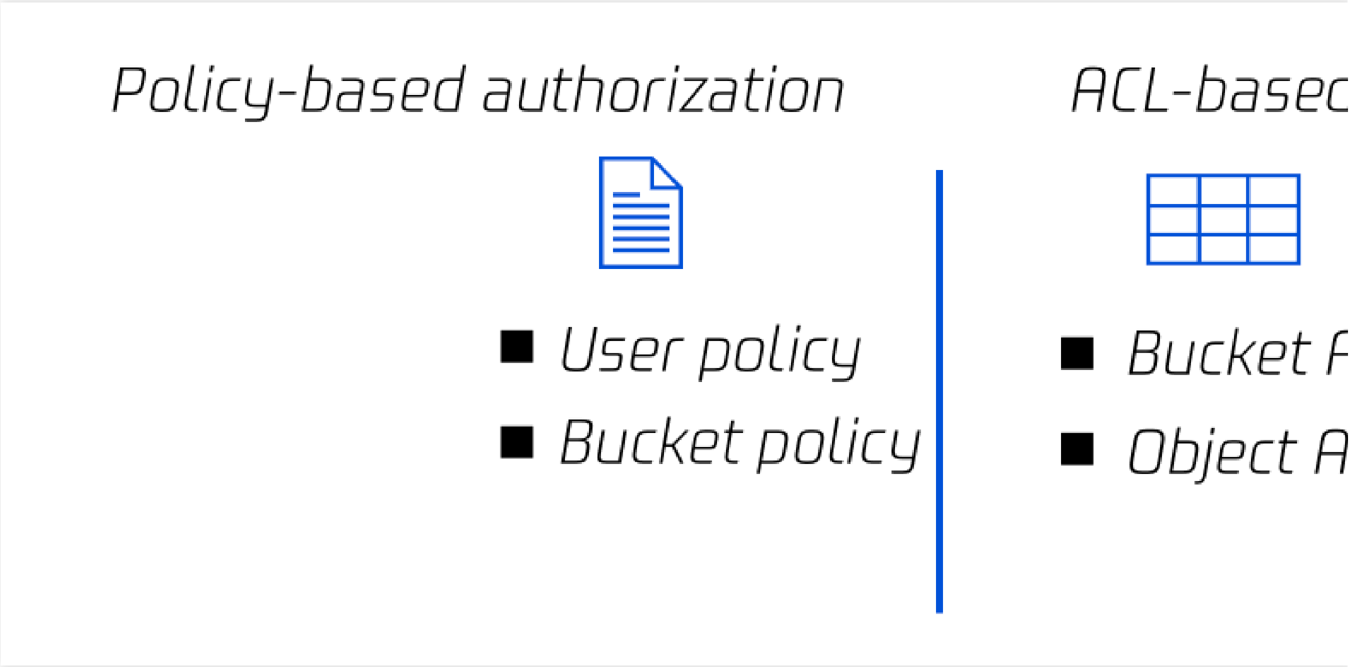
分類方法1：リソースベース vs ユーザーベース


User-based authorization
User policy
Resource-b
Bucket p
Bucket A
Object A

リソースを出発点とする場合：権限を具体的なリソースにバインドします。バケットポリシー、バケットACLおよびオブジェクトACLが含まれ、COSコンソールまたはCOS APIによって設定します。

ユーザーを出発点とする場合：ユーザーポリシー（CAMポリシー）では権限をユーザーにバインドします。ポリシー作成時にユーザーを入力する必要はなく、リソース、アクション、条件などを指定し、CAMコンソールで設定します。

分類方法2：ポリシー vs ACL


Policy-based authorization
User policy
Bucket policy
ACL-base
Bucket
Object A

ポリシー：ユーザーポリシー（CAMポリシー）とバケットポリシーはどちらも完全なポリシー構文に基づいて権限承認を行うものです。権限承認の動作は各APIの対応する動作に細分化され、許可/ 拒否のエフェクトを指定す

ることができます。
ACL：バケットACLとオブジェクトACLはどちらもアクセス制御リスト（ACL）をベースにして実装されます。ACLはリソースにバインドされた、被付与者と付与される権限を指定したリストです。整理され抽象化された権限に対応し、許可のエフェクトのみを指定することができます。

リソースベースのポリシー

リソースベースのポリシーにはバケットポリシー、バケットACL、オブジェクトACLの3種類が含まれます。**バケット**と**オブジェクト**の次元でそれぞれアクセス制御を行うことができます。具体的には次の表をご参照ください。

次元	タイプ	記述方式	サポートするプリンシパル	サポートするリソース粒度	サポートするアクション	サポートするエフェクト
Bucket	アクセスポリシー言語（Policy）	JSON	サブアカウント、ロール、Tencent Cloudサービス、他のルートアカウント、匿名ユーザーなど	バケット、オブジェクト、プレフィックスなど	それぞれの具体的な操作	許可/明示的な拒否
Bucket	アクセス制御リスト（ACL）	XML	他のルートアカウント、サブアカウント、匿名ユーザー	バケット	整理された読み取り/書き込み権限	許可のみ
Object	アクセス制御リスト（ACL）	XML	他のルートアカウント、サブアカウント、匿名ユーザー	オブジェクト	整理された読み取り/書き込み権限	許可のみ

バケットポリシー（Bucket Policy）

バケットポリシー（Bucket Policy）はJSON言語を使用して記述され、匿名IDまたはTencent CloudのあらゆるCAMアカウントに対し、バケット、バケット操作、オブジェクトまたはオブジェクト操作への権限付与をサポートしています。Tencent Cloud COSのバケットポリシーは、そのバケット内のほとんどすべての操作の管理に用いることができます。ACLでは記述できないアクセスポリシーを、バケットポリシーを使用して管理することをお勧めします。その他の内容についてはバケットポリシーのドキュメントをご参照ください。

注意：

Tencent Cloudのルートアカウントは、そのアカウント下のリソース（バケットを含む）に対する最大の権限を有しています。バケットポリシーではほとんどすべての操作を制限できますが、ルートアカウントは常にPUT

Bucket Policy操作の権限を有しており、ルートアカウントがこの操作を呼び出す際、バケットポリシーのチェックは行われません。

以下は、匿名ユーザーに対し広州にあるバケットexamplebucket-1250000000内のすべてのオブジェクトへのアクセスを許可するポリシーです。署名のチェックは必要なく、そのままバケット内のすべてのオブジェクトをダウンロード（GetObject）できます。すなわち、URLを知っている匿名ユーザーは誰でもオブジェクトをダウンロードできます（パブリック読み取り方式に類似）。

```
{
  "Statement": [
    {
        "Principal": "*",
        "Effect": "Allow",
        "Action": ["cos:GetObject"],
        "Resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*
      }
    ],
    "Version": "2.0"
  }
```

アクセス制御リスト（ACL）

アクセス制御リスト（ACL）はXML言語を使用して記述する、リソースにバインドされた、被付与者と付与される権限を指定したリストです。各バケットとオブジェクトにはそれぞれに、これらとバインドされたACLがあり、匿名ユーザーまたはその他のTencent Cloudのルートアカウントに対し、基本的な読み取り/書き込み権限を付与することができます。その他の内容については、ACLのドキュメントをご参照ください。

注意：

発行されたACLにその記述があるかどうかにかかわらず、リソースの所有者は常にリソースに対してFULL_CONTROL権限を有します。

以下はバケットACLの例です。バケット所有者（ユーザーUIN：100000000001）の完全制御権限を記述しています。

```
<AccessControlPolicy>
  <Owner>
    <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Root
        <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

以下はオブジェクトACLの例です。オブジェクト所有者（ユーザーUIN：100000000001）の完全制御権限を記述し、すべての人が読み取り可能な（匿名ユーザーのパブリック読み取り）権限を付与しています。

```
<AccessControlPolicy>
  <Owner>
    <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
  </Owner>
  <AccessControlList>
```

```
      <Grant>
        <Grantee>
          <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
        </Grantee>
        <Permission>FULL_CONTROL</Permission>
      </Grant>
      <Grant>
        <Grantee>
          <URI>http://cam.qcloud.com/groups/global/AllUsers</URI>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </AccessControlList>
  </AccessControlPolicy>
```

ユーザーポリシー

ユーザーはCAMで、ルートアカウント下の異なるタイプのユーザーに対し、異なる権限を付与することができます。

ユーザーポリシーとバケットポリシーの最大の違いは、ユーザーポリシーはエフェクト（Effect）、アクション（Action）、リソース（Resource）、条件（Condition、オプション）のみを記述し、プリンシパル（Principal）は記述しない点です。このため、ユーザーポリシーの作成完了後、サブユーザー、ユーザーグループまたはロールに対しバインド操作を実行する必要があります。またユーザーポリシーは、匿名ユーザーへのアクションおよびリソース権限の付与はサポートしていません。

プリセットポリシーを使用した権限のバインドを行うことも、ユーザーポリシーを自ら作成した後に指定のプリンシパルにバインドすることで、アカウント下のユーザーに対するアクセス管理を実現することもできます。その他の内容については、ユーザーポリシーのドキュメントをご参照ください。

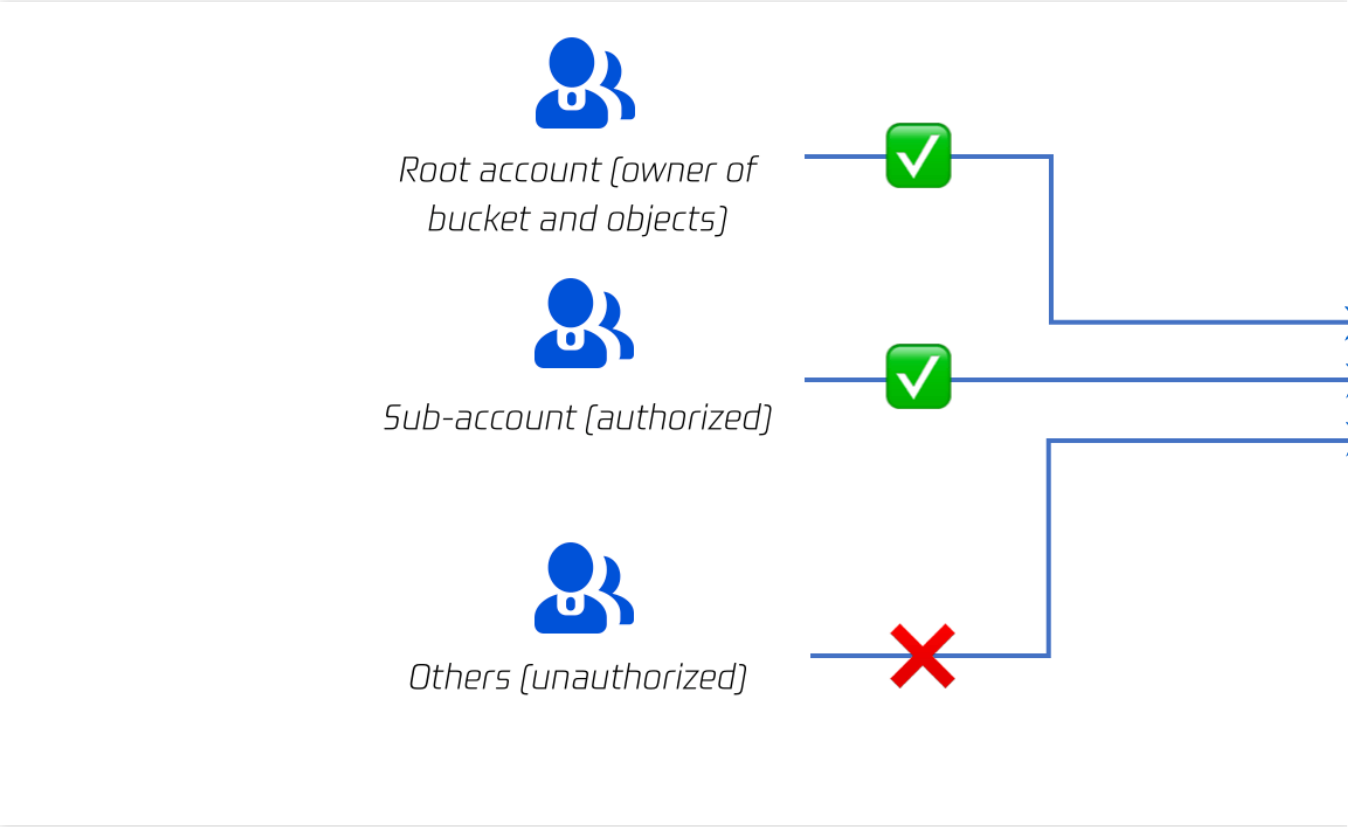
以下は、広州にあるバケットexamplebucket-1250000000のすべてのCOS操作権限を付与するポリシーです。ポリシーを保存した後、さらにCAMのサブユーザー、ユーザーグループまたはロールにバインドすることで発効します。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["cos:*"],
      "Resource": [
          "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*",
          "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/"
      ]
    }
  ],
    "Version": "2.0"
}
```

COSの権限承認とID認証のフロー

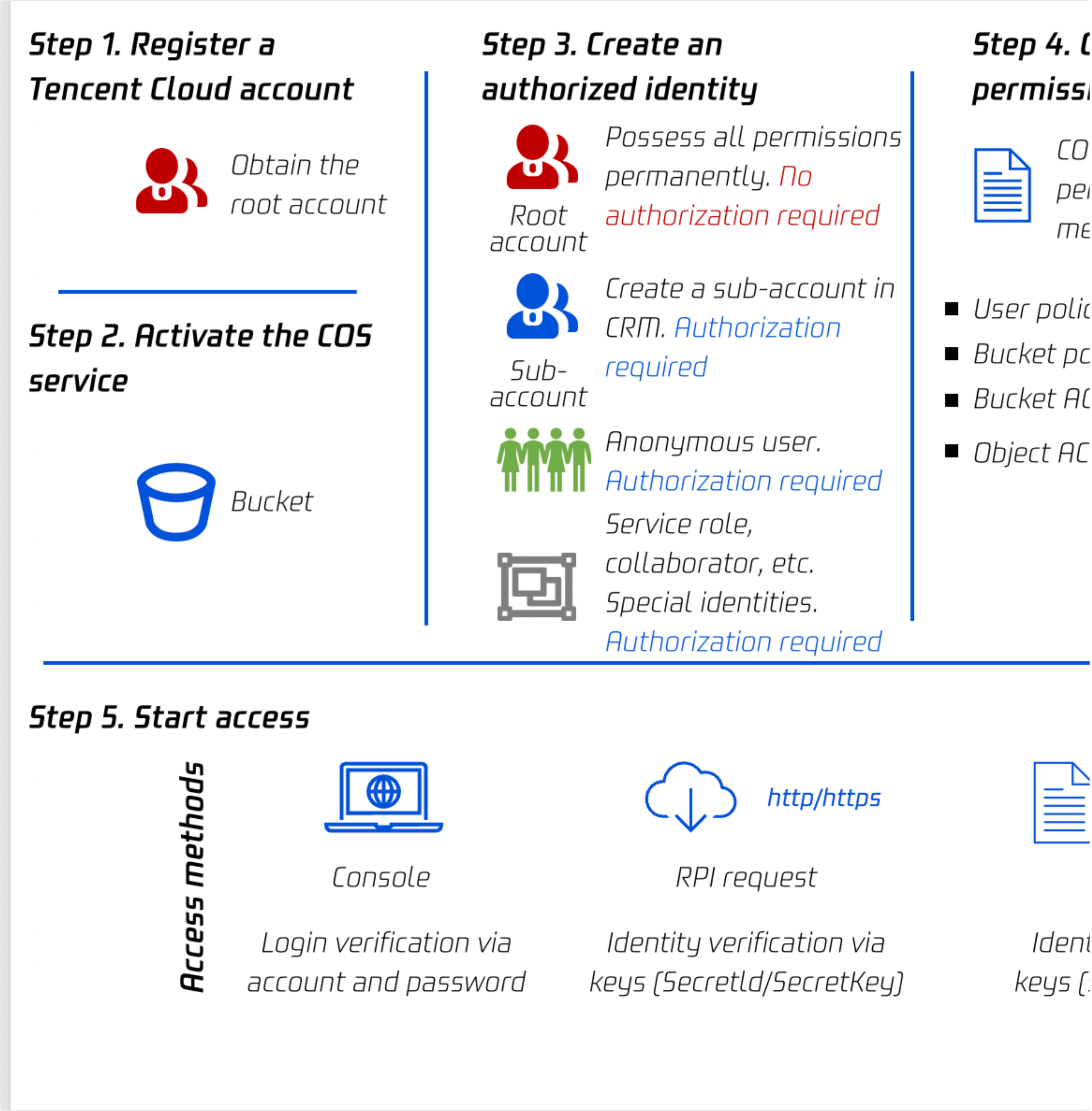
最終更新日：：2024-06-26 11:09:29

デフォルトでは、Cloud Object Storage（COS）のリソース（バケット、オブジェクト）はすべてプライベート読み取り/書き込みであり、オブジェクトのURLを取得しても、匿名ユーザーは署名がないため、URLからリソースの内容にアクセスすることはできません。


Root account (owner of bucket and objects)
✓
Sub-account (authorized)
✓
Others (unauthorized)
✗

主な手順

COSの権限承認およびID認証のフローは、Tencent Cloudアカウントの登録から始まる5段階となっています。Tencent Cloudアカウントの登録、COSサービスの有効化、権限承認IDの作成、IDへの権限設定、アクセスとID認証の開始、の5つです。


Step 1. Register a Tencent Cloud account
Obtain the root account
Step 2. Activate the COS service
Bucket
Step 3. Create an authorized identity
Root account
Possess all permissions permanently. No authorization required
Sub-account
Create a sub-account in CRM. Authorization required
Anonymous user. Authorization required
Service role, collaborator, etc. Special identities. Authorization required
Step 4. C
permissi
CO
pe
me
User polic
Bucket pc
Bucket AC
Object AC
Step 5. Start access
Access methods
Console
Login verification via account and password
http/https
RPI request
Identity verification via keys (SecretId/SecretKey)
Ident
keys (

ステップ1：Tencent Cloudアカウントの登録

Tencent Cloudアカウントの登録後、お客様のアカウントはルートアカウントとして存在します。ルートアカウントは最高権限を有します。

ステップ2：COSサービスのアクティブ化

COSサービスをアクティブ化すると、お客様の作成したすべてのバケットがルートアカウントの所有となります、ルートアカウントはすべてのリソースの最高使用権限を有するとともに、サブアカウントを作成してサブアカウントの権限承認を行う権限を有します。

ステップ3：権限承認IDの作成

注意：

バケットまたはオブジェクトをパブリック読み取りとして開放している場合を除き、COSへのアクセスには必ずID認証を経なければなりません。

ルートアカウントによって複数のIDを作成し、それぞれ異なるリソースの異なる使用権限を付与することができます。

例えば会社の同僚、特定部門のユーザーなどの指定したユーザーに権限承認を行いたい場合は、Cloud Access Management（CAM）コンソールでこれらのユーザーにサブアカウントを作成し、その後バケットポリシー、ユーザーポリシー（CAMポリシー）、バケットACL、オブジェクトACLなどの様々な権限承認方式によって、サブアカウントに指定したリソースの指定したアクセス権限を承認することができます。

例えば第三者が何のID認証も経ずに直接URLからオブジェクトをダウンロードすることを許可するなど、匿名ユーザーに権限承認を行いたい場合は、リソース権限をデフォルトのプライベート読み取りからパブリック読み取りに変更する必要があります。

Tencent Cloudのその他のサービス（CDNなど）でCOSバケットを使用したい場合も、同様の権限承認のフローに従う必要があります。お客様の許可の下で、これらのサービスはサービスロールによって正しくCOSにアクセスすることができます。作成済みのサービスロールはCAMコンソールで確認できます。

異なるTencent Cloudアカウント間で権限承認を行う状況で、1つのCOSバケットにのみ権限承認を行いたい場合は、 バケットポリシーまたはバケットACLによって直接別のルートアカウントに権限承認を行うことができます。複数のCOSバケットまたは複数のTencent Cloudリソース権限承認が必要な場合は、CAMコンソールでコラボレーターIDを作成し、より広範囲の権限承認を行うことができます。

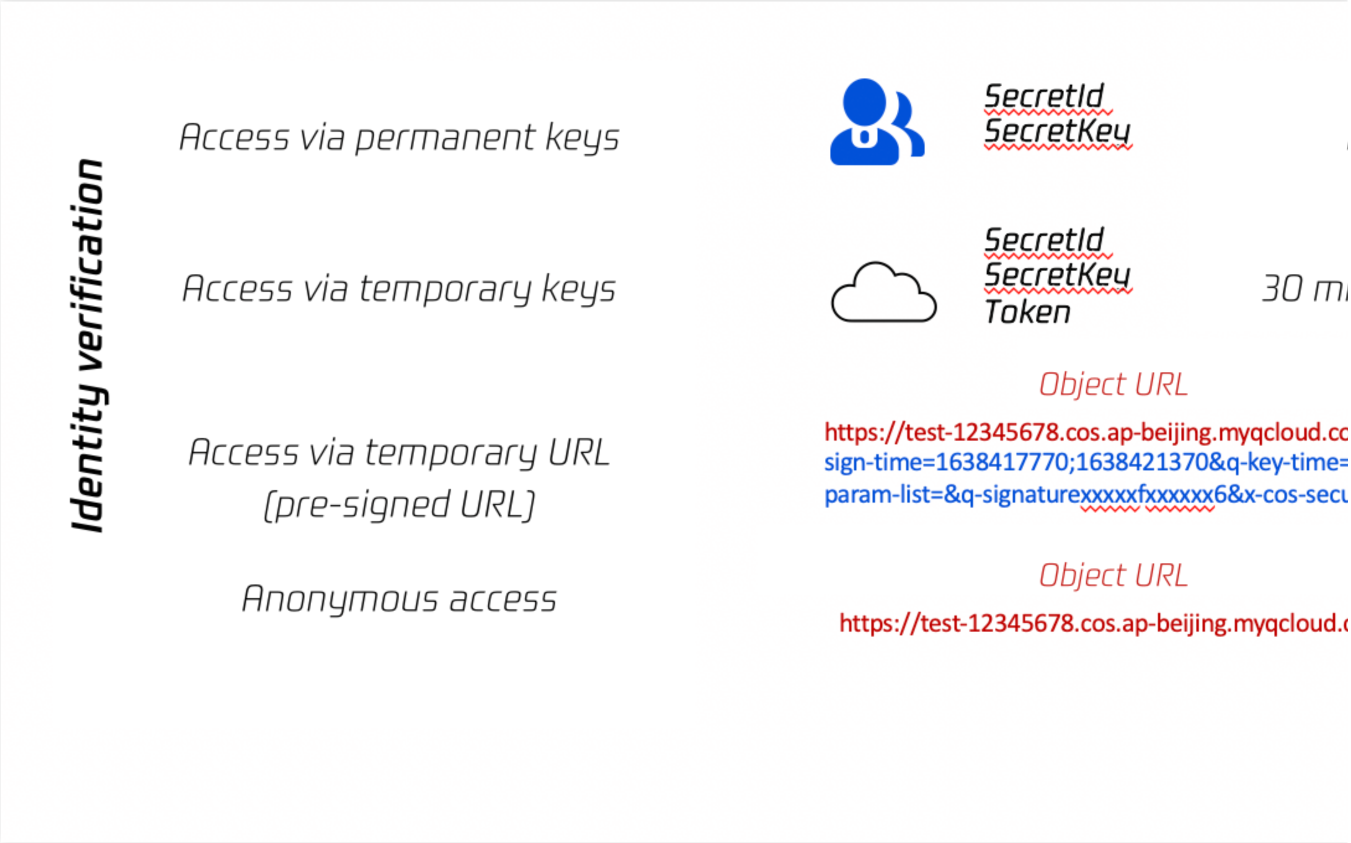
ステップ4：IDへの権限設定

COSは複数の権限設定方法をサポートしています。それにはバケットポリシー、ユーザーポリシー（CAMポリシー）、バケットACLおよびオブジェクトACLがあり、ご自身のユースケースに合った権限承認の方法を選択することができます。

ステップ5：アクセスとID認証の開始

COSにはコンソール、APIリクエスト、SDKなどの複数の手段でアクセスすることができます。セキュリティ上の観点から、バケットはデフォルトではプライベート読み取りとなっており、どの手段であってもID認証を経る必要があります。コンソールの場合は、アカウントパスワードを使用するとログインできます。APIリクエストおよびSDKの場合、すべてのユーザーはキー（SecretId/SecretKey）を使用してID認証を受ける必要があります。

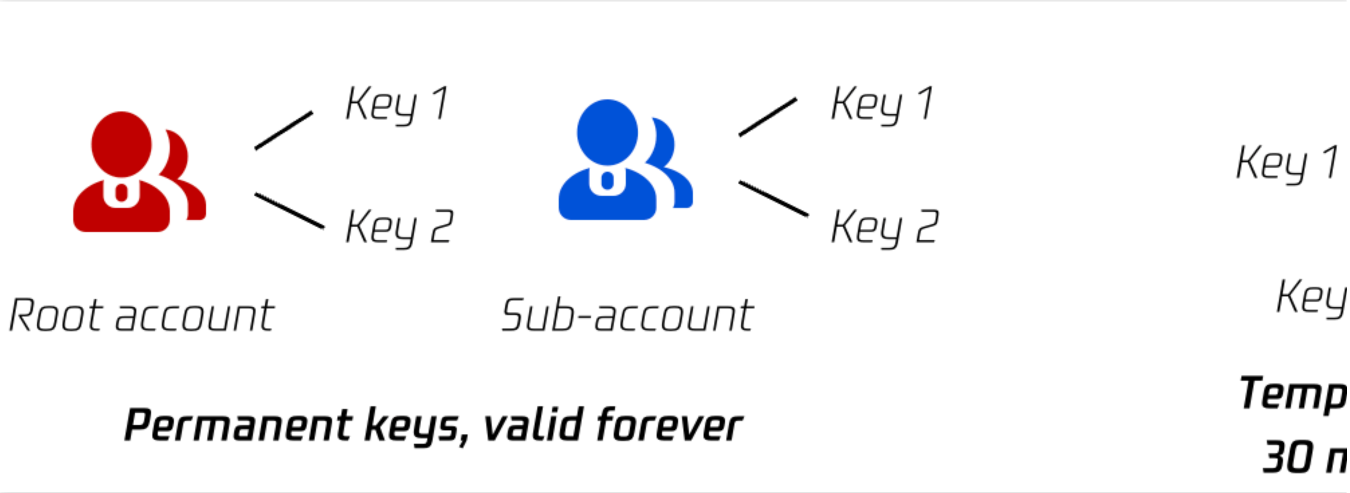
COSのID認証方式


Identity verification
Access via permanent keys
SecretId
SecretKey
Access via temporary keys
SecretId
SecretKey
Token
30 mi
Access via temporary URL
(pre-signed URL)
Object URL
https://test-12345678.cos.ap-beijing.myqcloud.cc
sign-time=1638417770;1638421370&q-key-time=
param-list=&q-signaturexxxxxfxxxxxx6&x-cos-secu
Anonymous access
Object URL
https://test-12345678.cos.ap-beijing.myqcloud.c

デフォルトではCOSバケットはプライベートであり、キー（パーマネントキー、一時キー）によるCOSアクセスであっても署名付きURLを使用したアクセスであっても、ID認証の段階を経る必要があります。特別な必要性がある場合はバケットをパブリック読み取りとして開放することもできますが、これはリスクを伴う操作であり、あらゆるユーザーがID認証を経ることなく、オブジェクトURLから直接オブジェクトをダウンロードできてしまいます。

1. パーマネントキーを使用したアクセス

キー（SecretId/SecretKey）は、ユーザーがTencent Cloud APIキーにアクセスしてID認証を受ける際に必要なセキュリティ証明書であり、APIキー管理で取得することができます。各ルートアカウントおよびサブアカウントはいずれも複数のキーを作成することができます。



パーマネントキーにはSecretIdとSecretKeyが含まれます。各ルートアカウントおよびサブアカウントはいずれも2対のパーマネントキーを生成することができます。パーマネントキーはアカウントの永続的なIDを表すもので、削除しなければ長期的に有効です。詳細については、パーマネントキーを使用したCOSアクセスをご参照ください。

2. 一時キーを使用したアクセス

一時キーにはSecretId、SecretKey、Tokenが含まれます。各ルートアカウントおよびサブアカウントはいずれも複数の一時キーを生成することができます。パーマネントキーと比較して、一時キーには有効期間（デフォルトでは1800秒）があり、有効期間はルートアカウントでは最長7200秒まで、サブアカウントでは最長129600秒までそれぞれ設定可能です。詳細については、フェデレーションによる一時アクセス証明書の取得をご参照ください。
一時キーはフロントエンドの直接送信などの、一時的な権限承認のシーンに適します。信頼性の低いユーザーに対しては、パーマネントキーではなく一時キーを発行することで、安全性を高めることができます。詳細については、一時キーを使用したCOSアクセスをご参照ください。

3. 一時URLを使用したアクセス（署名付きURL）

詳しい内容および使用説明については署名付きURLを使用したCOSアクセスをご参照ください。

オブジェクトのダウンロード

第三者がバケットからオブジェクトをダウンロードできるようにし、なおかつ相手にCAMアカウントまたは一時キーなどの方法を使用させたくない場合は、署名付きURLを使用して署名を第三者に渡し、一時的なダウンロード操作を完了させることができます。有効な署名付きURLを受領した人は誰でもオブジェクトをダウンロードできます。
コンソールまたはCOSBrowserから一時ダウンロードリンクを取得する（有効期間1～2時間）
コンソールまたはCOSBrowserから、一時ダウンロードリンクを直接素早く取得し、ブラウザでこのリンクを直接入力することでオブジェクトをダウンロードできます。詳細については、一時リンクのスピーディーな取得のドキュメントをご参照ください。

SDKを使用した署名付きURLの生成

SDKを使用して、有効期間をカスタマイズした署名付きURLを一括取得することができます。詳細については、SDKを使用した署名付きURLの一括取得のドキュメントをご参照ください。

署名ツールを使用した署名付きURLの生成

プログラミングに不慣れなユーザーに適しており、有効期間をカスタマイズした署名付きURLを取得することができます。詳細については、署名ツールの使用に関するドキュメントをご参照ください。

ご自身での署名リンクの結合

署名付きURLは実際にはオブジェクトのURLの後に署名を結合したものです。このため、SDK、署名生成ツールなどによってご自身で署名を生成し、URLと署名を結合して署名リンクとすることもできます。ただし、署名生成のアルゴリズムは複雑なため、一般的な状況ではこの方法の使用は推奨しません。

オブジェクトのアップロード

第三者がオブジェクトをバケットにアップロードできるようにし、なおかつ相手にCAMアカウントまたは一時キーなどの方法を使用させたくない場合は、署名付きURLを使用して署名を第三者に渡し、一時的なアップロード操作を完了させることができます。有効な署名付きURLを受領した人は誰でもオブジェクトをアップロードできます。

手段1：SDKを使用した署名付きURLの生成

各言語のSDKがアップロード署名付きURLの生成メソッドを提供しています。生成メソッドについては、署名付きURLによるアップロード権限承認を参照し、使いやすい開発言語を選択してください。

手段2：ご自身での署名リンクの結合

署名付きURLは実際にはオブジェクトのURLの後に署名を結合したものです。このため、SDK、署名生成ツールなどによってご自身で署名を生成し、URLと署名を結合して署名リンクとし、オブジェクトのアップロードに用いることもできます。署名生成のアルゴリズムは複雑なため、一般的な状況ではこの方法の使用は推奨されないことに注意が必要です。

4. 匿名アクセス

デフォルトではCOSバケットはプライベートであり、キー（パーマネントキー、一時キー）によるCOSアクセスであっても署名付きURLを使用したアクセスであっても、ID認証の段階を経る必要があります。

特別な必要性がある場合はバケットまたはオブジェクトをパブリック読み取りとして開放することもできますが、あらゆるユーザーがいかなるID認証も経ることなく、オブジェクトURLから直接オブジェクトをダウンロードできてしまいます。

注意：

リソースをパブリック読み取りとして開放することにはセキュリティ上のリスクが伴います。リソースのリンクが一度漏洩すると、誰でもアクセス可能となり、悪意あるユーザーによってトラフィックが不正使用されるおそれがあります。

バケットをパブリック読み取りとして開放する

コンソール上でバケット全体をパブリック読み取りに設定することができます。この場合、バケット内の各オブジェクトはすべてオブジェクトURLから直接ダウンロード可能となります。設定方法については、バケットアクセス権限の設定をご参照ください。

オブジェクトをパブリック読み取りとして開放する

コンソール上で単一のオブジェクトをパブリック読み取りに設定することができます。この場合、そのオブジェクトのみ、URLから直接ダウンロード可能となり、その他のオブジェクトは影響を受けません。設定方法については、オブジェクトのアクセス権限の設定をご参照ください。

フォルダをパブリック読み取りとして開放する

コンソール上でフォルダをパブリック読み取りに設定することができます。この場合、そのフォルダ下のすべてのオブジェクトがURLから直接ダウンロード可能となり、フォルダ外のオブジェクトは影響を受けません。設定方法については、フォルダの権限の設定をご参照ください。

最小権限の原則説明

最終更新日：：2024-06-26 11:09:29

概要

Cloud Object Storage（COS）を使用する際、一時キーを使用して対応するリソースの操作権限をユーザーに与えるか、またはご自身のサブユーザーまたはコラボレーターに対し、対応するユーザーポリシーを付与することで、それらのアカウントがCOS上のリソース操作を補助できるよう許可するか、あるいはバケットに対応するバケットポリシーを追加し、指定のユーザーがバケット内で指定された操作を行い、指定されたリソースを操作することを許可するかの、いずれかが必要となる可能性があります。これらの権限設定を行う際は、データアセットのセキュリティを保障するため、必ず**最小権限の原則**に従う必要があります。
最小権限の原則とは、権限承認を行う際、必ず権限の範囲を明確化し、**指定するユーザー**に、**どのような条件**の下で、**どのような操作**を実行でき、**どのようなリソース**にアクセスできる権限を承認するかを明確にする必要があることを指します。

注意事項

権限承認の際は最小権限の原則を厳守し、ユーザーに対し限られた操作（例えば `action:GetObject` の権限承認など）の実行、限られたリソース（例えば `resource:examplebucket-1250000000/exampleobject.txt` の権限承認など）へのアクセスのみを許可することをお勧めします。
大きすぎる権限を与えることで、意図しない越権操作が行われ、データセキュリティリスクが生じることを避けるため、ユーザーにすべてのリソースへのアクセス（例えば `resource:*` など）、またはすべての操作（例えば `action:*` など）の権限を承認することは可能な限り避けるよう強く推奨します。
潜在的なデータセキュリティリスクの例を次に挙げます。
データ漏洩：ユーザーに指定のリソースをダウンロードする権限、例えば `examplebucket-1250000000/data/config.json` および `examplebucket-1250000000/video/` のダウンロード権限を与えても、権限ポリシーに `examplebucket-1250000000/*` を設定していた場合、このバケット下のすべてのオブジェクトのダウンロードが許可されることになり、越権操作行為があった場合は意図しないデータ漏洩が発生します。
データが上書きされる：ユーザーにリソースをアップロードする権限、例えば `examplebucket-1250000000/data/config.json` および `examplebucket-1250000000/video/` のアップロード権限を与えても、権限ポリシーに `examplebucket-1250000000/*` を設定していた場合、このバケット下のすべてのオブジェクトのアップロードが許可されることになり、越権操作行為があった場合は意図しないオブジェクトの上書きが発生します。このようなリスクを防止するためには、最小権限の原則に従うほか、バージョン管理によってデータのすべての過去バージョンを保持し、追跡に役立てることも可能です。

権限の漏洩：ユーザーにバケットのオブジェクトリスト `cos:GetBucket` の出力を許可しても、権限ポリシーに `cos:*` を設定していた場合、バケット権限の再承認、オブジェクトの削除、バケットの削除を含む、このバケットのすべての操作が許可されることになり、データに極めて大きなリスクをもたらします。

使用ガイド

最小権限の原則の下では、ポリシーに次の情報を明確に指定する必要があります。

プリンシパル（principal）：どのサブユーザー（ユーザーIDの入力が必要）、コラボレーター（ユーザーIDの入力が必要）、匿名ユーザーまたはユーザーグループに権限を付与したいかを明確に指定する必要があります。一時キーを使用してアクセスする場合、この項の指定は不要です。

ステートメント（statement）：次のいくつかのパラメータに、対応するパラメータを明確に入力します。

エフェクト（effect）：このポリシーが「許可」であるか「明示的な拒否」であるかを明確に述べる必要があります。allowとdenyの2種類が含まれます。

アクション（action）：このポリシーが許可または拒否するアクションを明確に述べる必要があります。アクションは単一のAPIのアクションまたは複数のAPIアクションのセットとすることができます。

リソース（resource）：このポリシーが権限を承認する具体的なリソースを明確に述べる必要があります。リソースは6つの部分で記述されます。リソース範囲の指定は、 `exampleobject.jpg` などの指定されたファイル、または `examplePrefix/*` などの指定されたディレクトリとすることができます。業務上の必要がない限り、すべてのリソースにアクセスする権限（ワイルドカード `*` ）をユーザーにみだりに付与しないでください。

条件（condition）：ポリシー発効の制約条件を記述します。条件はオペレーター、アクションキーとアクション値から構成されています。条件値には時間、IPアドレスなどの情報を含めることができます。

一時キーの最小権限ガイド

一時キー申請の過程で、権限ポリシーのPolicyフィールドを設定することで、操作およびリソースを制限し、権限を指定された範囲内に限定することができます。一時キー生成に関する説明については、一時キーの生成および使用ガイドのドキュメントをご参照ください。

権限承認の例

Java SDKを使用して、あるオブジェクトへのアクセス権限を承認

Java SDKを使用して、バケット `examplebucket-1250000000` 内のオブジェクト `exampleObject.txt` のダウンロード権限を承認したい場合、それに応じて設定する必要があるコードは次のようになります。

```
// githubが提供するmaven統合メソッドによってjava sts sdkをインポートします
import java.util.*;
import org.json.JSONObject;
import com.tencent.cloud.CosStsClient;

public class Demo {
    public static void main(String[] args) {
        TreeMap<String, Object> config = new TreeMap<String, Object>();

        try {
            String secretId = System.getenv("secretId");//ユーザーのSecretIdです。サブ
            String secretKey = System.getenv("secretKey");//ユーザーのSecretKeyです。
            // ご自身のSecretIdに置き換えます
            config.put("SecretId", secretId);
            // ご自身のSecretKeyに置き換えます
            config.put("SecretKey", secretKey);

            // 一時キーの有効期間の単位は秒であり、デフォルトでは1800秒、最長7200秒まで設定可能
            config.put("durationSeconds", 1800);

            // ご自身のbucketに置き換えます
            config.put("bucket", "examplebucket-1250000000");
            // bucketの所在リージョンに置き換えます
            config.put("region", "ap-guangzhou");

            // ここを許可するパスのプレフィックスに変更します。ご自身のウェブサイトのユーザーログ
            // 「*」を入力すると、ユーザーにすべてのリソースへのアクセスを許可することになります。
            config.put("allowPrefix", "exampleObject.txt");

            // キーの権限リストです。シンプルアップロード、フォームアップロード、マルチパートアッ
            String[] allowActions = new String[] {
                    // データをダウンロードします
                    "name/cos:GetObject"
            };
            config.put("allowActions", allowActions);

            JSONObject credential = CosStsClient.getCredential(config);
            //成功すると一時キー情報が返されます。次のようにキー情報を印刷します
            System.out.println(credential);
        } catch (Exception e) {
            //失敗するとエラーがスローされます
            throw new IllegalArgumentException("no valid secret !");
        }
    }
}
```

APIを使用して、あるオブジェクトへのアクセス権限を承認

APIを使用して、バケット `examplebucket-1250000000` 内のオブジェクト `exampleObject.txt` およびディレクトリ `examplePrefix` 下のすべてのオブジェクトのダウンロード権限を承認したい場合、そのために書き込む必要があるPolicyは次のようになります。

```
{
  "version": "2.0",
  "statement":[
    {
      "action":[
        "name/cos:GetObject"
      ],
      "effect": "allow",
      "resource":[
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/exampleObject.
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/examplePrefix/
```

```
            ]
        }
    ]
}
```

署名付きURLの最小権限ガイド

署名付きURL方式によって一時的なアップロードおよびダウンロード操作を実現できます。また、署名付きURLは誰にでも送信することができ、有効な署名付きURLを受領した人は誰でもオブジェクトをアップロードまたはダウンロードできます。

注意：

一時キーとパーマネントキーはどちらも署名付きURLの生成に用いることができますが、最小権限の原則に従って一時キーの生成を行い、一時キーを使用して署名の計算を行うことを強く推奨します。セキュリティ上のリスクを防止するため、権限が大きすぎるパーマネントキーは可能な限り使用しないでください。

権限承認の例

ユーザーに対し、署名付きURLを使用してオブジェクトをダウンロードする権限を承認

一時キーを使用して署名付きのダウンロードリンクを生成し、返す必要のあるいくつかのパブリックヘッダー（例えばcontent-type、content-languageなど）の上書きを設定します。Javaのサンプルコードは次のとおりです。

```
// 取得した一時キー (tmpSecretId、tmpSecretKey、sessionToken)を渡します
String tmpSecretId = "SECRETID";
String tmpSecretKey = "SECRETKEY";
String sessionToken = "TOKEN";
COSCredentials cred = new BasicSessionCredentials(tmpSecretId, tmpSecretKey, sessio
// bucketのリージョンを設定します。COSリージョンの略称についてはhttps://cloud.tencent.com/dc
// clientConfigにはregion、https(デフォルトではhttp)、タイムアウト、プロキシなどを設定するset
Region region = new Region("COS_REGION");
ClientConfig clientConfig = new ClientConfig(region);
// httpsプロトコルを使用したURLを生成したい場合はこの行を設定します。設定することをお勧めします。
// clientConfig.setHttpProtocol(HttpProtocol.https);
// cosクライアントを生成します
COSClient cosClient = new COSClient(cred, clientConfig);
// バケットの命名形式はBucketName-APPIDです
String bucketName = "examplebucket-1250000000";
// ここでのkeyはオブジェクトキーです。オブジェクトキーはオブジェクトのバケット内での固有識別子です
String key = "exampleobject";
GeneratePresignedUrlRequest req =
        new GeneratePresignedUrlRequest(bucketName, key, HttpMethodName.GET);
// ダウンロード時に返されるhttpヘッダーを設定します
ResponseHeaderOverrides responseHeaders = new ResponseHeaderOverrides();
String responseContentType = "image/x-icon";
String responseContentLanguage = "zh-CN";
// レスポンスヘッダーに含まれるファイル名の情報を設定します
String responseContentDispositon = "filename=\\"exampleobject\\"";
String responseCacheControl = "no-cache";
String cacheExpireStr =
        DateUtils.formatRFC822Date(new Date(System.currentTimeMillis() + 24L * 3600
responseHeaders.setContentType(responseContentType);
responseHeaders.setContentLanguage(responseContentLanguage);
responseHeaders.setContentDisposition(responseContentDispositon);
responseHeaders.setCacheControl(responseCacheControl);
responseHeaders.setExpires(cacheExpireStr);
req.setResponseHeaders(responseHeaders);
// 署名の有効期限を設定します(オプション)。設定しない場合は、デフォルトでClientConfigの署名有効
// ここでは署名が30分後に期限切れとなるよう設定します
Date expirationDate = new Date(System.currentTimeMillis() + 30L * 60L * 1000L);
req.setExpiration(expirationDate);
URL url = cosClient.generatePresignedUrl(req);
System.out.println(url.toString());
cosClient.shutdown();
```

ユーザーポリシーの最小権限ガイド

ユーザーポリシーとはCAMコンソールで追加するユーザー権限ポリシーを指し、ユーザーにCOSリソースへのアクセス権限を与えるために用いられます。ユーザーアクセスポリシー概要の設定説明については、 アクセスポリシーの言語概要のドキュメントをご参照ください。

権限承認の例

アカウントに対し、あるオブジェクトへのアクセス権限を承認

アカウントUIN `100000000001` に対し、バケット `examplebucket-1250000000` 内のオブジェクト `exampleObject.txt` のダウンロード権限を承認したい場合、それに応じたアクセスポリシーは次のようになります。

```
{
   "version": "2.0",
   "principal":{
      "qcs": [
         "qcs::cam::uin/100000000001:uin/100000000001"
      ]
   },
    "statement":[
        {
            "action":[
                "name/cos:GetObject"
                ],
                "effect": "allow",
                "resource":[
                    "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000.ap-g
                ]
            }
        ]
    }
```

サブアカウントに対し、あるディレクトリへのアクセス権限を承認

サブアカウントUIN `100000000011` （ルートアカウントのUINは `100000000001` ）に対し、バケット `examplebucket-1250000000` 内のディレクトリ `examplePrefix` 下のオブジェクトのダウンロード権限を承認したい場合、それに応じたアクセスポリシーは次のようになります。

```
{
   "version": "2.0",
   "principal":{
      "qcs": [
         "qcs::cam::uin/100000000001:uin/100000000011"
      ]
   },
    "statement":[
        {
            "action":[
                "name/cos:GetObject"
            ],
            "effect": "allow",
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000.ap-g
            ]
        }
    ]
}
```

バケットポリシーの最小権限ガイド

バケットポリシーとはバケット内で設定するアクセスポリシーを指し、指定のユーザーがバケットおよびバケット内のリソースに対し指定された操作を行うことを許可します。バケットポリシーの設定については、バケットポリシーの追加のドキュメントをご参照ください。

権限承認の例

サブアカウントに対し特定のオブジェクトへのアクセス権限を承認

サブアカウントUIN `100000000011` （ルートアカウントのUINは `100000000001` ）に対し、バケット `examplebucket-1250000000` 内のオブジェクト `exampleObject.txt` およびディレクトリ `examplePrefix` 下のすべてのオブジェクトのダウンロード権限を承認したい場合、それに応じたアクセスポリシーは次のようになります。

```
{
  "Statement": [
    {
      "Action": [
        "name/cos:GetObject"
      ],
      "Effect": "allow",
      "Principal": {
        "qcs": [
          "qcs::cam::uin/100000000001:uin/100000000011"
        ]
      },
      "Resource": [
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/exampleObject.
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/examplePrefix/
      ]
    }
  ],
  "version": "2.0"
}
```

アクセスポリシーの評価フロー

最終更新日：：2024-06-26 11:09:29

COSバケットおよびバケット内のリソースにアクセスする際は権限承認を経てからアクセスする必要があります。Tencent Cloudの権限システムでは、リソースが所属するルートアカウントはデフォルトでバケットおよびバケット内のリソースに対しすべての管理権限を有します。CAMユーザー（その他のルートアカウント、コラボレーター、サブアカウント）および匿名ユーザーなどのその他のタイプのユーザーは、ルートアカウントによる権限承認を経てからでなければアクセスできません。

アカウントのアクセスポリシーには、ユーザーグループポリシー、ユーザーポリシー、バケットアクセス制御リスト（ACL）、バケットポリシー（Policy）などの異なるポリシータイプがあります。アクセスポリシーの評価においては、次のいくつかの重要な要素があります。

1. ユーザーのID認証：ユーザーがCOS上のリソースにアクセスする場合、次の2つの状況があります。

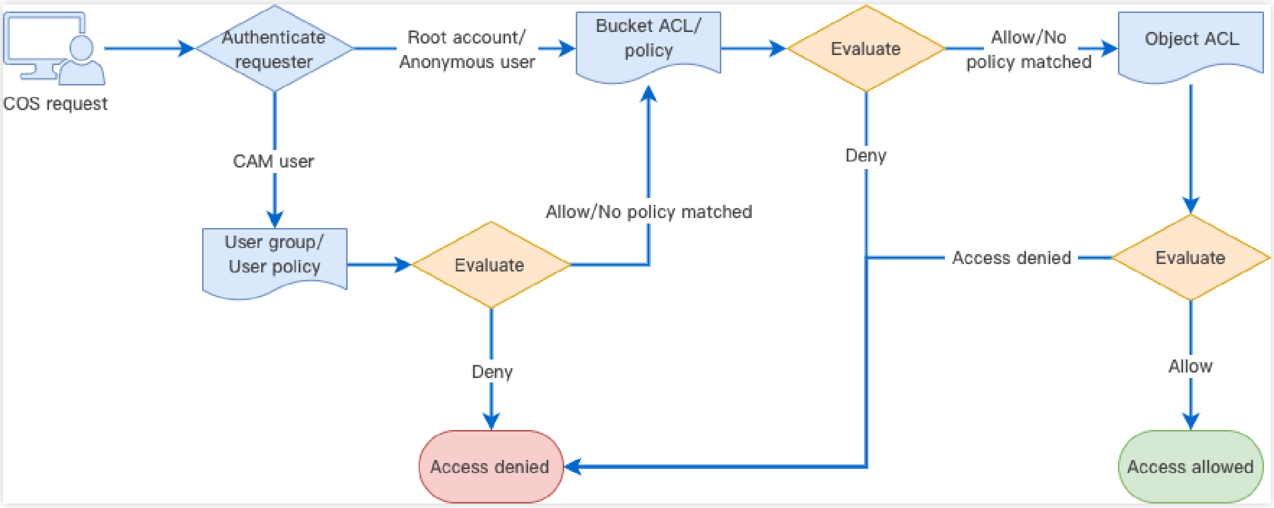
リクエスト署名がある場合、COSはリクエスト署名からユーザーのアカウント情報を解析した後、リクエストをCAMに転送してID認証を行います。

署名のないリクエストの場合は、匿名ユーザーと認識され、認証の次の段階に進みます。

2. アクセスポリシーの種類：アクセスポリシーにはユーザーグループ、ユーザー、バケットなどの複数のタイプが含まれます。アクセスポリシーの順序はアクセスポリシーの種類によって決定されます。

3. ポリシーのコンテキスト情報：ユーザーのリソースアクセスリクエストを処理する際は、ユーザーグループポリシー、ユーザーポリシー、バケットポリシーなどの複数のポリシーに記録された権限の詳細に基づいて総合的に判断し、リクエストを許可するかどうかを最終的に決定します。

アクセスポリシーの評価フロー


COS request
Authenticate requester
Root account/ Anonymous user
Bucket ACL/ policy
Evaluate
Allow/No policy matched
Object ACL
CAM user
Deny
User group/ User policy
Evaluate
Allow/No policy matched
Access denied
Evaluate
Deny
Allow
Access denied
Access allowed

Tencent Cloud COSがリクエストを受信すると、最初にリクエスト送信者のIDを確認し、リクエスト送信者が関連の権限を有しているかどうかを検証します。検証のプロセスには、ユーザーポリシー、バケットアクセスポリシー、リソースベースのアクセス制御リストのチェックが含まれ、リクエストに対する認証を行います。

Tencent Cloud COSはリクエストを受信した際、最初にID認証を行い、ID認証の結果に基づいてリクエスト送信者のIDを分類します。IDのカテゴリーに応じて異なるアクションがとられる可能性があります。

1. 検証済みのTencent Cloudルートアカウント：ルートアカウントはその所属リソースに対しすべての操作権限を有します。一方、アカウントに属さないリソースについては、リソース権限の評価が必要であり、認証に合格するとリソースへのアクセスが許可されます。
2. 検証済みのCAMユーザー（サブアカウントまたはコラボレーター）：ユーザーポリシーの評価 —— CAMユーザーは必ず親アカウントにあたるルートアカウントの権限承認を受けていなければ、関連のアクセスが許可されません。CAMユーザーが他のルートアカウントに属するリソースにアクセスしたい場合は、CAMユーザーが属するルートアカウントのリソース権限の評価が必要であり、認証に合格するとリソースへのアクセスが許可されます。
3. ID特性を持たない匿名ユーザー：リソース権限の評価 —— バケットアクセスポリシーまたはバケット、オブジェクトのアクセス制御リストの権限を評価し、認証に合格するとリソースへのアクセスが許可されます。
4. 上記のユーザーカテゴリー以外のリクエスト送信者：アクセスが拒否されます。

アクセスポリシーの評価根拠

Tencent Cloudの権限システムでは、アクセスポリシーの評価フローにおいて、全プロセスでポリシーのコンテキスト情報に基づいて権限の評価を行います。また同時に次のいくつかの基本原則があります。

1. デフォルトでは、すべてのリクエストが暗黙的に拒否（deny）されます。ルートアカウントはデフォルトでアカウント下のすべてのリソースのアクセス権限を有します。
2. ユーザーグループポリシー、ユーザーポリシー、バケットポリシーまたはバケット/オブジェクトのアクセス制御リストに明示的な許可が存在する場合は、このデフォルト値を上書きします。
3. いずれかのポリシーの中に明示的な拒否がある場合は、あらゆる許可を上書きします。
4. 有効な権限の範囲はIDベースのポリシー（ユーザーグループポリシー、ユーザーポリシー）とリソースベースのポリシー（バケットポリシーまたはバケット/オブジェクトのアクセス制御リスト）を合わせたものとなります。

説明：

明示的な拒否：ユーザーポリシー、ユーザーグループポリシー、バケットPolicyの中で特定のユーザーに対して明確なDenyポリシーが存在する場合。例えば、ルートアカウントがユーザーポリシーの中で、サブユーザーUIN 100000000011が `GET Object` 操作を行うことを明確にDenyと設定している場合、そのサブユーザーはそのルートアカウント下のオブジェクトリソースをダウンロードすることはできません。

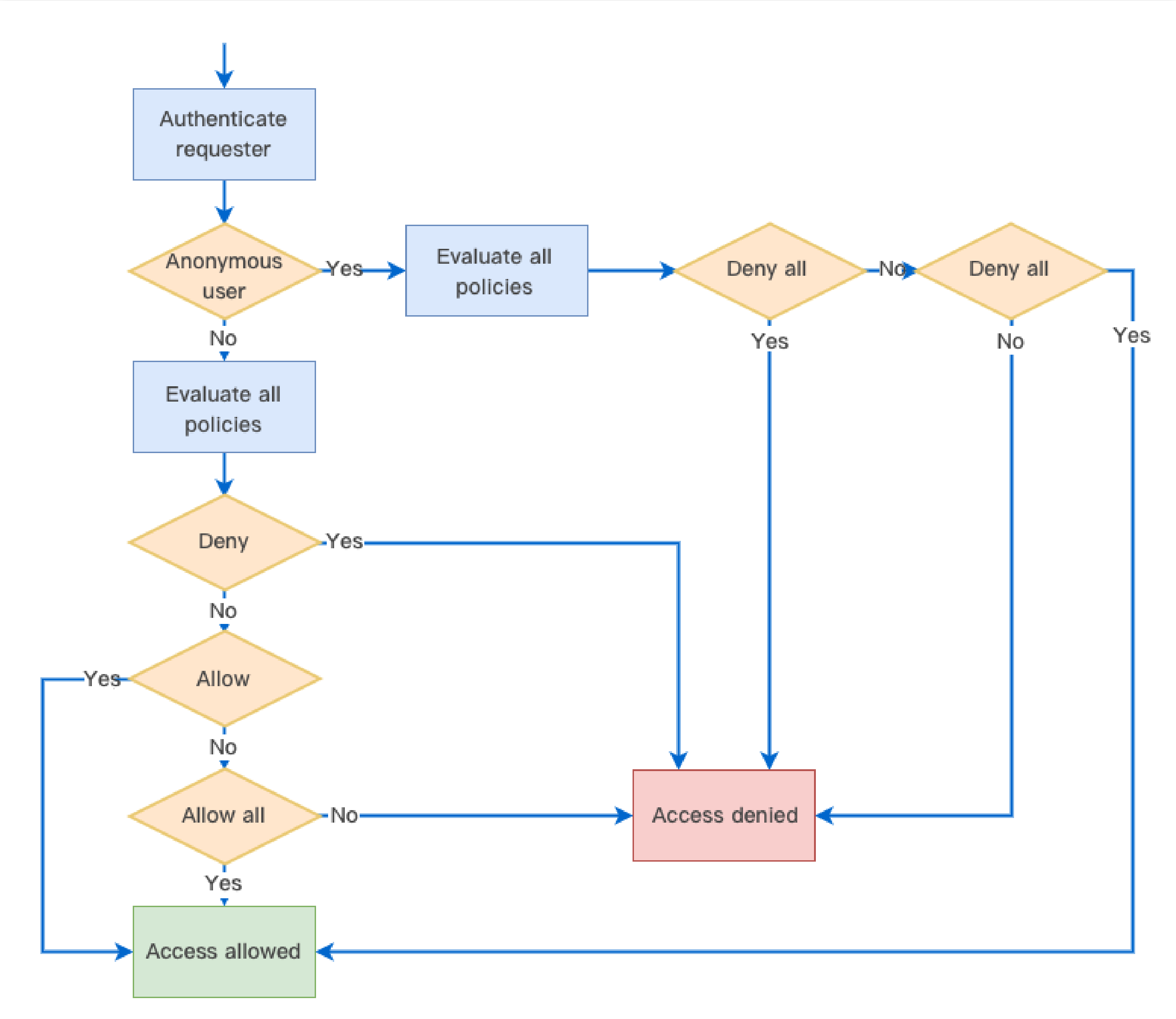
明示的な許可：ユーザーポリシー、ユーザーグループポリシー、バケットPolicy、バケットACLの中で、 `grant-\\*` によって特定のユーザーを明確に指定し、特定のユーザーに対し明確なAllowポリシーを設けます。

すべての人を拒否：バケットPolicyの中で、 `Deny anyone` と明確に指定します。すべての人を拒否すると、任意の署名なしリクエストが拒否されます。署名付きリクエストに対してはIDベースのポリシーによる認証が行われます。

すべての人を許可：バケットPolicyの中で、 `Allow anyone` と明確に指定するか、またはバケットACLの中で `public-\\*` と明確に指定します。

有効な権限の範囲はIDベースのポリシーとリソースベースのポリシーを合わせたものとなります。1回の完全な認証において、COSは最初にユーザーのIDを解析し、ユーザーのIDに基づいて、そのユーザーがアクセス権限を持つリソースの権限チェックを行います。同時にリソースベースのポリシーに基づき、ユーザーを匿名ユーザーとみなして権限チェックを行います。2回のチェックのうち1回が成功するとアクセスが許可されます。

アクセスポリシーの評価根拠は次の図のとおりです。最初にリクエストの署名の有無に基づき、ユーザーが匿名ユーザーかどうかを評価します。ユーザーが匿名ユーザーであれば、「すべての人を拒否」または「すべての人を許可」のポリシーがないかどうかを評価し、これに基づいてアクセス許可またはアクセス拒否の判断を行います。ユーザーが正当なCAMユーザーまたはリソースを所有するルートアカウントの場合は、明示的な拒否、明示的な許可または「すべての人を許可」のポリシーがないかどうかを評価し、これに基づいてアクセス許可またはアクセス拒否の判断を行います。


Authenticate requester
Anonymous user
Yes
No
Evaluate all policies
Deny all
No
Yes
Deny all
No
Yes
Evaluate all policies
Deny
Yes
No
Allow
Yes
No
Allow all
No
Yes
Access denied
Access allowed

ポリシーのコンテキスト情報

ポリシーのコンテキスト情報とは、ポリシーに記録される権限の詳細を指します。最小権限の原則の下で、ユーザーはポリシーの中で次の情報を明確に指定する必要があります。

プリンシパル（principal）：どのサブユーザー（ユーザーIDの入力が必要）、コラボレーター（ユーザーIDの入力が必要）、匿名ユーザーまたはユーザーグループに権限を付与したいかを明確に指定する必要があります。一時キーを使用してアクセスする場合、この項の指定は不要です。

ステートメント（statement）：次のいくつかのパラメータに、対応するパラメータを明確に入力します。

エフェクト（effect）：このポリシーが「許可」であるか「明示的な拒否」であるかを明確に述べる必要があります。allowとdenyの2種類が含まれます。

アクション（action）：このポリシーが許可または拒否するアクションを明確に述べる必要があります。アクションは単一のAPIのアクションまたは複数のAPIアクションのセットとすることができます。

リソース（resource）：このポリシーが権限を承認する具体的なリソースを明確に述べる必要があります。リソースは6セグメント式で記述します。リソース範囲の指定は、exampleobject.jpgなどの指定されたファイル、またはexamplePrefix/*などの指定されたディレクトリとすることができます。業務上の必要がない限り、すべてのリソースにアクセスする権限（ワイルドカード*）をユーザーにみだりに付与しないでください。
条件（condition）：ポリシー発効の制約条件を記述します。条件はオペレーター、アクションキーとアクション値から構成されています。条件値には時間、IPアドレスなどの情報を含めることができます。
ポリシーの作成は一定のポリシー構文に従って行う必要があります。アクセスポリシーの言語概要を参照し、業務上の必要性に応じて作成することができます。ユーザーポリシーおよびバケットポリシーの作成例については、ユーザーポリシー構文の構造およびバケットポリシーの例をそれぞれご参照ください。

アクセスポリシーの評価の例

ルートアカウントUIN 100000000001がサブアカウントUIN 100000000011にユーザープリセットポリシーをバインドし、このサブアカウントにルートアカウント下のリソースを読み取り専用として許可したとします。このユーザーポリシーの詳細は次のとおりです。このユーザーポリシーはこのサブアカウントに対し、すべての `List` 、 `Get` 、 `Head` 操作および `OptionsObject` 操作の実行を許可しています。

```
{
    "version": "2.0",
    "statement": [
        {
            "action": [
                "cos:List*",
                "cos:Get*",
                "cos:Head*",
                "cos:OptionsObject"
            ],
            "resource": "*",
                "effect": "allow"
            }
        ]
    }
```

また、ルートアカウントはあるプライベート読み取り/書き込みバケット `examplebucket-1250000000` に次のバケットポリシーを追加しました。

```
{
  "Statement": [
    {
        "Principal": {
          "qcs": [
            "qcs::cam::anyone:anyone"
          ]
        },
        "Effect": "Deny",
        "Action": [
          "name/cos:GetObject"
        ],
        "Resource": [
          "qcs::cos:ap-guangzhou:uid/100000000011:examplebucket-1250000000/*"
        ]
      }
    ],
    "version": "2.0"
  }
```

このバケットポリシーはすべてのユーザーのオブジェクトダウンロード実行（ `GetObject` ）の操作を明示的に拒否しています。このため、アクセスポリシー評価のフローにおいては、次が行われます。

1. このサブアカウントが署名パラメータによって `GetObject` をリクエストした場合、このリクエストが表すユーザーのIDが対応するユーザーポリシーとマッチした場合、アクセスポリシーの評価検証は合格となります。
2. このサブアカウントが署名のないパラメータによって `GetObject` をリクエストした場合、システムによって匿名のリクエストと判断され、バケットポリシーによってアクセスが**拒否**されます。

権限制御方法の紹介

バケットポリシー

最終更新日：：2024-06-26 11:09:29

バケットポリシーは設定したバケットおよびバケット内のオブジェクトに対して機能します。バケットポリシーによってCAMサブアカウント、その他のルートアカウント、また匿名のユーザーに対しても、バケットおよびバケット内のオブジェクトの操作権限を付与することができます。

概要

注意：

Tencent Cloudのルートアカウントは、そのアカウント下のリソース（バケットを含む）に対する最大の権限を有しています。バケットポリシーではほとんどすべての操作を制限できますが、ルートアカウントは常にPUT Bucket Policy操作の権限を有しており、ルートアカウントがこの操作を呼び出す際、バケットポリシーのチェックは行われません。

バケットポリシー（Bucket Policy）はJSON言語を使用して記述し、匿名IDまたはTencent CloudのあらゆるCAMアカウントに対し、バケット、バケット操作、オブジェクトまたはオブジェクト操作の権限を付与できます。Tencent Cloud COSのバケットポリシーは、そのバケット内のほとんどすべての操作の管理に用いることができます。ACLでは記述できないアクセスポリシーを、バケットポリシーを使用して管理することをお勧めします。

ユースケース

注意：

バケット作成およびバケットリスト取得という2つのサービスレベルの操作権限は、CAMコンソールで設定する必要があります。

そのCOSバケットに誰がアクセス可能かをお知りになりたい場合は、バケットポリシーの使用をお勧めします。バケットを検索し、バケットポリシーをチェックすることで、アクセス可能な人が誰かを知ることができます。次のようなケースで推奨されます。

特定のバケットについて権限を付与する

バケットポリシーがACLに比べてより柔軟である

ユーザーポリシーと異なり、バケットポリシーがクロスアカウント権限付与および匿名ユーザーへの権限付与をサポートしている

バケットポリシーの構成

バケットポリシーはJSON言語を使用して記述します。構文は、プリンシパル（principal）、エフェクト（effect）、アクション（action）、リソース（resource）、条件（condition）などの基本要素を含めたアクセスポリシー言語の統一ルールに従います。詳細については、アクセスポリシーの言語概要をご参照ください。

このうち、バケットポリシーのリソース範囲はそのバケット内に限定されますが、バケット全体、指定されたディレクトリ、指定されたオブジェクトに対する権限を付与することができます。

注意：

バケットポリシーを追加する際は、必ず業務の必要性に応じて、最小権限の原則に従って権限を付与してください。他のユーザーに対し、すべてのリソース `(resource:*)` またはすべてのアクション `(action:*)` の権限を直接与えてしまうと、権限が大きすぎるためにデータセキュリティ上のリスクが生じる場合があります。

コンソール設定の例

説明：

COSコンソールを使用してバケットポリシーを設定する場合、ユーザーが持つバケットに関連する権限を付与する必要があります。例えば、バケットタグの取得やバケットリストアップの権限などです。

バケットポリシーのサイズ制限は20KBです。

例：サブアカウントが持つバケットの特定のディレクトリの全権限を付与します。設定情報は次のとおりです。

設定項目	設定値
エフェクト	許可
プリンシパル	サブアカウント、サブアカウントのUIN。このサブアカウントは現在のルートアカウント下のサブアカウントである必要があります。例：100000000011
リソース	特定のディレクトリのプレフィックス。例： `folder/sub-folder/*`
アクション	すべての操作
条件	なし

コンソールは、グラフィカル設定とポリシー設定という2種類の方式でのバケットポリシーの追加と管理をサポートしています。

グラフィカル設定

ターゲットバケットの**権限管理**に進み、**Policy権限設定 > グラフィック設定**を選択し、**ポリシーの追加**をクリックして、ポップアップウィンドウでポリシーの設定を行います。

ステップ1：テンプレートの選択（オプション）

COSは複数のポリシーテンプレートをご提供しており、様々な被付与者、リソース範囲の組み合わせを選択することで、お客様がバケットポリシーの設定を迅速に行えるようサポートします。テンプレートがニーズに合わない場合はこのステップをスキップするか、またはステップ2：ポリシーの設定で権限付与操作を追加または削除することができます。

被付与者

すべてのユーザー（匿名アクセス可）：匿名ユーザーに操作権限を開放したい場合は、この項目を選択します。次の手順でポリシーを設定する際に、すべてのユーザーが自動的に追加され、 `*` と表示されます。

指定ユーザー：指定するサブアカウント、ルートアカウントまたはクラウドサービスに操作権限を開放したい場合は、この項目を選択します。次の手順でポリシーを設定する際に、具体的なアカウントUINを指定する必要があります。

リソース範囲

バケット全体：バケットの設定項目に関連する権限を付与したい場合、またはリソース範囲としてバケット全体を指定したい場合は、この項目を選択します。次の手順でポリシーを設定する際に、バケット全体がリソースとして自動的に追加されます。

指定ディレクトリ：リソース範囲を指定のフォルダに限定したい場合は、この項目を選択します。次の手順でポリシーを設定する際に、具体的なディレクトリを指定する必要があります。

テンプレート

権限を付与したいアクションのグループです。COSはお客様が選択した被付与者およびリソース範囲に基づき、推奨するポリシーテンプレートをご提供します。テンプレートがニーズに合わない場合はこのステップをスキップするか、または次のステップの「ポリシーの設定」で権限付与操作を追加または削除することができます。

カスタムポリシー（プリセット設定はご提供していません）：テンプレートをご利用にならない場合はこの項目を選択し、次のステップの「ポリシーの設定」で、必要に応じてご自身でポリシーを追加できます。

その他のテンプレート：COSはお客様が選択した被付与者およびリソース範囲の組み合わせに基づき、それぞれに推奨するテンプレートをご提供します。必要なテンプレートにチェックを入れると、次のステップのポリシー設定で、COSが自動的に必要な操作を追加します。

テンプレートの説明は次の表をご参照ください。

被付与者	リソース範囲	ポリシーテンプレート	説明
すべての組み合わせ		カスタムポリシー	任意の被付与者、リソース範囲の組み合わせの場合、このテンプレートを選択するとプリセットポリシーのご提供は行われません。次のステップのポリシー設定で、ご自身でポリシーを直接追加することができます。
すべてのユーザー	バケット全体	読み取り専用オブジェクト（オブ	匿名ユーザーの場合、COSはファイル読み取り（ダウンロードなど）、ファイル書き込み（アップロード、変更など）の推奨テンプレートをご提供します。COSの推奨テンプレートには、バケット内の全オブジェクトのリストアップ、読み取り/書き込み権限、バケット設定などのその他の機密性の高い

（匿名アクセス可能）		ジェクトリストのリストアップを含まない）	権限は含まれず、他の余分な権限を開放しないようにすることで、データの安全性を高めています。必要に応じて、後のステップでご自身で動作権限を追加または削除することができます。
		読み取り/書き込みオブジェクト（オブジェクトリストのリストアップを含まない）	
	指定ディレクトリ	読み取り専用オブジェクト（オブジェクトリストのリストアップを含まない）	
		読み取り/書き込みオブジェクト（オブジェクトリストのリストアップを含まない）	
指定ユーザー	バケット全体	読み取り専用オブジェクト（オブジェクトリストの	指定ユーザーとバケット全体の組み合わせの場合、COSは最も多くの推奨テンプレートをご提供します。ファイル読み取り、書き込みおよびファイルリストアップのほか、次のような機密性の高い権限のテンプレートも含まれ、これらは信頼できるユーザーへの使用に適しています。 読み取り/書き込みバケットおよびオブジェクトACL：バケット取得、変更ACL、オブジェクトACL。GetObjectACL、PutObjectACL、GetBucketACL、

		リストアップを含まない）	PutBucketACLを含む バケット一般設定項目：バケットタグ、クロスドメイン、back-to-originなどの機密性が高くない権限。 バケットの機密性の高い設定項目：バケットポリシー、バケットACL、バケット削除などにかかわる機密性の高い権限であり、慎重に使用する必要があります。
		読み取り専用オブジェクト（オブジェクトリストのリストアップを含む）	
		読み取り/書き込みオブジェクト（オブジェクトリストのリストアップを含まない）	
		読み取り/書き込みオブジェクト（オブジェクトリストのリストアップを含む）	
		バケットおよびオブジェクトの読み取り/書き込みACL	
		バケット一般設定項目	

		バケットの機密性の高い設定項目	
	指定ディレクトリ	読み取り専用オブジェクト（オブジェクトリストのリストアップを含まない）	指定ユーザーと指定ディレクトリの組み合わせの場合、COSはファイル読み取り（ダウンロードなど）、ファイル書き込み（アップロード、変更など）以外にも、オブジェクトリストのリストアップ権限を含む推奨テンプレートをご提供します。指定したユーザー向けに指定したフォルダのファイルの読み取り、書き込み、リストアップ権限を開放したい場合は、この組み合わせを選択することをお勧めします。必要に応じて、後のステップでご自身で動作権限を追加または削除することができます。
		読み取り専用オブジェクト（オブジェクトリストのリストアップを含む）	
		読み取り/書き込みオブジェクト（オブジェクトリストのリストアップを含まない）	
		読み取り/書き込みオブジェクト（オブジェクトリストのリストアップを含む）	

ステップ2：ポリシーの設定

ステップ1で選択した被付与者、指定ディレクトリおよびテンプレートの組み合わせに対し、COSはポリシー設定において対応する操作、被付与者、リソースなどを自動的に追加します。お客様が指定ユーザー、指定ディレクトリを選択された場合は、ポリシー設定の際に具体的なユーザーUINとディレクトリを指定する必要があります。

説明：

ディレクトリに対する権限付与の場合、入力するリソースパスの後に `/*` を付ける必要があることにご注意ください。例えば、ディレクトリのtest権限の場合、 `test/*` と入力する必要があります。

COSがご提供する推奨テンプレートがニーズに合わない場合は、お客様ご自身でポリシーの内容を調整し、被付与者、リソースおよびアクションを追加または削除することもできます。下図に示します。

設定項目の説明は次のとおりです。

エフェクト：**許可**または**拒否**を選択できます。ポリシー構文の「allow」と「deny」に対応します。

ユーザー：すべてのユーザー(`*`)、ルートアカウント、サブアカウント、クラウドサービスを含む被付与者を追加、削除できます。

リソース：バケット全体または指定ディレクトリのリソースを追加できます。

アクション：権限付与が必要な操作を追加、削除します。

条件：ユーザーのアクセスIPの制限など、権限付与の際の指定条件です。

ポリシー文法

グラフィカル設定を使用する場合を除き、バケットポリシーに精通しているユーザーは、ターゲットバケットの**権限管理** > **Policy権限設定** > **ポリシー設定**で、直接JSON言語を使用してポリシーを作成することができます。

バケットポリシーを作成後は、APIまたはSDKによってバケットポリシーを追加することもできます。下図に示します。

JSONポリシーの例

次のポリシーの記述例は、ルートアカウントID 100000000001（APPIDは1250000000）下のサブアカウントID 100000000011に対し、北京リージョンのバケットのexamplebucket-bj下のディレクトリ `folder/sub-folder` 内のオブジェクトについて、すべての操作の権限を許可するものです。

```
{
  "Statement": [
    {
      "Principal": {
        "qcs": [
          "qcs::cam::uin/100000000001:uin/100000000011"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "name/cos:*"
      ],
      "Resource": [
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-bj-1250000000/folder/sub-
      ]
    }
  ],
  "version": "2.0"
}
```

操作方法

COSではコンソール、API、SDKなどの複数の方式でバケットポリシーを追加することができます。コンソールはグラフィカル設定と一般的な権限付与テンプレートをサポートしており、ポリシー言語に不慣れな場合でもすぐにポリシーを追加できるようになっています。

操作方法		説明
コンソール		直感的で使いやすいWebページ
API		RESTful APIで直接COSをリクエスト
SDK	JavaScript	豊富なSDK demoをご用意し、各種開発言語をサポートします。
	Node.js	
	小程序	

その他のバケットポリシーの例

バケットポリシー（Policy）による権限付与の例

他のルートアカウント下のサブアカウント操作名のバケットへの権限付与

注意：

バケットポリシーを追加する際は、必ず業務の必要性に応じて、最小権限の原則に従って権限を付与してください。他のユーザーに対し、すべてのリソース `(resource:*)` またはすべてのアクション `(action:*)` の権限を直接与えてしまうと、権限が大きすぎるためにデータセキュリティ上のリスクが生じる場合があります。

次に、サブネット、プリンシパル、VPC IDを制限するバケットポリシーの例についてご説明します。

事例1

サブネットの10.1.1.0/24ネットワークセグメントからのリクエストおよびvpcidがaqp5jrc1のリクエストを制限します。構文の例は次のとおりです。

```
{
  "Statement": [
    {
      "Action": [
        "name/cos:*"
      ],
      "Condition": {
        "ip_equal": {
          "qcs:ip": [
            "10.1.1.0/24"
          ]
        },
        "string_equal": {
          "vpc:requester_vpc": [
            "vpc-aqp5jrc1"
          ]
        }
      },
      "Effect": "deny",
      "Principal": {
        "qcs": [
          "qcs::cam::anyone:anyone"
        ]
      },
      "Resource": [
        "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
      ]
    }
  ],
  "version": "2.0"
}
```

事例2

vpcidがaqp5jrc1のリクエスト、特定のプリンシパルおよび特定のバケットのリクエストを制限します。構文の例は次のとおりです。

```
{
  "Statement": [
    {
      "Action": [
        "name/cos:*"
      ],
      "Condition": {
        "string_equal": {
          "vpc:requester_vpc": [
            "vpc-aqp5jrc1"
          ]
        }
      },
      "Effect": "allow",
      "Principal": {
        "qcs": [
          "qcs::cam::uin/100000000001:uin/100000000002"
        ]
      },
      "Resource": [
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/*"
      ]
    }
  ],
  "version": "2.0"
}
```

ユーザーポリシー

最終更新日：：2024-06-26 11:09:29

Tencent CloudルートアカウントはCloud Access Management（CAM）コンソールでCAMユーザーを作成し、ポリシーをバインドすることで、CAMユーザーに対しTencent Cloudのリソースを使用する権限を与えることができます。

概要

ユーザーはCAMで、ルートアカウント下の異なるタイプのユーザーに対し、異なる権限を付与することができます。これらの権限はアクセスポリシー言語で記述し、ユーザーを出発点として権限承認を行うため、**ユーザーポリシー**と呼ばれます。

ユーザーポリシーとバケットポリシーとの違い

ユーザーポリシーとバケットポリシーの最大の違いは、ユーザーポリシーはエフェクト（Effect）、アクション（Action）、リソース（Resource）、条件（Condition、オプション）のみを記述し、プリンシパル（Principal）は記述しない点です。このため、ユーザーポリシーの使用方法は次のようになります。

ユーザーポリシーは作成完了後、さらにサブユーザー、ユーザーグループまたはロールに対しバインド操作を実行する必要があります。

ユーザーポリシーは**アクションおよびリソース権限の匿名ユーザーへの付与をサポートしていません**。

プリセットポリシーとカスタムポリシー

ユーザーポリシーには、プリセットポリシーとカスタムポリシーの2種類が含まれます。プリセットポリシーを使用した権限のバインド、またはユーザーポリシーを自ら作成して権限のバインドを行うことができます。詳細については、CAMの権限承認ガイドをご参照ください。

ユースケース

ユーザーが行えること、および推奨されるユーザーポリシーについてお知りになりたい場合は、CAMユーザーを検索し、その所属するユーザーグループの権限を確認することで、ユーザーが何を行うことができるかを知ることができます。次のようなケースで推奨されます。

バケット作成（PutBucket）、バケットリストアップ（GetService）などの、Cloud Object Storage（COS）サービスレベルの権限を設定したい場合。

ルートアカウント下のすべてのCOSバケットおよびオブジェクトを使用したい場合。

ルートアカウント下の大量のCAMユーザーに同等の権限を付与したい場合。

ユーザーポリシー構文

ポリシー文法

バケットポリシーと同様に、ユーザーポリシーもJSON言語を使用して記述し、アクセスポリシー言語の統一ルール（プリンシパル、エフェクト、アクション、リソース、条件など）に従います。ただし、ユーザーポリシーはユーザー/ユーザーグループに直接バインドされるため、ユーザーポリシーではプリンシパル（Principal）の入力は不要です。

次の表はユーザーポリシーとバケットポリシーとの違いを比較したものです。

要素	ユーザーポリシー	バケットポリシー
プリンシパル	**入力不要**	入力必須
エフェクト	入力必須	入力必須
アクション	入力必須	入力必須
リソース	入力必須	**このバケット内のリソース**
条件	オプション	オプション

バケットポリシーの例

以下は、典型的なユーザーポリシーの例です。このポリシーは、広州にあるバケットexamplebucket-1250000000のすべてのCOS操作権限を付与するポリシーです。ポリシーを保存した後、さらにCAMのサブユーザー、ユーザーグループまたはロールにバインドすることで発効します。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["cos:*"],
      "Resource": [
          "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*",
          "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/"
      ]
    }
  ],
    "Version": "2.0"
}
```

ユーザーポリシーによるサブアカウントへのCOSアクセス権限承認

前提条件

CAMサブアカウントを作成済みであることが必要です。作成方法についてはサブアカウントの作成をご参照ください。

設定手順

CAMではプリセットポリシーとカスタムポリシーをご提供しています。プリセットポリシーはCAMの提供するシステムプリセットポリシーであり、COS関連ポリシーについてはプリセットポリシーをご覧ください。カスタムポリシーはユーザーがリソース、アクションなどの要素を自ら定義することができ、よりフレキシブルです。カスタムポリシーを新規作成し、サブアカウントへの権限承認を行う方法については以下でご説明します。

1. CAMコンソールにログインします。
2. **ポリシー > カスタムポリシーの新規作成 > ポリシー構文で作成**を選択し、ポリシー作成ページに進みます。
3. 実際のニーズに応じて**空白テンプレート**を選択して権限承認ポリシーをカスタマイズするか、またはCOSにバインドされた**システムテンプレート**を選択します。ここでは**空白テンプレート**を例にとります。
4. **空白テンプレート**を選択し、ポリシー構文を入力します。次の基本要素を含める必要があります。

resource：権限を承認するリソース。

すべてのリソース（ `"*"` ）

指定するバケット(`"qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"`)

バケット内の指定するディレクトリまたはオブジェクト(`"qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/test/*"`)

action：権限を承認するアクション。

effect：エフェクト。 `"allow"` （許可）または `"deny"` （拒否）を選択します。

condition：発効条件。オプションです。

COSはユーザーポリシーの例をご提供しています。次のドキュメントをご参照の上、ポリシーの内容をコピーして**ポリシー内容**のエディタボックス内に貼り付け、入力に間違いがないことを確認してから**完了**をクリックします。

サブアカウントのCOSアクセス権限の承認

COS API権限承認ポリシー使用ガイド

5. 作成完了後、CAMコンソールの**ポリシー > カスタムポリシー**で、作成したカスタムポリシーを確認し、ポリシーをサブアカウントにバインドすることができます。

6. サブアカウントにチェックを入れ、**OK**をクリックして権限を承認すると、限定されたCOSリソースにサブアカウントを使用してアクセスできるようになります。

プリセットポリシー

1. CAMではいくつかのプリセットポリシーをご提供しています。CAMコンソールの**ポリシー > プリセットポリシー**で確認し、「COS」を検索してフィルタリングすることができます。
2. ポリシー名をクリックし、**ポリシー構文 > JSON**に進み、具体的なポリシーの内容を確認します。プリセットポリシーのリソース（`resource`）はCOSのすべてのリソース(`"*"`)に設定されており、変更はサポートされていません。COSバケット、オブジェクトの一部について権限を承認したい場合は、JSONのプリセットポリシーをコピーし、カスタムポリシーを作成することができます。

表1と表2に、CAMがご提供するCOS関連のプリセットポリシーとその説明について列記します。

表1：COSプリセットポリシー

プリセットポリシー	説明	JSONポリシー
QcloudCOSBucketConfigRead	この権限を有するユーザーはCOSバケット設定を読み取ることができます	{ "version": "2.0", "statement": [{ "effect": "allow", "action": ["cos:GetBucket*", "cos:HeadBucket"], "resource": "*" }]}
QcloudCOSBucketConfigWrite	この権限を有するユーザーはCOS	{ "version": "2.0", "statement": [{ "effect": "allow", "action": ["cos:PutBucket*"], "resource": "*" }]}

	バケット設定を変更することができます	
QcloudCOSDataFullControl	COSバケット内のデータの読み取り、書き込み、削除、リストアップを含むアクセス権限	{ "version": "2.0", "statement": [{ "effect": "allow", "action": ["cos:GetService", "cos:GetBucket", "cos:ListMultipartUploads", "cos:GetObject*", "cos:HeadObject", "cos:GetBucketObjectVersions", "cos:OptionsObject", "cos:ListParts", "cos:DeleteObject", "cos:PostObject", "cos:PostObjectRestore", "cos:PutObject*", "cos:InitiateMultipartUpload", "cos:UploadPart", "cos:UploadPartCopy", "cos:CompleteMultipartUpload", "cos:AbortMultipartUpload", "cos:DeleteMultipleObjects", "cos:AppendObject"], "resource": "*" }]}

表2：COSの操作とプリセットポリシーとの関係

説明	操作	QcloudCOS Bucket ConfigRead	QcloudCOS Bucket ConfigWrite	QcloudCOS Data FullControl	QcloudCOS Data ReadOnly	Q D W
バケットリストアップ	GetService	✗	✗	✓	✗	✗

バケット作成	PutBucket	✕	✓	✕	✕	✕
バケット削除	DeleteBucket	✕	✕	✕	✕	✕
バケット基本情報取得	HeadBucket	✓	✕	✕	✕	✕
バケット設定項目取得	GetBucket*	✓	✕	✕	✕	✕
バケット設定項目変更	GetBucket*	✕	✓	✕	✕	✕
バケットアクセス権限取得	GetBucketAcl	✓	✕	✕	✕	✕
バケットアクセス権限変更	PutBucketAcl	✕	✓	✕	✕	✕
バケット内オブジェクトリストアップ	GetBucket	✓	✕	✓	✕	✕
バケット内オブジェクトの全バージョンリストアップ	GetBucketObjectVersions	✓	✕	✓	✕	✕
オブジェクトアップロード	PutObject	✕	✕	✓	✕	✓
マルチパート	ListParts InitiateMultipartUpload UploadPart	✕	✕	✓	✕	✓

アップロード	UploadPartCopy CompleteMultipartUpload AbortMultipartUpload ListMultipartUploads					
オブジェクト追加	AppendObject	✕	✕	✓	✕	✕
オブジェクトダウンロード	GetObject	✕	✕	✓	✓	✕
オブジェクトメタデータ確認	HeadObject	✕	✕	✓	✓	✕
クロスドメイン（CORS）事前チェック	OptionsObject	✕	✕	✓	✓	✕

その他のユーザーポリシーの例

サブアカウントのCOSアクセス権限の承認

COS API権限承認ポリシー使用ガイド

ACL

最終更新日：：2024-06-26 11:09:29

基本概念

アクセス制御リスト（ACL）はXML言語を使用して記述する、リソースにバインドされた、被付与者と付与される権限を指定したリストです。各バケットとオブジェクトにはそれぞれに、これらとバインドされたACLがあり、匿名ユーザーまたはその他のTencent Cloudのルートアカウントに対し、基本的な読み取り/書き込み権限を付与することができます。

注意：

リソースにバインドされたACLを使用した管理にはいくつかの制限があります。

リソースの所有者は常にリソースに対してFULL_CONTROL権限を有し、その取り消しまたは変更はできません。

匿名ユーザーはリソースの所有者にはなれません。この場合、オブジェクトリソースの所有者はバケットの作成者（Tencent Cloudルートアカウント）となります。

権限はCloud Access Management（CAM）のルートアカウントまたは事前定義済みのユーザーグループにのみ付与することができます。カスタマイズしたユーザーグループに権限を付与することはできず、サブユーザーへの権限付与も推奨されません。

権限の付加条件はサポートしていません。

明示的な拒否の権限はサポートしていません。

1つのリソースにつき最大100のACLポリシーを持つことができます。

ユースケース

注意：

匿名ユーザーにアクセスを開放すること（パブリック読み取り）はハイリスクな操作であり、トラフィックが不正使用される危険性があります。やむを得ずパブリック読み取りを使用する場合は、リンク不正アクセス防止の設定によってセキュリティ保護を実行できます。

バケットとオブジェクトに簡単なアクセス権限のみを設定したい場合、または匿名ユーザーにアクセスを開放したい場合はACLを選択できます。ただし、より多くの状況下では、バケットポリシーまたはユーザーポリシーの方が柔軟性が高いため、これらを優先的に使用することをお勧めします。ACLの適用ケースには次のものがあります。

簡単なアクセス権限のみを設定する場合。

コンソールでアクセス権限をすばやく設定する場合。

あるオブジェクト、ディレクトリまたはバケットへのアクセスをすべての匿名インターネットユーザーに開放したい場合は、ACLによる操作の方が便利です。

ACLの要素

被付与者 Grantee

サポートする被付与者のIDは、何らかのCAMルートアカウントまたは何らかの事前定義済みのCAMユーザーグループです。

注意：

他のTencent Cloudルートアカウントにアクセス権限を付与する際、この権限を付与されたルートアカウントはそのアカウント下のサブユーザー、ユーザーグループまたはロールにアクセス権限を与えることができます。

Cloud Object Storage（COS）は、匿名のユーザーまたはCAMユーザーグループにWRITE、WRITE_ACPまたはFULL_CONTROL権限を与えることを強く非推奨とします。一度権限を許可すると、ユーザーグループはお客様のリソースのアップロード、ダウンロード、削除などを行うことができるようになり、このことはお客様のデータの消失、料金引き落としなどのリスクをもたらす場合があります。

バケットまたはオブジェクトのACLにおいてサポートされるIDには次のものがあります。

クロスアカウント：ルートアカウントのIDを使用してください。アカウントIDは**アカウントセンター**のアカウント情報で取得します（例：100000000001）。

事前定義済みのユーザーグループ：URIタグを使用して事前定義済みのタグ付けをしたユーザーグループを使用してください。次のユーザーグループをサポートしています。

匿名ユーザーグループ - `http://cam.qcloud.com/groups/global/AllUsers` このグループは、リクエストが署名済みかどうかにかかわらず、誰でも権限なしにリソースにアクセスできることを表します。

認証ユーザーグループ - `http://cam.qcloud.com/groups/global/AuthenticatedUsers` このグループは、Tencent Cloud CAMアカウント認証を経たすべてのユーザーがリソースにアクセスできることを表します。

操作 Permission

Tencent Cloud COSがリソースのACLにおいてサポートする操作は、実際には一連の操作の集合であり、バケットおよびオブジェクトACLにはそれぞれ異なる意味があります。

バケットの操作

次の表に、バケットACLで設定可能な操作のリストを列記しています。

操作セット	説明	許可される行為
READ	オブジェクトのリストアップ	HeadBucket，GetBucketObjectVersions，ListMultipartUploads
WRITE	オブジェクトのアップロード、上書き、削除	PutObject，PutObjectCopy，PostObject，InitiateMultipartUpload， UploadPart，UploadPartCopy，CompleteMultipartUpload， DeleteObject

READ_ACP	バケットのACLの読み取り	GetBucketAcl
WRITE_ACP	バケットのACLの書き込み	PutBucketAcl
FULL_CONTROL	上記4種類の権限のセット	上記のすべての行為のセット

注意：

バケットのWRITE、WRITE_ACPまたはFULL_CONTROL権限の付与は慎重に行ってください。バケットのWRITE権限の付与は、被付与者に既存のあらゆるオブジェクトの上書きまたは削除を許可するものです。

オブジェクトの操作

次の表に、オブジェクトACLで設定可能な操作のリストを列記しています。

操作セット	説明	許可される行為
READ	オブジェクトの読み取り	GetObject，GetObjectVersion，HeadObject
READ_ACP	オブジェクトのACLの読み取り	GetObjectAcl，GetObjectVersionAcl
WRITE_ACP	オブジェクトのACLの書き込み	PutObjectAcl，PutObjectVersionAcl
FULL_CONTROL	上記3種類の権限のセット	上記のすべての行為のセット

説明：

オブジェクトについてはWRITE操作セットの権限はサポートしていません。

既定ACL

COSでは一連の既定ACLによる権限承認がサポートされており、権限を簡単に記述できるようになっています。既定ACLを使用して記述する場合、PUT Bucket/ObjectまたはPUT Bucket/Object aclにx-cos-aclヘッダーを含め、必要な権限を記述する必要があります。同時にリクエスト本文にもXMLの記述内容が含まれる場合は、ヘッダー内の記述が優先的に選択され、リクエスト本文のXMLの記述は無視されます。

バケットの既定ACL

規定名	説明
private	作成者（ルートアカウント）はFULL_CONTROL権限を有し、その他の人は権限を持ちません（デフォルト）
public-read	作成者はFULL_CONTROL権限を有し、匿名ユーザーグループはREAD権限を有します
public-read-write	作成者と匿名ユーザーグループの両方がFULL_CONTROL権限を有します。通常はこの権限の付与は非推奨です

authenticated-read	作成者はFULL_CONTROL権限を有し、認証ユーザーグループはREAD権限を有します

オブジェクトの既定ACL

規定名	説明
default	空の記述であり、この場合は各レベルのディレクトリに基づく明示的な設定およびバケットの設定によって、リクエストを許可するかどうかを決定します（デフォルト）
private	作成者（ルートアカウント）はFULL_CONTROL権限を有し、その他の人は権限を持ちません
public-read	作成者はFULL_CONTROL権限を有し、匿名ユーザーグループはREAD権限を有します
authenticated-read	作成者はFULL_CONTROL権限を有し、認証ユーザーグループはREAD権限を有します
bucket-owner-read	作成者はFULL_CONTROL権限を有し、バケット所有者はREAD権限を有します
bucket-owner-full-control	作成者とバケット所有者の両方がFULL_CONTROL権限を有します

説明：

オブジェクトについてはpublic-read-write権限はサポートしていません。

事例

バケットのACL

バケットを作成する際、COSはデフォルトのACLを作成し、リソース所有者に対しリソースの完全制御権限（FULL_CONTROL）を与えます。その例を次に示します。

```
<AccessControlPolicy>
  <Owner>
    <ID>Owner-Cononical-CAM-User-Id</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>Owner-Cononical-CAM-User-Id</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

オブジェクトのACL

オブジェクトを作成する際、COSはデフォルトではACLを作成しません。この場合はオブジェクトの所有者がバケット所有者となります。オブジェクトはバケットの権限を継承し、それはバケットのアクセス権限と一致します。オブジェクトにはデフォルトのACLがないため、Bucket Policyの中のアクセス者およびその行為に対する定義に従って、リクエストが許可されるかどうかを判断します。詳細については、アクセスポリシーの言語概要のドキュメントをご参照ください。

オブジェクトについてその他のアクセス権限を付与したい場合は、これを基本としてさらにACLを追加し、オブジェクトのアクセス権限を記述することができます。例えば匿名ユーザーに対し、単一オブジェクトの読み取り専用の権限を付与する場合の例は次のとおりです。

```
<AccessControlPolicy>
  <Owner>
    <ID>Owner-Cononical-CAM-User-Id</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>Owner-Cononical-CAM-User-Id</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
```

```
      <Grant>
        <Grantee>
          <URI>http://cam.qcloud.com/groups/global/AllUsers</URI>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </AccessControlList>
  </AccessControlPolicy>
```

タグに基づくプロジェクトリソースの管理

最終更新日：：2024-06-26 11:09:29

概要

説明：

このドキュメントでは主に、タグシステムの下でのタグおよびタグ認証によるプロジェクトリソース管理の利用方法についてご説明します。過去のバージョンのコンソール上でプロジェクトを使用したことがあり、かつプロジェクト認証方式でサブアカウントのアクセス権限を付与したことがある、一部の古くからのユーザーに適用します。

プロジェクト管理はプロジェクトの次元でリソースに対する集中管理を行うものです。プロジェクトの機能をサポートしているクラウド製品リソースをプロジェクトに追加し、Cloud Access Management（CAM）コンソールの**ポリシー>カスタムポリシーの新規作成>製品機能またはプロジェクトによる権限の作成**によってプロジェクトポリシーを生成することができます。プロジェクトポリシーをプロジェクトに関連するユーザーまたはユーザーグループにバインドすることで、ユーザーまたはユーザーグループにプロジェクトリソースの操作権限を許可することができます。

Cloud Object Storage（COS）は過去のバージョンのコンソール上で、プロジェクトベースでユーザーに関連する権限管理操作を提供していますが、プロジェクトポリシーにはプロジェクトに追加された全製品下の全リソースの完全なアクセス権限が含まれ、**多次元的なタグ付けおよびカテゴライズのニーズを満たすことができない**だけでなく、**精密な権限管理を行うこともできません**でした。新バージョンのCOSコンソールでは、COSはタグ方式によるプロジェクトリソースの権限管理のみサポートしています。

COSはタグサービスを利用して旧プロジェクト機能との互換性を実現しています。タグサービスのシステムでは、プロジェクトは特殊なタグであり、そのタグキーは `project` となります。プロジェクトコンソールでプロジェクトを新規作成し、そのプロジェクト下にバケットを作成することも引き続き可能です。COSはお客様のバケット作成時にバケットのプロジェクトの帰属関係を自動的にタグにダブルライトすることで、コンソール上で表示できるようにします。

説明：

バケットのカテゴリー管理のニーズがおありの場合は、プロジェクトの手段ではなく、直接タグによってバケットを管理することで、権限制御や請求書分割などのタスクを実現することをお勧めします。コンソール上でのタグの追加方法に関しては、バケットタグの設定をご参照ください。

プロジェクトについてお知りになりたい場合は、CAMのプロジェクトとタグをご参照ください。タグサービスについてお知りになりたい場合は、タグの製品ドキュメントをご参照ください。

タグのメリットについてお知りになりたい場合は、タグ使用のメリットをご参照ください。

サブアカウントへのプロジェクトアクセス権限の付与

サブアカウントに対してプロジェクトへのアクセス権限を付与するには、次の手順に従って操作を行ってください。

1. プロジェクト管理コンソールにログインし、プロジェクトを新規作成し、プロジェクト名をカスタマイズして送信します。その後、そのプロジェクト下に作成したバケットまたはCVMなどのリソースを選択します。
すでにプロジェクトがあり、かつすでに帰属するストレージまたはコンピューティングリソースがある場合は、この手順をスキップできます。
2. プロジェクトの作成が完了し、対応するリソースをバインドした後、ポリシー管理ページに進み、**カスタムポリシーの新規作成>タグによる権限承認**をクリックし、追加するCOSサービスおよび承認する操作権限を選択します。対応するプロジェクトのタグを選択し、サブアカウントに対し、このプロジェクトタグ下のすべてのリソースへのアクセスを承認します。
3. **次のステップ**をクリックし、この権限を承認するユーザー/ユーザーグループ/ロールを選択し、最後に**完了**をクリックします。

説明：

デフォルトのポリシー内容は、サブアカウントに対しこのプロジェクトタグ下のすべてのリソースへのアクセスを付与するものです。ユーザーがタグ下の一部のリソースについてのみ、指定した操作を行えるようにしたい場合は、構文の構造ドキュメントを参照し、ポリシー構文の中の `action` （指定した操作の設定）および `resource` （操作可能なリソースの設定）を変更することができます。

サブアカウントでバケットを作成できるようにしたい場合は、サブアカウントに対しさらに `PUT Bucket` の操作権限を付与する必要があります。ポリシー管理ページで、**カスタムポリシーの新規作成>ポリシージェネレーターで作成**をクリックすると、サブアカウントに対し対応する権限を付与することができます。

権限承認方式の選択方法

最終更新日：：2024-06-26 11:09:29

バケットポリシー、ユーザーポリシー、バケットACL、オブジェクトACLの違い

ユーザーポリシー、バケットポリシー、バケットACL、オブジェクトACLの間の違いについては表1のとおりです。

ユーザーポリシーはユーザーベースの権限承認方式であり、バケットポリシー、バケットACL、オブジェクトACLはリソースベースの権限承認方式です。

ユーザーポリシーとバケットポリシーの権限承認はアクセスポリシー言語に基づいて行われ、権限制御の柔軟性の程度がより高くなっています。権限はそれぞれのアクション（action）について具体的に付与され、なおかつ権限のエフェクト（effect）をdenyまたはallowとすることができます。バケットACLとオブジェクトACLはどちらもアクセス制御リスト（ACL）をベースにした権限承認であり、権限制御の柔軟性の程度は相対的に低く、より簡単に使用することができますが、サポートされる権限は基本的な読み取り/書き込み権限のみです。

ユーザーポリシーは匿名ユーザーへの権限承認をサポートしていません。バケットポリシー、バケットACL、オブジェクトACLは匿名ユーザーへの権限承認をサポートしています。

ユーザーポリシーはCloud Access Management（CAM）コンソールで設定し、バケットポリシー、バケットACL、オブジェクトACLはCloud Object Storage（COS）コンソールで設定します。

表1 権限承認方式の違いの比較

比較項目		ユーザーポリシー	バケットポリシー	バケットACL	オブジェクトACL
分類		ユーザーベースの権限承認	リソースベースの権限承認	リソースベースの権限承認	リソースベースの権限承認
権限制御の柔軟性の程度		柔軟性が高い	柔軟性が高い	柔軟性が低い	柔軟性が低い
コンソール設定		CAMコンソール	COSコンソール	COSコンソール	COSコンソール
アクセス制御要素	ユーザー	このアカウントが管理するすべてのCAM ID（サブアカウント、ロールなど）	サブアカウント、ルートアカウント、匿名ユーザー	サブアカウント、他のルートアカウント、匿名ユーザー	サブアカウント、他のルートアカウント、匿名ユーザー
			サブアカウント		

			のTencent Cloudサービス、ロールなど		
		クロスアカウント権限承認を行う場合は先にコラボレーターとしての追加が必要	クロスアカウント権限承認を直接サポート		
	エフェクト	許可+拒否	許可+拒否	許可のみ	許可のみ
	リソース	すべてのリソース、COSのすべてのバケット、指定のバケット（プレフィックス、オブジェクトなど）	指定のバケット（プレフィックス、オブジェクトなど）	バケット全体	指定のオブジェクト
	アクション	具体的な各アクション	具体的な各アクション（バケット作成、バケットリストアップを除く）	簡略化された読み取り/書き込み権限	簡略化された読み取り/書き込み権限
	条件	サポートしています	サポートしています	サポートしていません	サポートしていません

適切な権限承認方式を選択するにはどうすればよいですか。

表1に列記した違いから、それぞれの権限承認方式の優劣を判断し、お客様ご自身の必要性に応じて適切な権限承認方式をご選択いただくことができます。
ただし、どの方式を選択する場合でも、できる限り統一性を保つことをお勧めします。バケットポリシー、ユーザーポリシー、ACLの数が増えるにつれて、権限の維持管理が難しくなります。

ACLを選択するのはどのような場合ですか。

注意：

匿名ユーザーにアクセスを開放すること（パブリック読み取り）はハイリスクな操作であり、トラフィックが不正使用される危険性があります。やむを得ずパブリック読み取りを使用する場合は、リンク不正アクセス防止の設定によってセキュリティ保護を実行できます。

バケットとオブジェクトに簡単なアクセス権限のみを設定したい場合、または匿名ユーザーにアクセスを開放したい場合はACLを選択できます。ただし、より多くの状況下では、バケットポリシーまたはユーザーポリシーの方が柔軟性が高いため、これらを優先的に使用することをお勧めします。

簡単なアクセス権限のみを設定する場合。

コンソールでアクセス権限をすばやく設定する場合。

あるオブジェクト、ディレクトリまたはバケットへのアクセスをすべての匿名インターネットユーザーに開放したい場合は、ACLによる操作の方が便利です。

各アカウント下に設定できるACLの数は1000個までです。この上限を超える場合はバケットポリシーまたはユーザーポリシーを代わりに選択してください。

ユーザーポリシーを選択するのはどのような場合ですか。

ユーザーが行えることについてお知りになりたい場合は、ユーザーポリシーを推奨します。CAMユーザーを検索し、その所属するユーザーグループの権限を確認することで、ユーザーが何を行うことができるかを知ることができます。次のようなケースで推奨されます。

バケット作成、バケットリストアップなどの、COSサービスレベルの権限を設定したい場合。

ルートアカウント下のすべてのCOSバケットおよびオブジェクトを使用したい場合。

ルートアカウント下の大量のCAMユーザーに同等の権限を付与したい場合。

バケットポリシーを選択するのはどのような場合ですか。

そのCOSバケットに誰がアクセス可能かをお知りになりたい場合は、バケットポリシーの使用をお勧めします。バケットを検索し、バケットポリシーをチェックすることで、アクセス可能な人が誰かを知ることができます。次のようなケースで推奨されます。

特定のバケットについて単独で権限を付与する場合。

ACLと異なり、権限は特定のアクション（action）について具体的に付与され、権限のエフェクト（effect）をdenyまたはallowとする必要があります。

ユーザーポリシーと異なり、バケットポリシーはクロスアカウント権限承認および匿名ユーザーへの権限承認をサポートしています。

アクセスポリシー言語

アクセスポリシーの言語概要

最終更新日：：2024-06-26 11:09:29

注意：

サブユーザーまたはコラボレーターにアクセスポリシーを追加する際は、必ず業務の必要性に応じて、最小権限の原則に従って権限を付与してください。サブユーザーまたはコラボレーターに対し、すべてのリソース `(resource:*)` またはすべてのアクション `(action:*)` の権限を直接与えてしまうと、権限の範囲が大きすぎるためにデータセキュリティ上のリスクが生じる場合があります。

概要

アクセスポリシーはCloud Object Storage（COS）リソースへのアクセス権限を付与するために用いられます。アクセスポリシーにはJSONをベースにしたアクセスポリシー言語を使用します。アクセスポリシー言語の権限によって、指定するプリンシパル（principal）に指定のCOSリソースに対する指定のアクションを実行させることができます。

アクセスポリシー言語はバケットポリシー（Bucket Policy）の基本要素および使用法を記述するものです。ポリシー言語の説明に関してはCAMポリシー管理をご参照ください。

アクセスポリシーの要素

アクセスポリシー言語には次の基本的な意味を持つ要素が含まれます。

プリンシパル（principal）：ポリシーが権限を付与するエンティティを記述します。例えばユーザー（ルートアカウント、サブアカウント、匿名ユーザー）、ユーザーグループなどです。この要素はバケットアクセスポリシーに対して有効ですが、ユーザーアクセスポリシーには追加すべきではありません。

ステートメント（statement）：1つまたは複数の権限の詳細情報を記述します。この要素にはエフェクト、アクション、リソース、条件などのいくつかの他の要素の権限または権限セットが含まれます。1つのポリシーには1つのステートメント要素しかありません。

エフェクト（effect）：ステートメントによる結果が「許可」であるか「明示的な拒否」であるかを記述します。allowとdenyの2種類が含まれます。この要素は入力必須項目です。

アクションaction：許可する、または拒否するアクションを記述します。アクションはAPI（nameプレフィックスで記述）または機能セット（permidプレフィックスが付いた特定のAPIセット）とすることができます。この要素は入力必須項目です。

リソース（resource）は権限が適用されるリソースについて記述します。リソースは6セグメント式で説明されます。リソース定義の詳細は各製品によって異なります。リソースの指定方法については、作成したリソース

テートメントに対応する製品ドキュメントをご参照ください。この要素は必須項目です。
条件（condition）：ポリシー発効の制約条件を記述します。条件はオペレーター、アクションキーとアクション値から構成されています。条件値には時間、IPアドレスなどの情報を含めることができます。一部のサービスは、条件に対しほかの値を指定することを認めています。この要素は入力必須項目ではありません。

要素の使用法

プリンシパルの指定

プリンシパル（principal）の要素はリソースへのアクセスを許可される、または拒否されるユーザー、アカウント、サービスまたはその他のエンティティを指定するために用いられます。principalの要素はバケット内でのみ作用します。ユーザーポリシーでは指定する必要がありませんが、これはユーザーポリシーが特定のユーザーに直接付加されるものだからです。principalの指定の例は次のとおりです。

```
"principal":{
"qcs": [
"qcs::cam::uin/100000000001:uin/100000000001"
]
}
```

匿名ユーザーに権限を付与：

```
"principal":{
"qcs": [
"qcs::cam::anonymous:anonymous"
]
}
```

ルートアカウントUIN 100000000001に権限を付与：

注意：

```
"principal":{
"qcs": [
"qcs::cam::uin/100000000001:uin/100000000001"
]
}
```

サブアカウントUIN 100000000011（ルートアカウントのUINは100000000001）に権限を付与：

注意：

操作を行う前に、サブアカウントがルートアカウントのサブアカウントリストに追加されていることを確認する必要があります。

```
"principal":{
"qcs": [
"qcs::cam::uin/100000000001:uin/100000000011"
]
}
```

エフェクトの指定

リソースへのアクセス権限を明示的に付与（許可）していない場合、アクセスは暗黙的に拒否されます。リソースへのアクセスを明示的に拒否（deny）することもでき、そうした場合はユーザーがそのリソースに確実にアクセ

スできなくなります。これは他のポリシーがアクセス権限を付与している場合であっても同様です。許可のエフェクトを指定する例を次に挙げます。

```
"effect" : "allow"
```

アクションの指定

COSはポリシーの中で、ある特定のCOSアクションを指定することができると定義しています。指定するアクションは発行されるAPIのリクエスト操作 と完全に同じです。一部のバケット操作およびオブジェクト操作を次に列記します。その他の具体的な操作については、API操作リストのドキュメントをご参照ください。

バケットの操作

説明	対応するAPIインターフェース
name/cos:GetService	GET Service
name/cos:GetBucket	GET Bucket (List Objects)
name/cos:PutBucket	PUT Bucket
name/cos:DeleteBucket	DELETE Bucket

オブジェクト操作

説明	対応するAPIインターフェース
name/cos:GetObject	GET Object
name/cos:PutObject	PUT Object
name/cos:HeadObject	HEAD Object
name/cos:DeleteObject	DELETE Object

アクション許可の指定の例を次に示します。

```
"action":[
"name/cos:GetObject",
"name/cos:HeadObject"
]
```

リソースの指定

リソース（resource）要素は単一または複数の操作オブジェクト（COSバケットまたはオブジェクトなど）を記述します。すべてのリソースには下記の6セグメント式の記述方法を使用することができます。

```
qcs:project_id:service_type:region:account:resource
```

パラメータの説明は次のとおりです。

パラメータ	説明	入力必須かどうか
qcs	qcloud serviceの略称であり、Tencent Cloudのクラウドサービスであることを表します。	はい
project_id	プロジェクト情報を記述します。CAMのレガシーロジックとの互換性のため	オプション

	にのみ使用されます。	
service_type	「COS」のような、製品の略称を記述します。	はい
region	リージョンの情報を説明します。Tencent Cloud COSがサポートするアベイラビリティリージョンをご参照ください。	はい
account	リソース所有者のルートアカウント情報を記述します。現在、リソース所有者の記述方式は2種類をサポートしています。1つはuin方式、すなわちルートアカウントのUINアカウントであり、 `uin/${OwnerUin}` で表され、uin/100000000001のようになります。もう1つはuid方式、すなわちルートアカウントのAPPIDであり、 `uid/${appid}` で表され、uid/1250000000のようになります。現在、COSのリソース所有者はすべてuid方式を使用して記述され、ルートアカウントの開発者APPIDとなります。	はい
resource	具体的なリソースの詳細を記述します。COSサービスではバケットのXML APIアクセスドメイン名を使用して記述します。	はい

バケットexamplebucket-1250000000を指定する例を次に挙げます。

```
"resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"]
```

バケットexamplebucket-1250000000内の/folder/フォルダ下の全オブジェクトを指定する例を次に挙げます。

```
"resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/folder/
```

バケットexamplebucket-1250000000内のオブジェクト、/folder/exampleobjectを指定する例を次に挙げます。

```
"resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/folder/
```

条件の指定

アクセスポリシー言語は権限承認を受ける際の条件を指定することができます。例えばユーザーのアクセス元の制限、権限承認時間の制限などです。現在サポートされている条件オペレーターのリストおよび一般的な条件キーとその例などの情報を次に列記します。

条件オペレーター	意味	条件名	例

ip_equal	IP一致	qcs:ip	{"ip_equal":{"qcs:ip ":"10.121.2.0/24"}}
ip_not_equal	IP不一致	qcs:ip	{"ip_not_equal":{"qcs:ip ":["10.121.1.0/24", "10.121.2.0/24"]}}

以下は、アクセスIPが10.121.2.0/24のネットワークセグメント内という条件を満たす例です。

```
"ip_equal":{"qcs:ip ":"10.121.2.0/24"}
```

以下は、アクセスIPが101.226.100.185および101.226.100.186であるという条件を満たす例です。

```
"ip_equal":{
    "qcs: ip": [
        "101.226.100.185",
        "101.226.100.186"
    ]
}
```

実際の例

ルートアカウントが匿名ユーザーを許可し、アクセス元のIPが101.226.100.185または101.226.100.186であった場合、華南リージョンのバケットexamplebucket-1250000000内のオブジェクトに対し、GET（ダウンロード）およびHEAD操作を実行し、認証は必要ありませんでした。その他の事例については、権限設定の関連事例をご参照ください。

```
{
    "version": "2.0",
    "principal":{
        "qcs": [
            "qcs: : cam: : anonymous: anonymous"
        ]
    },
    "statement":[
        {
            "action":[
                "name/cos: GetObject",
                "name/cos: HeadObject"
            ],
            "condition":{
                "ip_equal":{
                    "qcs: ip": [
                        "101.226.100.185",
                        "101.226.100.186"
                    ]
                }
            },
            "effect": "allow",
            "resource":[
                "qcs: : cos: ap-guangzhou: uid/1250000000: examplebucket-1250000000
            ]
        }
    ]
}
```

発効条件

最終更新日：：2024-06-26 11:09:29

概要

発効条件とはアクセスポリシー言語の一部であり、完全な発効条件には次の要素が含まれます。

条件キー：発効条件の具体的な種類を表します。例えば、ユーザーのアクセス元のIP、権限承認時間などです。

条件オペレーター：発効条件の判断方法を表します。

条件値：条件キーの値です。

詳細についてはCAM発効条件をご参照ください。

説明：

条件キーを使用してポリシーを作成する際は、必ず最小権限の原則を遵守し、適用可能なリクエスト（action）にのみ該当する条件キーを追加してください。アクション（action）を指定する際にワイルドカード「*」を使用すると、リクエストが失敗しますので避けてください。

Cloud Access Management（CAM）コンソールを使用してポリシーを作成する際は、構文形式にご注意ください。version、principal、statement、effect、action、resource、conditionの構文要素はアルファベットの先頭の文字を大文字にするか、またはすべて小文字にする必要があります。

発効条件の例

次のバケットポリシーの例における発効条件（condition）は、ユーザーが10.217.182.3/24または111.21.33.72/24ネットワークセグメントに属している場合にのみ、 `cos:PutObject` アクションの権限付与が完了することを表します。このうち、

条件キーは `qcs:ip` であり、発効条件の種類がIPであることを表します。

条件オペレーターは `ip_equal` であり、発効条件の判断方法がIPアドレスの同一性であることを表します。

条件値は配列 `["10.217.182.3/24","111.21.33.72/24"]` であり、発効条件判断の規定値を表します。ユーザーが配列の中の任意のIPがあるネットワークセグメントに属している場合、条件判断はすべてtrueとなります。

```
{
    "version":"2.0",
    "statement":[
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"allow",
            "action":[
                "name/cos:PutObject"
            ],
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ],
            "condition":{
                "ip_equal":{
                    "qcs:ip":[
                        "10.217.182.3/24",
                        "111.21.33.72/24"
                    ]
                }
            }
        }
    ]
}
```

COSのサポートする条件キー

説明：

条件キー `tls-version` は現在北京リージョンでのみサポートしています。その他のリージョンでも順次サポート予定です。

Cloud Object Storage（COS）のサポートする条件キーには2種類あり、1つはIP、VPC、HTTPSを含むすべてのリクエストに適用可能なもので、もう1つはリクエストヘッダーおよびリクエストパラメータによる条件キーであり、一般的にこのリクエストヘッダーまたはリクエストパラメータを持つリクエストにのみ適用できます。これらの条件キーに関する説明および実際の使用例については、条件キーの説明およびユースケースのドキュメントをご参照ください。

説明：

発効条件、条件キーなどの概念はすべてユーザーリクエストを対象としてCAMが実行するものです。ライフサイクル、バケットコピールールが有効になっている場合の削除、コピーなどの動作はCOSが行うものであり、ユーザーによるリクエストではないため、条件キーがライフサイクル、バケットコピールールに対して起こす動作は無効となります。

すべてのリクエストに適用可能な条件キー

1種類目はすべてのリクエストに適用可能な条件キーです。 `qcs:ip` 、 `qcs:vpc` および `cos:secure-transport` が含まれ、それぞれ、リクエスト元のIPネットワークセグメント、VPC ID、HTTPSプロトコルを使用しているかどうかを表します。すべてのリクエストが使用できます。

条件キー	適用リクエスト	意味	タイプ
cos:secure-	すべてのリクエスト	リクエストにHTTPSプロトコルが適用されている	Boolean

transport		かどうか確認	
qcs:ip	すべてのリクエスト	リクエスト元のIPネットワークセグメント	IP
qcs:vpc	すべてのリクエスト	リクエスト元のVPC ID	String
cos:tls-version	すべてのhttpsリクエスト	httpsリクエストに使用しているTLSバージョン	Numeric

リクエストヘッダーおよびリクエストパラメータによる条件キー

2種類目は、リクエストヘッダー（Header）およびリクエストパラメータ（Param）による条件キーです。リクエストごとにリクエストヘッダーおよびリクエストパラメータが異なるため、これらの条件キーは一般的にこの種類のヘッダーまたはリクエストパラメータが含まれるリクエストにのみ適します。
例えば、条件キー `cos:content-type` は、リクエストヘッダー `Content-Type` を使用する必要があるアップロードクラスのリクエスト（PutObjectなど）に適用できます。条件キー `cos:response-content-type` はGetObjectオブジェクトにのみ適用できます。このリクエストのみがリクエストパラメータ `response-content-type` をサポートしているためです。
COSが現在サポートしている、リクエストヘッダーおよびリクエストパラメータによる条件キーと、それらの条件キーが適用可能なリクエストは下表のとおりです。

条件キー	適用リクエスト	リクエストヘッダー/リクエストパラメータの確認	タイプ
cos:x-cos-storage-class	PutObject PostObject InitiateMultipartUpload AppendObject	リクエストヘッダー：x-cos-storage-class	String
cos:versionid	GetObject DeleteObject PostObjectRestore PutObjectTagging GetObjectTagging DeleteObjectTagging HeadObject	リクエストパラメータ：versionid	String
cos:prefix	GetBucket（List Objects） GET Bucket Object versions List Multipart Uploads ListLiveChannels	リクエストパラメータ：prefix	String
cos:x-cos-acl	PutObject PostObject PutObjectACL	リクエストヘッダー：x-cos-acl	String

	PutBucket PutBucketACL AppendObject Initiate Multipart Upload		
cos:content-length	このリクエストは適用範囲が広いため、リクエストボディ付きのリクエストなどの代表的なリクエストに注目	リクエストヘッダー：Content-Length	Numeric
cos:content-type	このリクエストは適用範囲が広いため、リクエストボディ付きのリクエストなどの代表的なリクエストに注目	リクエストヘッダー：Content-Type	String
cos:response-content-type	GetObject	リクエストパラメータ：response-content-type	String
qcs:request_tag	PutBucket PutBucketTagging	リクエストヘッダー：x-cos-tagging リクエストパラメータ：tagging	String

条件オペレーター

COSの条件キーは次の条件オペレーターをサポートしており、文字列（String）、数値型（Numeric）、ブール型（Boolean）およびIPなどの様々なタイプの条件キーに適用できます。

条件オペレーター	意味	タイプ
string_equal	文字列一致（大文字と小文字を区別）	String
string_not_equal	文字列不一致（大文字と小文字を区別）	String
string_like	文字列が類似（大文字と小文字を区別）。現在は文字列の前後へのワイルドカード `*` の追加をサポートしています（例： `image/*` ）	String
ip_equal	IP一致	IP
ip_not_equal	IP不一致	IP
numeric_equal	数値一致	Numeric
numeric_not_equal	数値不一致	Numeric
numeric_greater_than	数値が大きい	Numeric

numeric_greater_than_equal	数値が同じか大きい	Numeric
numeric_less_than	数値が小さい	Numeric
numeric_less_than_equal	数値が同じか小さい	Numeric

_if_existの意味

上記のすべての条件オペレーターは、その後に `_if_exist` を追加することで単一条件オペレーターとすることができます。例えば、 `string_equal_if_exist` などです。条件オペレーターに `_if_exist` が含まれるかどうかで異なる点は、リクエストに、条件キーに対応するリクエストヘッダーまたはリクエストパラメータが含まれない場合にどのように処理されるかの違いです。

条件オペレーターに `_if_exist` が含まれない場合、例えば `string_equal` の場合は、リクエストに対応するリクエストヘッダー/リクエストパラメータが含まれないとき、デフォルトで条件にヒットし、 `False` となります。

条件オペレーターに `_if_exist` が含まれる場合、例えば `string_equal_if_exist` の場合は、リクエストに対応するリクエストヘッダー/リクエストパラメータが含まれないとき、デフォルトで条件にヒットし、 `True` となります。

事例

事例1：指定されたオブジェクトバージョンのダウンロードを許可

例えば、次のバケットポリシーについて、エフェクトがallowの場合は、リクエストパラメータversionidによる“MTg0NDUxNTc1NjIzMTQ1MDAwODg”のGetObjectリクエストの承認が許可されることを表します。条件にヒットした場合（True）、allowの権限付与ポリシーに基づいてリクエストは承認されます。条件にヒットしなかった場合（False）、allowの権限付与ポリシーに基づいて、リクエストは権限を得られず、リクエストは失敗します。

```
{
    "version":"2.0",
    "statement":[
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"allow",
            "action":[
                "name/cos:GetObject"
            ],
            "condition":{
                "string_equal":{
                    "cos:versionid":"MTg0NDUxNTc1NjIzMTQ1MDAwODg"
                }
            },
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ]
        }
    ]
}
```

条件オペレーターが `string_equal` または `string_equal_if_exist` の場合、conditionのヒット状況およびリクエストが承認されるかどうかは下表のとおりとなります。

条件オペレーター	リクエスト	conditionにヒットするか	リクエストが承認されるか
string_equal	versionidなし	FALSE	不承認
string_equal_if_exist	versionidなし	TRUE	承認
string_equal	versionid付き、指定のもの	TRUE	承認
string_equal_if_exist	versionid付き、指定のもの	TRUE	承認
string_equal	versionid付き、指定のもの以外	FALSE	不承認
string_equal_if_exist	versionid付き、指定のもの以外	FALSE	不承認

事例2：指定されたオブジェクトバージョンのダウンロードを拒否

次のバケットポリシーの例では、エフェクトがdenyであり、リクエストパラメータversionidによる「MTg0NDUxNTc1NjIzMTQ1MDAwODg」のGetObjectリクエストが拒否されることを表します。条件にヒットした場合（True）、denyの権限付与ポリシーに基づいてリクエストは失敗します。条件にヒットしなかった場合（False）、denyの権限付与ポリシーに基づいて、リクエストは拒否されません。

```
{
    "version":"2.0",
    "statement":[
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"deny",
            "action":[
                "name/cos:GetObject"
            ],
            "condition":{
                "string_equal":{
                    "cos:versionid":"MTg0NDUxNTc1NjIzMTQ1MDAwODg"
                }
            },
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ]
        }
    ]
}
```

条件オペレーターが `string_equal` または `string_equal_if_exist` の場合、conditionのヒット状況およびリクエストが拒否されるかどうかは下表のとおりとなります。

条件オペレーター	リクエスト	conditionにヒットするか	リクエストが拒否される/拒否されない
string_equal	versionidなし	FALSE	拒否されない
string_equal_if_exist	versionidなし	TRUE	拒否される
string_equal	versionid付き、指定のもの	TRUE	拒否される
string_equal_if_exist	versionid付き、指定のもの	TRUE	拒否される
string_equal	versionid付き、指定のもの以外	FALSE	拒否されない
string_equal_if_exist	versionid付き、指定のもの以外	FALSE	拒否されない

関連説明

特殊文字はurlencodeでの処理が必要

リクエストパラメータ内の特殊文字はurlencodeでの処理が必要です。このため、バケットポリシーの中で、リクエストパラメータによる条件キーを使用する場合は、先にurlencode処理をしておく必要があります。例えば、 `cos:response-content-type` 条件キーを使用する際、条件値が"image/jpeg"であれば、必ずurlencodeで"image%2Fjpeg"に変換してからバケットポリシーに入力する必要があります。

最小権限の原則に従い、*の使用を避ける

条件キーを使用する際は最小権限の原則に従い、権限の設定が必要なactionのみを追加し、ワイルドカード「*」の使用は避けてください。ワイルドカード「*」を乱用すると、一部のリクエストが失敗する場合があります。例えば次の例のように、GetObject以外の他のリクエストはいずれもリクエストパラメータresponse-content-typeの使用をサポートしていません。

deny + string_equal_if_exist条件オペレーターはリクエスト内にこの条件キーがない場合、デフォルトでtrueとして処理します。このため、PutObject、PutBucketなどのリクエストを発行した際に、このdeny statementにヒットし、リクエストが拒否されます。

allow + string_equalはリクエストにこの条件キーがない場合、デフォルトでfalseとして処理します。このため、PutObject、PutBucketなどのリクエストを発行した際に、このallow statementにヒットすることができず、リクエストが許可されません。

```
{
    "version":"2.0",
    "statement":[
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"allow",
            "action":[
                "*"
            ],
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ],
            "condition":{
                "string_equal":{
                    "cos:response-content-type":"image%2Fjpeg"
                }
            }
        },
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"deny",
            "action":[
                "*"
            ],
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ],
            "condition":{
                "string_not_equal_if_exist":{
                    "cos:response-content-type":"image%2Fjpeg"
                }
            }
        }
    ]
}
```

別の方法として、allow+string_equal_if_existおよびdeny + string_not_equalを使用すると、response-content-typeリクエストパラメータを含まないリクエストが許可されます。

deny + string_equal条件オペレーターはリクエスト内にこの条件キーがない場合、デフォルトでfalseとして処理します。このため、PutObject、PutBucketなどのリクエストを発行した際に、このdeny statementにヒットせず、リクエストは拒否されません。

allow + string_equal_if_existはリクエスト内にこの条件キーがない場合、デフォルトでtrueとして処理します。このため、PutObject、PutBucketなどのリクエストを発行した際に、allow statementにヒットすることができ、リクエストは権限を取得します。

ただし、このように条件オペレーターを使用すると、GetObjectにresponse-content-typeを含めるかどうかの制限が行えなくなります。GetObjectにresponse-content-typeリクエストパラメータが含まれない場合は、他のリクエストと同様にデフォルトで許可されます。GetObjectにresponse-content-typeリクエストパラメータが含まれる場

合のみ、ご自身で定めた条件に従って、リクエストパラメータの内容が意図するものと一致しているかを確認することができ、それによって条件付きの権限付与を実現できます。

```
{
    "version":"2.0",
    "statement":[
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
                },
                "effect":"allow",
                "action":[
                    "*"
                ],
                "resource":[
                    "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
                ],
                "condition":{
                    "string_equal_if_exist":{
                        "cos:response-content-type":"image%2Fjpeg"
                    }
                }
            },
            {
                "principal":{
                    "qcs":[
                        "qcs::cam::uin/1250000000:uin/1250000001"
                    ]
                },
                "effect":"deny",
                "action":[
                    "*"
                ],
                "resource":[
                    "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
                ],
                "condition":{
                    "string_not_equal":{
                        "cos:response-content-type":"image%2Fjpeg"
                    }
                }
            }
        ]
    }
```

このため、より安全な方法は、最小権限の原則に従い、ワイルドカード「*」を使用せず、actionをGetObjectに限定することとなります。

次の例では、ポリシーの発効条件は、「GetObjectリクエストに必ずresponse-content-typeが含まれ、かつリクエストパラメータの値が必ずimage%2Fjpeg"である場合にのみ権限を得られる」と厳格に限定されています。

その他のリクエストは次の例におけるポリシーの影響を受けないため、最小権限の原則に従って、追加で単独の権限を付与することができます。

```
{
    "version":"2.0",
    "statement":[
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"allow",
            "action":[
                "name/cos:GetObject"
            ],
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ],
            "condition":{
                "string_equal":{
                    "cos:response-content-type":"image%2Fjpeg"
                }
            }
        },
        {
            "principal":{
                "qcs":[
                    "qcs::cam::uin/1250000000:uin/1250000001"
                ]
            },
            "effect":"deny",
            "action":[
                "name/cos:GetObject"
            ],
            "resource":[
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"
            ],
            "condition":{
                "string_not_equal_if_exist":{
                    "cos:response-content-type":"image%2Fjpeg"
                }
            }
        }
    ]
}
```

リクエスト方法の紹介
パーマネントキーを使用したCOSアクセス

最終更新日：：2024-06-26 11:09:29

背景の説明

RESTful APIによって、Cloud Object Storage（COS）に対しHTTP匿名リクエストまたはHTTP署名リクエストを送信することができます。匿名リクエストは一般的に、静的ウェブサイトのホスティングなどの、パブリックアクセスを必要とするケースに用いられます。そのほかの大多数のケースでは署名リクエストが必要となります。

署名リクエストは匿名リクエストより、署名値を1つ多く持っています。署名はキー（SecretId/SecretKey）およびリクエスト情報を暗号化して生成した文字列をベースとしたものです。SDKは署名を自動計算しますので、お客様はユーザー情報の初期化の際にキーを設定しさえすれば、署名の計算について気にする必要はありません。RESTful APIを通じて送信するリクエストには、署名アルゴリズムに基づいて計算した署名を追加する必要があります。

パーマネントキーの取得

パーマネントキーはCAMコンソールのAPIキー管理ページで取得できます。パーマネントキーにはSecretIdとSecretKeyが含まれ、アカウントの永続的なIDを表します。有効期限はありません。

SecretId：APIを呼び出した人のID識別に用いられます。

SecretKey：署名文字列の暗号化およびサーバーによる署名文字列の検証に用いられるキーです。

パーマネントキーを使用したCOSアクセス

APIリクエストによるCOSアクセス

APIリクエストを使用する場合、プライベートバケットの場合は必ず署名リクエストを使用しなければなりません。パーマネントキーで署名を生成し、Authorizationヘッダーに追加することで、署名リクエストを作成できます。リクエストをCOSに送信すると、COSは署名とリクエストが一致しているかを検証します。

説明：

署名生成のアルゴリズムは複雑なため、SDKを直接使用してリクエストを送信することで、このプロセスを省略することをお勧めします。

1. パーマネントキーによる署名生成

署名アルゴリズムの説明については、リクエスト署名のドキュメントをご参照ください。COSは署名生成ツール

もご提供しており、SDKで署名を生成することもできます。SDKの署名実装をご参照ください。ご自身でプログラムを作成して署名を生成することも可能ですが、署名アルゴリズムは複雑なため、通常その方法は推奨されません。

2. Authorizationヘッダーの入力

APIリクエストを送信する際、署名を標準Http Authorizationヘッダーに入力します。次はGetObjectリクエストの例です。

```
GET /<ObjectKey> HTTP/1.1
Host: <BucketName-APPID>.cos.<Region>.myqcloud.com
Date: GMT Date
```

```
Authorization: q-sign-algorithm=sha1&q-ak=SecretId&q-sign-time=KeyTime&q-key-time=K
```

SDKツールによるCOSアクセス

1. パーマネントキーによるID情報の初期化

SDKツールのインストールが完了した後、最初にユーザーのID情報の初期化が必要です。ルートアカウントまたはサブアカウントのパーマネントキー（SecretIdおよびSecretKey）を入力します。

2. SDKを直接使用するCOSリクエスト

初期化すると、SDKツールを使用してアップロード・ダウンロードなどの基本操作を直接行うことができるようになります。SDKツールがお客様の代わりにキーによって署名を生成し、COSにリクエストを送信するため、APIリクエストのようにご自身で署名を生成する必要はありません。

例えば、Java SDKのコードは次のようになります。その他の言語のdemoについては、SDKの概要のクイックスタートドキュメントをご参照ください。

```
// 1 ユーザーID情報（secretId、secretKey）を初期化します。
// SECRETIDおよびSECRETKEYについてはCAMコンソールhttps://console.tencentcloud.com/cam/c
String secretId = "SECRETID";
String secretKey = "SECRETKEY";
COSCredentials cred = new BasicCOSCredentials(secretId, secretKey);
// 2 bucketのリージョンを設定します。COSリージョンの略称についてはhttps://cloud.tencent.com/
// clientConfigにはregion、https(デフォルトではhttp)、タイムアウト、プロキシなどを設定するset
Region region = new Region("COS_REGION");
ClientConfig clientConfig = new ClientConfig(region);
// ここではhttpsプロトコルの設定と使用を推奨します
// バージョン5.6.54からは、デフォルトでhttpsを使用します
```

```
clientConfig.setHttpProtocol(HttpProtocol.https);
// 3 cosクライアントを生成します。
COSClient cosClient = new COSClient(cred, clientConfig);
```

一時キーを使用したCOSアクセス

最終更新日：：2024-06-26 11:09:29

一時キーの説明

一時キーは、Security Token Service（STS）が提供する一時アクセスのための証明書です。一時キーはTmpSecretId、TmpSecretKey、Tokenの3つの部分から成ります。パーマネントキーと異なり、一時キーには次のような特徴があります。

- 有効期間が短く（30分～36時間）、パーマネントキーを開示する必要がないため、アカウント漏洩のリスクが低くなります。
- 一時キーを取得する際、policyパラメータを渡して一時的な権限を設定することで、使用者の権限の範囲をさらに限定することができます。

このため、一時キーはフロントエンドの直接転送などの、一時的な権限承認のシーンに適します。信頼性の低いユーザーに対しては、パーマネントキーではなく一時キーを発行することで、安全性を高めることができます。

一時キーの取得

一時キーは、当社のご提供するCOS STS SDK方式で取得するか、またはSTSのクラウドAPI方式によって直接取得することができます。

詳細については、一時キーの生成および使用ガイドをご参照ください。

一時キーの権限

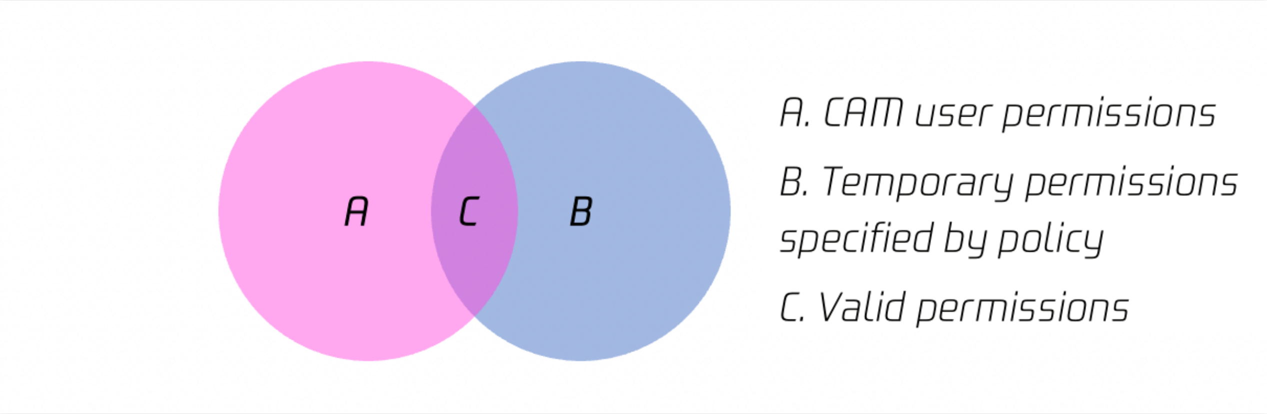
一時キーを申請する前に、Cloud Access Management（CAM）ユーザー（Tencent Cloudルートアカウントまたはサブアカウント）を取得しておく必要があります。Policyパラメータを設定することで、一時キーに一時ポリシーを追加して使用者の権限を制限することができます。

policyパラメータを設定しない場合、取得した一時キーはCAMユーザーと同等の権限を有します。

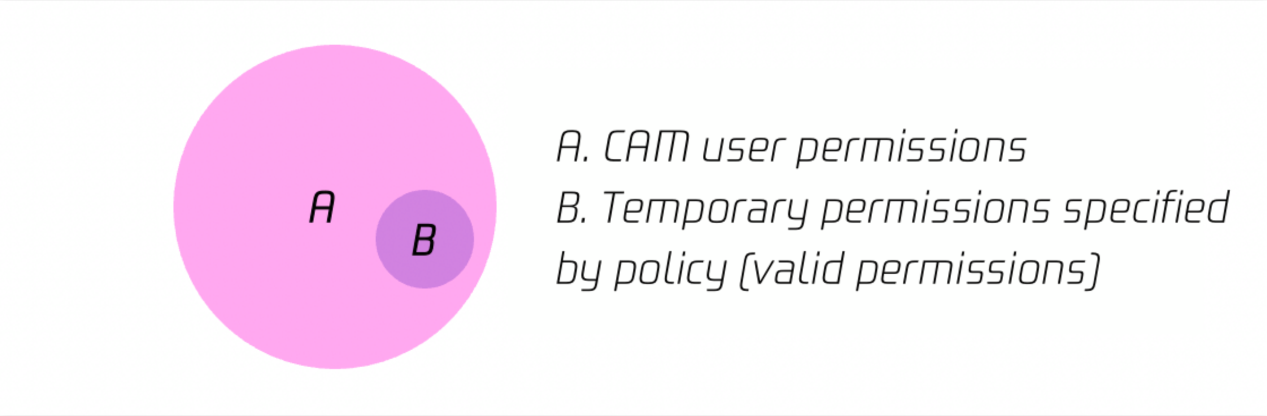
policyパラメータを設定した場合、取得した一時キーの権限は、CAMユーザーの権限をベースにして、さらにpolicyで設定した範囲内に制限されます。

例えば、「A」がCAMユーザーの従来の権限を表し、「B」がpolicyパラメータによって一時キーに設定した権限を表すとした場合、「A」と「B」の共通部分が一時キーの最終的に有効な権限となります。

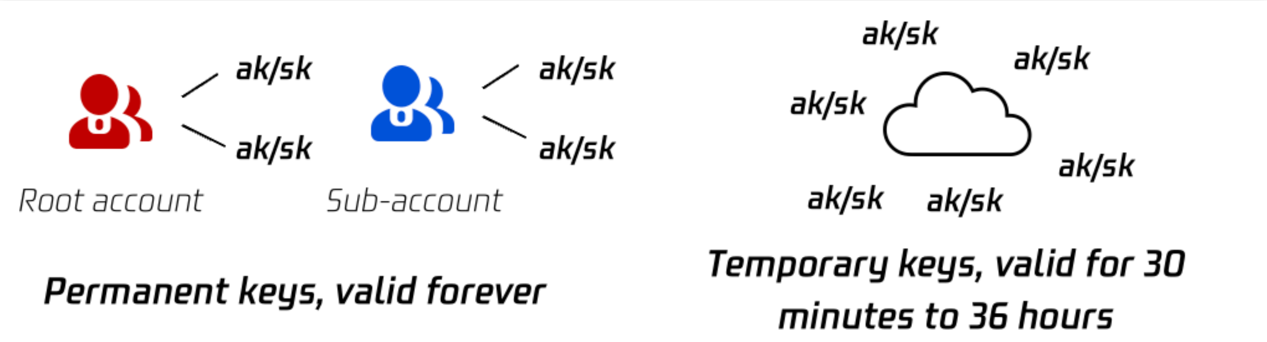
下図のように、CAMユーザー権限とpolicyの一時権限の共通部分が有効な権限です。


A
C
B
A. CAM user permissions
B. Temporary permissions specified by policy
C. Valid permissions

下図のように、policyがCAMユーザー権限の範囲内にある場合は、policyが有効な権限となります。


A
B
A. CAM user permissions
B. Temporary permissions specified by policy (valid permissions)

一時キーを使用したCOSアクセス


ak/sk
ak/sk
Root account
ak/sk
ak/sk
Sub-account
Permanent keys, valid forever
ak/sk
ak/sk
ak/sk
ak/sk
ak/sk
ak/sk
Temporary keys, valid for 30 minutes to 36 hours

一時キーにはSecretId、SecretKey、Tokenが含まれます。各ルートアカウントおよびサブアカウントはいずれも複

数の一時キーを生成することができます。パーマネントキーと異なり、一時キーの有効期間は30分から36時間しかありません。一時キーはフロントエンドの直接転送などの、一時的な権限承認のシーンに適します。信頼性の低いユーザーに対しては、パーマネントキーではなく一時キーを発行することで、安全性を高めることができます。詳細については、一時キーの生成および使用ガイドおよびフロントエンドの直接転送に用いる一時キーの使用ガイドをご参照ください。

APIリクエストの送信

パーマネントキーと同じように、一時キーによっても署名を生成することができます。リクエストヘッダーにAuthorizationを入力することで、署名リクエストを作成します。COSはリクエストを受信すると、署名が有効か、および一時キーが有効期間内かどうかをチェックします。

署名アルゴリズムの説明については、リクエスト署名をご参照ください。COSは署名生成ツールもご提供しており、SDKで署名を生成することもできます。SDKの署名実装をご参照ください。

SDKツールの使用

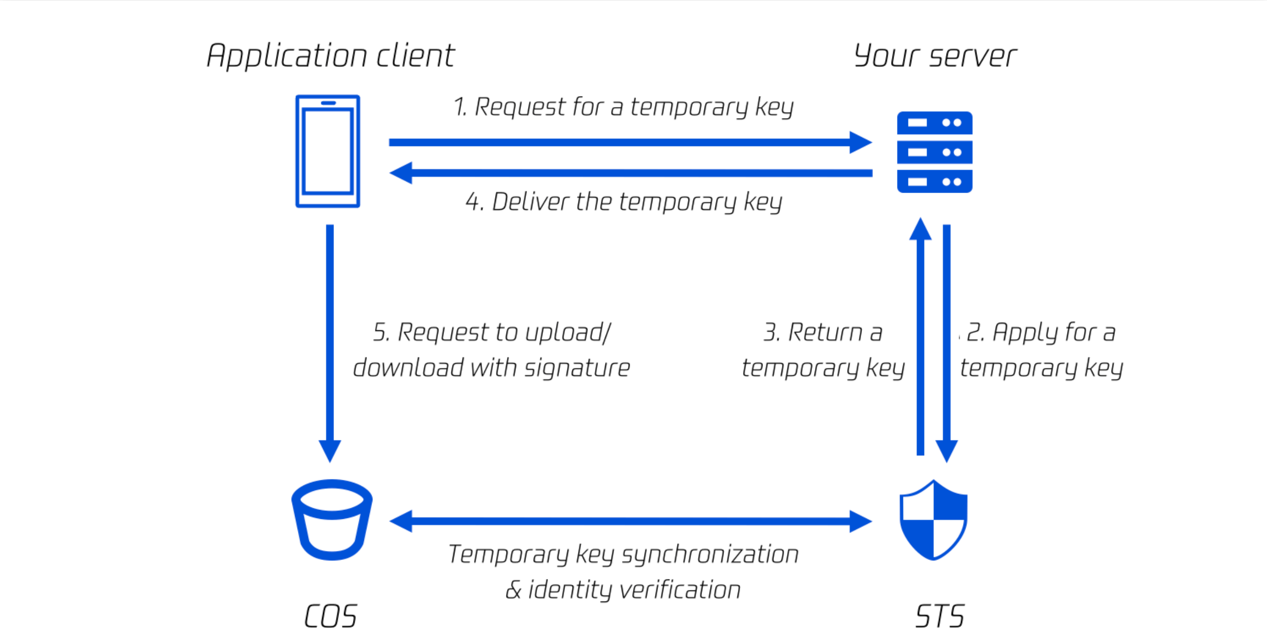
SDKツールをインストールすると、一時キーを使用してユーザーのID情報を初期化できるほか、一時キー（SecretId、SecretKey、Token）を使用してCOSClientを初期化し、署名を生成せずに、SDKを使用して直接アップロード、ダウンロードなどの操作を行うことができるようになります。一時キーの生成については一時キーの生成および使用ガイドをご参照ください。

Java SDKについては次の例をご参照ください。その他の言語のdemoについては、SDKの概要をご参照ください。

```
// 1 取得した一時キー (tmpSecretId、tmpSecretKey、sessionToken)を渡します
String tmpSecretId = "SECRETID";
String tmpSecretKey = "SECRETKEY";
String sessionToken = "TOKEN";
BasicSessionCredentials cred = new BasicSessionCredentials(tmpSecretId, tmpSecretKe
// 2 bucketのリージョンを設定します。COSリージョンの略称についてはhttps://cloud.tencent.com/
// clientConfigにはregion、https(デフォルトではhttp)、タイムアウト、プロキシなどを設定するset
Region region = new Region("COS_REGION");
ClientConfig clientConfig = new ClientConfig(region);
// 3 cosクライアントを生成します
COSClient cosClient = new COSClient(cred, clientConfig);
```

一時キーのユースケース

一時キーは主に第三者に対しCOSへの一時的なアクセス権限を承認するために用いられます。例えば、ユーザーがクライアントAppを開発し、データをCOSバケットに保存したとします。この場合、パーマネントキーを直接Appクライアント上に置くことはセキュリティ上問題がありますが、クライアントに対してはアップロード・ダウンロードの権限を承認する必要があります。このようなケースでは一時キーを使用することができます。


Application client
Your server
1. Request for a temporary key
4. Deliver the temporary key
5. Request to upload/ download with signature
3. Return a temporary key
2. Apply for a temporary key
Temporary key synchronization & identity verification
COS
STS

上の図のように、ユーザーがAppクライアントを開発し、ユーザーのサーバー上にパーマネントキーを保存し、一時キーを使用してフロントエンド直接転送を行うには次のいくつかの手順が必要です。

1. Appクライアントからユーザーサーバーに、データアップロード・ダウンロード用の一時キーをリクエストします。
2. ユーザーサーバーはパーマネントキーのIDを使用して、STSサーバーに一時キーを申請します。
3. STSサーバーはユーザーサーバーに一時キーを返します。
4. ユーザーサーバーは一時キーをAppクライアントに送信します。
5. Appクライアントは一時キーを使用して署名リクエストを生成し、COSに対しデータのアップロード・ダウンロードをリクエストします。

一時キーはフロントエンドデータの直接転送のユースケースに適しています。次のベストプラクティスをご参照の上、一時キーをご利用ください。

Web端末直接転送の実践

ミニプログラム直接転送の実践

モバイルアプリケーション直接転送の実践

署名付きURLを使用したCOSアクセス

最終更新日：：2024-06-26 11:09:29

Cloud Object Storage（COS）は署名付きURLを使用したオブジェクトのアップロード、ダウンロードをサポートしています。その原理は、URLに署名を埋め込んで署名付きリンクを生成するものです。署名の有効期限によって、署名付きURLの有効期間を管理することができます。

署名付きURLを使用してダウンロードを行い、一時URLを取得してファイル、フォルダの一時的な共有に用いることができます。あるいは長い署名有効期間を設定することで、長期間有効なURLを取得し、ファイルの長期的な共有に用いることもできます。詳細については、ファイルの共有をご参照ください。

また、署名付きURLを使用してアップロードを行うこともできます。詳細については、ファイルのアップロードをご参照ください。

ファイルの共有（ファイルのダウンロード）

COSはオブジェクトの共有をサポートしています。署名付きURLを使用することで、ファイルやフォルダを他のユーザーと期限付きで共有することができます。署名付きURLの原理は、署名をオブジェクトURLに埋め込んで結合するものです。その後の署名生成アルゴリズムについては、リクエスト署名をご参照ください。

バケットはデフォルトではプライベート読み取りであり、オブジェクトのURLから直接ダウンロードしようとするとアクセスエラーが表示されます。オブジェクトURLの後に有効な署名を結合すると、**署名付きURL**を取得できます。署名にはID情報が含まれるため、署名付きURLはオブジェクトのダウンロードに用いることができます。

説明：

やむを得ずパーマネントキーを使用して署名付きURLを生成する場合は、リスク回避のため、パーマネントキーの権限の範囲をアップロードまたはダウンロード操作のみに限定することをお勧めします。また、生成した署名の有効期間を、今回のアップロードまたはダウンロード操作に必要な最短の期限までに設定し、指定した署名付きURLの有効期限が過ぎるとリクエストが中断するようにします。失敗したリクエストは新しい署名を申請後に再度実行する必要があります。中断からの再開はサポートしていません。

```
// オブジェクトURL
https://test-12345678.cos.ap-beijing.myqcloud.com/test.png

// 署名付きURL（署名値を結合したオブジェクトURL）
https://test-12345678.cos.ap-beijing.myqcloud.com/test.png?q-sign-algorithm=sha1&q-
```

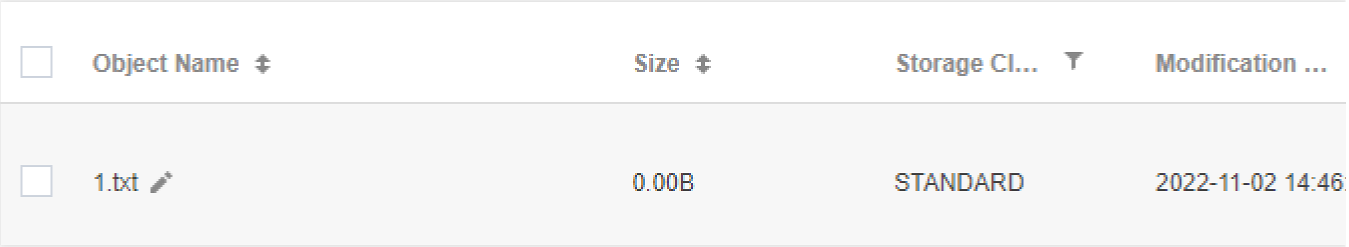
次にファイル共有の方法をいくつかご紹介します。これらの方法は本質的にはすべて署名を自動生成し、オブジェクトURLの後ろに結合することで、ダウンロードやプレビューに直接用いることができる一時リンクを生成するものです。

一時リンクのクイック取得（有効期間1～2時間）

コンソールまたはCOSBrowserツールによってオブジェクトの一時リンクをクイック取得することができます。

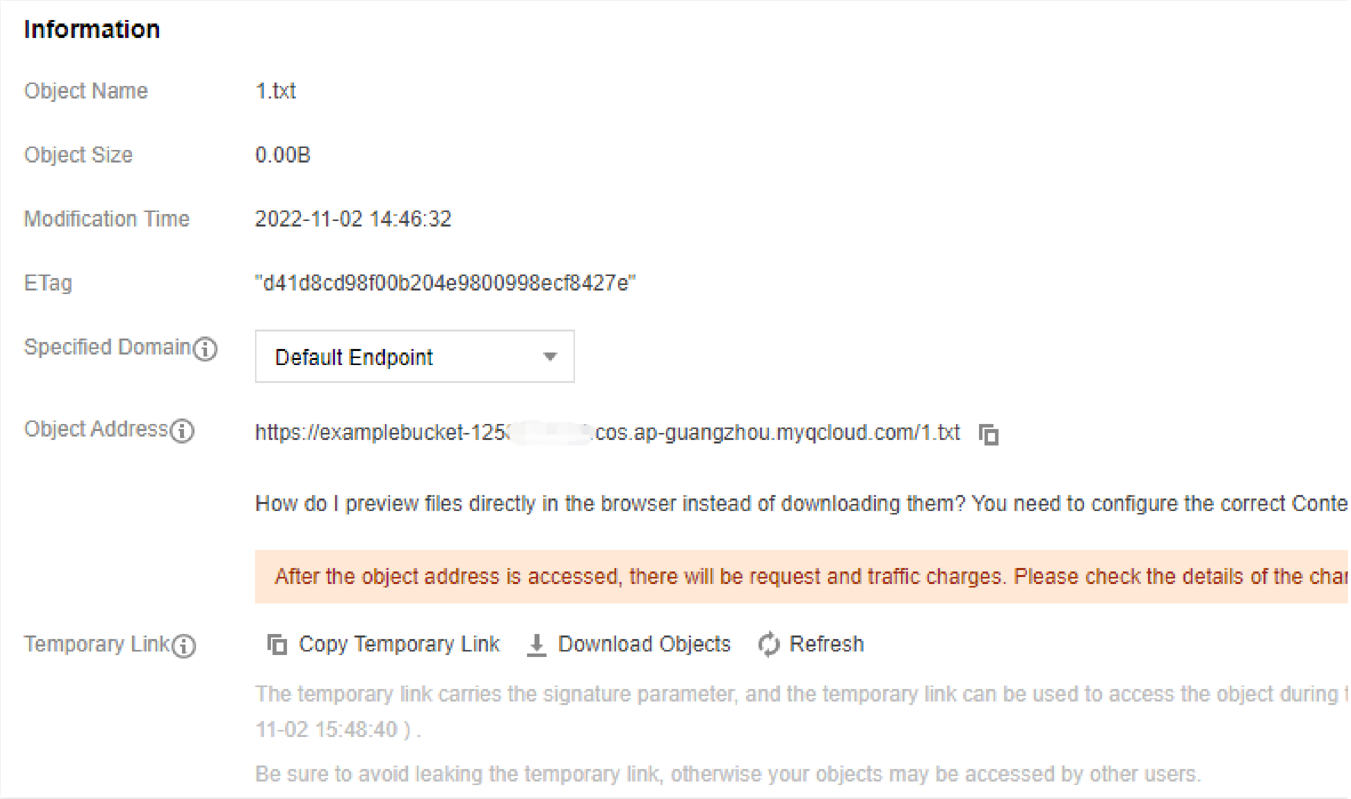
コンソール（Webページ）

1. COSコンソールにログインし、バケット名をクリックし、「ファイルリスト」に進み、オブジェクトの**詳細**をクリックします。



☐	Object Name	Size	Storage Cl...	Modification ...
☐	1.txt	0.00B	STANDARD	2022-11-02 14:46

2. オブジェクトの詳細ページに進み、一時リンクをコピーします。有効期間は1時間です。



Information

Object Name: 1.txt

Object Size: 0.00B

Modification Time: 2022-11-02 14:46:32

ETag: "d41d8cd98f00b204e9800998ecf8427e"

Specified Domain: Default Endpoint

Object Address: https://examplebucket-125[redacted].cos.ap-guangzhou.myqcloud.com/1.txt

How do I preview files directly in the browser instead of downloading them? You need to configure the correct Conte

After the object address is accessed, there will be request and traffic charges. Please check the details of the char

Temporary Link

The temporary link carries the signature parameter, and the temporary link can be used to access the object during t
11-02 15:48:40) .

Be sure to avoid leaking the temporary link, otherwise your objects may be accessed by other users.

COSBrowser（クライアント）

ドキュメントファイルリンクの発行を参照し、ルートアカウントキーを使用して最長2時間の一時リンクを取得することができます。サブアカウントキーを使用する場合は、最長1.5日間の一時リンクを取得することができます。

時間をカスタマイズした一時リンクの取得

署名ツールの使用

適するケース：プログラミングに不慣れなユーザー

操作手順は次のとおりです。

1. ファイルリンク：COSコンソールにログインし、オブジェクトの詳細から、署名のない「オブジェクトアドレス」を取得します。
2. APIキー管理からSecretIdとSecretKeyを取得します。
3. COS署名ツールをクリックし、署名リンクを取得します。

有効期間：秒、分、時間、日レベルの設定をサポートします。

SDKを使用した署名付きURLの一括取得

適するケース：一時リンクを一括取得したい場合、プログラミングの基礎を習得したユーザー

コンソールおよびCOSBrowserから取得する一時リンクは有効期間が短いため、より長時間の一時リンクが必要な場合は、SDKを使用して署名付きURLを生成し、署名の有効期間の管理を実現することもできます。生成メソッドについては署名付きURLによるダウンロード権限承認を参照し、使いやすい開発言語を選択してください。

署名付きURLの生成には一時キーまたはパーマネントキーを使用することができます。両者の違いは、一時キーの最長有効期間は36時間以内であり、パーマネントキーには有効期限がないという点です。このことは署名付きURLの有効期間に間接的な影響を与えます。

パーマネントキーを使用した署名付きURLの生成（任意の期間）

パーマネントキーには有効期限がないため、署名付きURLの有効期間は設定した署名の有効期間によって決まります。SDKの署名付きURL生成メソッドを直接呼び出すことができます。操作手順は次のとおりです。

1. secret_id、secret_key、regionなどを入力してclientを初期化します。
2. バケット名、オブジェクト名、署名の有効期間を入力し、時間をカスタマイズした署名付きURLを生成します。詳細については下記の各言語のSDKドキュメントをご参照ください。

Android SDK	C SDK	C++ SDK	.NET SDK
Go SDK	iOS SDK	Java SDK	JavaScript SDK
Node.js SDK	PHP SDK	Python SDK	ミニプログラムSDK

一時キーを使用した署名付きURLの生成（36時間以内）

フロントエンドデータの直接転送のケースでは、一時キーの使用が必要な場合が多くあります。一時キーの説明と生成ガイドについては次をご参照ください。

一時キーを使用したCOSアクセス

一時キーの生成および使用ガイド

COSへのフロントエンド直接転送に用いる一時キーのセキュリティガイド

一時キーは最長36時間までであり、署名付きURLの有効期間は設定した署名の有効期間と一時キーの有効期間の最小値から決定されます。設定した署名の有効期間をX、一時キーの有効期間をY、リンクの実際の有効期間をTと

します。

```
T=min(X,Y)。X<=36のため、T<=36です。
```

一時キーを使用して署名付きURLを生成するには、次の2つの手順が必要です。

1. 一時キーの取得を行います。
2. 一時キーを取得すると、パーマネントキーに類似した関数を使用して署名付きURLを生成できるようになります。一時キーを使用してclientを初期化する場合、SecretId、SecretKeyの入力のほかにtokenも入力し、パラメータ `x-cos-security-token` も含める必要があることに注意が必要です。詳細については、下記の各言語のSDKドキュメントをご参照ください。

Android SDK	C SDK	C++ SDK	.NET SDK
Go SDK	iOS SDK	Java SDK	JavaScript SDK
Node.js SDK	PHP SDK	Python SDK	ミニプログラムSDK

フォルダの共有

フォルダは一種の特殊なオブジェクトであり、コンソールまたはCOSBrowser ツールを使用してフォルダを共有することができます。詳細については、フォルダの共有をご参照ください。

ファイルのアップロード

第三者がオブジェクトをバケットにアップロードできるようにし、なおかつ相手にCAMアカウントまたは一時キーなどの方法を使用させたくない場合は、署名付きURLを使用して署名を第三者に渡し、一時的なアップロード操作を完了させることができます。有効な署名付きURLを受領した人は誰でもオブジェクトをアップロードできます。

説明：

やむを得ずパーマネントキーを使用して署名付きURLを生成する場合は、リスク回避のため、パーマネントキーの権限の範囲をアップロードまたはダウンロード操作のみに限定することをお勧めします。また、生成した署名の有効期間を、今回のアップロードまたはダウンロード操作に必要な最短の期限までに設定し、指定した署名付きURLの有効期限が過ぎるとリクエストが中断するようにします。失敗したリクエストは新しい署名を申請後に再度実行する必要があります。中断からの再開はサポートしていません。

手段1：SDKを使用した署名付きURLの生成

各言語のSDKがアップロード署名付きURLの生成メソッドを提供しています。生成メソッドについては、署名付きURLによるアップロード権限承認を参照し、使いやすい開発言語を選択してください。

手段2：ご自身での署名リンクの結合

署名付きURLは実際にはオブジェクトURLの後に署名を結合したものです。このため、SDK、署名生成ツールなどによってご自身で署名を生成し、URLと署名を結合して署名リンクとし、オブジェクトのアップロードに用いることもできます。ただし、署名生成のアルゴリズムは複雑なため、一般的な状況ではこの方法の使用は推奨されません。

匿名でのCOSアクセス

最終更新日：：2024-06-26 11:09:29

Cloud Object Storage（COS）バケットはデフォルトではプライベートであり、COSへのアクセスはID認証を経る必要があり、オブジェクトのURLによるCOSアクセスには署名が必要です。ただし、リソース（バケット、オブジェクト、フォルダ）がパブリック読み取りに開放された場合は匿名アクセスが許可され、オブジェクトのURLを通じてリソースを直接ダウンロードすることが可能になります。

COSは権限開放の範囲に基づき、バケットレベル、オブジェクトレベル、フォルダレベルでのパブリック読み取り設定をサポートしています。

バケットをパブリック読み取りとして開放する

バケットをパブリック読み取り・プライベート書き込みとして開放すると、バケット内のすべてのオブジェクトに匿名でのアクセスが可能になります。設定方法については、バケットアクセス権限の設定をご参照ください。

オブジェクトをパブリック読み取りとして開放する

指定のオブジェクトをパブリック読み取り・プライベート書き込みとして開放すると、そのオブジェクトにオブジェクトのURLから直接アクセスすることが可能になります。設定方法については、オブジェクトのアクセス権限の設定をご参照ください。

フォルダをパブリック読み取りとして開放する

フォルダをパブリック読み取り・プライベート書き込みとして開放すると、そのフォルダ内のすべてのファイルに匿名でのアクセスが可能になります。設定方法については、フォルダの権限の設定をご参照ください。

パブリック読み取り権限の評価の仕組み

COS権限の評価の仕組みについては、アクセスポリシーの評価フローをご参照ください。バケット、フォルダ、オブジェクトレベルでのパブリック読み取り権限設定に競合が発生した場合、優先順位は次のようになります。あるオブジェクトのACLに対しては、オブジェクトACLの優先度が最も高くなります。オブジェクトACLが継承権限の場合は、フォルダACLに準じます。フォルダACLが継承権限の場合は、バケットACLに準じます。

CDNアクセラレーションを使用したアクセス

CDNアクセラレーションの概要

最終更新日：：2024-06-26 11:09:29

Content Delivery Network（CDN）を使用してCloud Object Storage（COS）のアクセラレーションを行うことで、バケット内のコンテンツを広範囲にダウンロード、配信することができます。特に、同一のコンテンツを繰り返しダウンロードするユースケースに適しています。back-to-origin認証機能を使用すると、CDNを使用したプライベート読み取りバケット内のコンテンツのアクセラレーションを実現できます。CDN認証機能を使用すると、コンテンツを正当なユーザーにのみダウンロード可能とすることができ、ダウンロードを開放することに伴うデータセキュリティおよびトラフィックコストなどの問題を防ぐことができます。

説明：

CDNアクセラレーションドメイン名を有効にした場合、CDNアクセラレーションドメイン名を使用してデータのダウンロードやアクセスを行うと、CDN back-to-originラフィックとCDNトラフィックが発生します。詳しくは、COSをCDNオリジンサーバーとした場合発生するトラフィックをご参照ください。

Content Delivery Network

CDNの定義

CDNは既存のInternet上に追加された新しいネットワークアーキテクチャのレイヤーであり、世界各国に分布する高性能なアクセラレーションノードで構成されます。これらの高性能なサービスノードは一定のキャッシュポリシーに従ってお客様の業務コンテンツを保存しており、お客様のユーザーがある業務コンテンツをリクエストすると、リクエストがユーザーから最も近いサービスノードにスケジューリングされ、サービスノードから直接迅速にレスポンスすることで、ユーザーのアクセス遅延を効果的に低減し、可用性を向上させます。

CDNはキャッシュおよびback-to-origin行為を行います。すなわち、ユーザーがあるURLにアクセスしたとき、エッジノードに解決されてもレスポンスを必要とするキャッシュコンテンツにヒットしなかった場合、またはキャッシュが期限切れの場合は、オリジンサーバーに戻ってレスポンスを必要とするコンテンツを取得するものです。

適用ケース

レスポンス遅延およびダウンロード速度に対する要件が比較的厳しいケース。

地域、国、大陸を越えて数GBから数TBのデータを転送する必要があるケース。

同一のコンテンツを集中的に繰り返しダウンロードする必要があるケース。

セキュリティのタイプ

back-to-origin認証：ユーザーのリクエストしたデータがエッジノードでキャッシュにヒットしなかった場合、CDNはback-to-originを行ってデータ内容を取得する必要があります。COSをオリジンサーバーとして使用し、back-to-origin認証を有効にすると、CDNエッジノードは特殊なサービスIDを使用してCOSオリジンサーバーにアクセスし、プライベートアクセスバケット内のデータの取得とキャッシュを実現することができます。
CDNサービス権限承認：CDNエッジノードはCDNサービス権限承認を追加することにより、特殊なサービスIDを使用してCOSオリジンサーバーにアクセスすることができるようになります。CDNサービス権限承認を追加してからでなければ、back-to-origin認証は有効化できません。
CDN認証設定：ユーザーがエッジノードにアクセスしてキャッシュデータを取得する際、エッジノードは認証設定ルールに基づいて、URLアクセスにおけるID認証フィールドを検証することで、権限のないアクセスを防止し、リンク不正アクセス防止を実現し、エッジノードのキャッシュデータの安全性と信頼性を高めます。

COSのアクセスノード

アクセスノードの定義

アクセスノードとはユーザーがバケットを作成する際、バケットのリージョンおよび名称に基づいて、システムが自動的にバケットに割り当てるアクセスドメイン名であり、このドメイン名によってバケット内のデータにアクセスすることができます。
静的ウェブサイト機能を有効にすると、静的ウェブサイトのアクセスノードを1つ追加で取得でき、それはデフォルトのアクセスノードとは性質の異なる、特殊な設定のレスポンス内容を表示するために用いられます。

アクセスノード

アクセスノード：ユーザーがバケットを作成すると、COSはバケットに対し自動的に1つのXMLアクセスノードを割り当てます。 このアクセスノードの形式は<BucketName-APPID>.cos.<Region>.myqcloud.comであり、RESTful APIによるアクセスに適用されます。ユーザーはアクセスドメイン名によって、またAPIドキュメントの実行ルールを確認してバケットの設定を行ったり、オブジェクトのアップロード、ダウンロード操作を行ったりすることができます。
静的ウェブサイトノード：コンソール上のバケット基本設定画面で静的ウェブサイトのホスティング機能を有効にすることができ、有効にすると1つのアクセスノードが提供されます。アクセスノードの形式は<BucketName-APPID>.cos-website.<Region>.myqcloud.comです。静的ウェブサイトは特殊なインデックスページ（IndexPage）、エラーページ（ErrorPage）およびリダイレクトなどをサポートしており、サポートはオブジェクトのダウンロード系の操作のみとなります。ユーザーは静的ウェブサイトのドメイン名によってコンテンツを取得できます。

アクセス権限

パブリック読み取り：バケットをパブリック読み取りに設定すると、バケットのアクセスドメイン名によって誰でもそのバケットにアクセスすることができます。ユーザーがパブリック読み取りバケットをオリジンサーバーと

してback-to-originを行う場合は、CDNアクセラレーションを直接有効化でき、CDN認証およびback-to-origin認証を使用する必要はありません。
プライベート読み取り：バケットをプライベート読み取りに設定すると、ユーザーはアクセスポリシーを作成することによってアクセス者を管理でき、これにはCDNサービス権限承認の管理なども含まれます。ユーザーがプライベート読み取りバケットをオリジンサーバーとしてback-to-originを行う場合、back-to-origin認証を有効にしていてもCDN認証を有効にしていなければ、権限範囲外の人がCDNによって直接バケットにアクセスすることが可能になってしまいます。このため、プライベート読み取りバケットについては、**CDN認証とback-to-origin認証を同時に有効にすることでデータの安全性を保障するよう強く推奨します。**

CDNアクセラレーションを使用したCOSアクセス

ユーザーはカスタムCDNアクセラレーションドメイン名によって、COSへのアクセラレーションアクセスを行うことができます。ユーザーは初めにICP登録済みのカスタムドメイン名をご自身で準備し、オリジンサーバーをCOSバケットに指定することで、カスタムCDNアクセラレーションドメイン名を使用したバケット内のオブジェクトに対するアクセラレーションアクセスを実現できます。
説明：
Tencent Cloud CDNのアクセラレーションドメイン名はデフォルトではIPアドレスを提供しません。ドメイン名の解決状況を確認したい場合は、digコマンドを使用して照会することができます。

パブリック読み取りバケット

バケットをパブリックアクセス許可に設定すると、CDN back-to-originをCOSアクセスノードに設定した場合、back-to-origin認証を有効にしなくても、CDNエッジノードがバケット内のオブジェクトデータを取得してキャッシュすることが可能です。
CDNコンソールで認証設定を有効にして、バケット内のデータを**限定的に**保護することも引き続き可能です。CDNのこの機能を有効にしているかどうかにかかわらず、バケットのアクセスドメイン名を知っているユーザーはバケット内のすべてのオブジェクトにアクセス可能なためです。CDN認証設定の違いによる、ドメイン名のパブリック読み取りバケットに対するアクセス機能の違いについては次の表をご参照ください。

CDN認証設定	CDNアクセラレーションドメイン名アクセス	COSドメイン名アクセス	一般的なケース
無効（デフォルト）	アクセス可	アクセス可	全サーバーでパブリックアクセス許可、CDNとオリジンサーバーどちらからのアクセスも可
有効	URLを使用した認証が必要	アクセス可	CDNアクセスに対してはリンク不正アクセス防止が有効、ただしオリジンサーバーアクセスは保護されないため非推奨

プライベート読み取りバケット

バケットがデフォルトのプライベート読み取りである場合、CDN back-to-originをCOSアクセスノードに設定すると、CDNエッジノードは**データの取得とキャッシュが一切できなくなります**。このため、CDNサービスIDをバケットアクセスポリシー（Bucket Policy）に追加するとともに、そのIDによる次の操作の実行を許可する必要があります。

GET Object：オブジェクトのダウンロード
HEAD Object：オブジェクトメタデータの照会
OPTIONS Object：クロスドメイン設定のプリフライトリクエスト

CDNコンソールとCOSコンソールはいずれもワンクリックでの権限承認機能をご提供しています。**CDNサービス権限承認の追加**をクリックすれば完了です。この操作の完了後、**back-to-origin認証**オプションを有効にする必要があります。これでCDNエッジがこのサービスIDを使用してCOS内のデータにアクセスできるようになります。

注意：

バケットがプライベート読み取りに設定されている場合は、必ず権限承認を追加し、back-to-origin認証を有効にしてください。これを行わなければCOSはアクセスを拒否します。

CDNエッジは各ルートアカウントにつき、1つのサービスアカウントを生成します。このため、アカウントの権限承認はアクセラレーションドメイン名が所属するルートアカウントに対してのみ有効であり、異なるアカウント間でバインドしたアクセラレーションドメイン名はアクセスが拒否されます。

CDNサービス権限承認を追加し、back-to-origin認証を有効にすると、CDNエッジノードはデータを直接取得およびキャッシュできるようになります。このため、プライベートデータの保護が必要な場合は、認証設定を有効にして、バケット内のデータを保護することを強く推奨します。CDN認証設定の違いによる、ドメイン名のプライベート読み取りバケットに対するアクセス機能の違いについては次の表をご覧ください。

CDN認証設定	CDNアクセラレーションドメイン名アクセス	COSドメイン名アクセス	一般的なケース
無効（デフォルト）	アクセス可	COSを使用した認証が必要	CDNドメイン名に直接アクセス可能、オリジンサーバーデータ保護
有効	URLを使用した認証が必要	COSを使用した認証が必要	全リンクのアクセスを保護、CDN認証リンク不正アクセス防止をサポート

CDNアクセラレーションの設定

最終更新日：：2024-06-26 11:09:29

ユースケース

レスポンス遅延およびダウンロード速度に対する要件が比較的厳しいケース。
地域、国、大陸を越えて数GBから数TBのデータを転送する必要があるケース。
同一のコンテンツを集中的に繰り返しダウンロードする必要があるケース。

注意：

ダウンロードリクエストがTencent Cloud VPC内（例えばTencent Cloud CVMを使用したバケットへのアクセスなど）の場合は、COSの標準ドメイン名をそのまま使用することをお勧めします。カスタムCDNアクセラレーションドメイン名を使用すると、ユーザーはパブリックネットワークを経由してCDNノードにアクセスしたことになり、CDN back-to-originトラフィック料金、CDNトラフィック料金などが追加で発生します。

関連説明

ドメイン名の定義、CDN back-to-origin認証およびCDN認証設定に関しては、CDNアクセラレーションの概要のドキュメントでご紹介しているため、ここには記載しません。
CDN back-to-origin認証、CDN認証設定はカスタムCDNアクセラレーションドメイン名およびCOSドメイン名のオリジンサーバーバケットへのアクセス方式に影響する場合があります。具体的には下表のとおりです。

バケットへのアクセス権限	CDN back-to-origin認証を有効にしているか	CDN認証設定を有効にしているか	カスタムCDNアクセラレーションドメイン名によって オリジンサーバーにアクセス可能か	COSオリジンサーバードメイン名によって オリジンサーバーにアクセス可能か	適用ケース
パブリック読み取り	無効	無効	アクセス可	アクセス可	全サーバーパブリックアクセス
パブリック読み取り	有効	無効	アクセス可	アクセス可	非推奨
パブリック読み取り	無効	有効	URLを使用した認証が必要	アクセス可	非推奨

パブリック読み取り	有効	有効	URLを使用した認証が必要	アクセス可	非推奨
プライベート読み取り+CDNサービス権限承認	有効	有効	URLを使用した認証が必要	COSを使用した認証が必要	全リンク保護
プライベート読み取り+CDNサービス権限承認	無効	有効	URLを使用した認証が必要	COSを使用した認証が必要	非推奨
プライベート読み取り+CDNサービス権限承認	有効	無効	アクセス可	COSを使用した認証が必要	オリジンサーバー保護
プライベート読み取り+CDNサービス権限承認	無効	無効	アクセス不可	COSを使用した認証が必要	非推奨
プライベート読み取り	無効	有効または無効	アクセス不可	COSを使用した認証が必要	CDN利用不可

注意：

上記の表中の**オリジンサーバー保護**のケースでは、CDN認証設定を有効にしていない場合、CDNエッジノードにキャッシュしたデータが悪意をもってプルされるおそれがあるため、同時にCDN認証設定を有効にすることでデータの安全性を保障するよう、強く推奨します。

CDNアクセラレーションの設定

ユーザーはCOSコンソール上でバケットにカスタムドメイン名をバインドした後、カスタムドメイン名でCDNアクセラレーションを有効にし、最後にカスタムドメイン名によってバケットへのアクセスをアクセラレーションすることができます。カスタムドメイン名をバインドする際、ユーザーはご自身でカスタムドメイン名のサービスプロバイダにCNAME解決の追加を依頼する必要があります。

注意：

現在COSでカスタムアクセラレーションドメイン名を使用している場合はCDNを有効にする必要があります。ご自身の状況に応じて判断してください。

1. ドメイン名で国内のCDNにアクセスするには、ICP登録が必要です。ただしICP登録はTencent Cloudを通じて行わなくてもよく、アクセスするドメイン名が確実にICP登録されていればアクセスできます。
2. ドメイン名で海外のCDNにアクセスする場合は、ICP登録の必要はありません。ただし、Tencent Cloud上で保存するデータおよび操作行為については、関係国の法律法令、ならびに『Tencent Cloudサービス契約』を遵守する必要があることに必ずご注意ください。

前提条件

1. ドメイン名を登録していること。Tencent Cloudのドメイン名登録またはその他のサービスプロバイダを通じてドメイン名を登録できます。
2. ドメイン名のICP登録を行っていること。ドメイン名で国内のCDNにアクセスするには、ドメイン名のICP登録を完了していることが必要です。

操作手順

注意：

カスタムドメイン名の追加およびCDNアクセラレーションの有効化は、COSコンソールとCDNコンソールのどちらでも行うことができます。CDNコンソールからカスタムドメイン名を追加したい場合は、 ドメイン名へのアクセスをご参照ください。

1. バケットの選択

COSコンソールにログインし、左側ナビゲーションバーの**バケットリスト**をクリックし、CDNアクセラレーションを有効化したいバケットをクリックして、バケットに進みます。

2. カスタムCDNアクセラレーションドメイン名の追加

ドメイン名と伝送管理>カスタムCDNアクセラレーションドメイン名をクリックし、**カスタムCDNアクセラレーションドメイン名**の設定項目で**ドメイン名の追加**をクリックし、設定可能な状態にします。

ドメイン名：バインド対象のカスタムドメイン名（例えば `www.example.com` ）を入力します。入力するドメイン名はICP登録済みであること、およびDNSサービスプロバイダが対応するCNAMEを設定していることが必要です。詳細については、CNAME設定をご参照ください。アクセスするカスタムCDNアクセラレーションドメイン名が次に該当する場合は、ドメイン名所有権の検証を行う必要があります。詳細については、ドメイン名所有権の検証のドキュメントをご確認ください。

このドメイン名を初めて接続する

このドメイン名が他のユーザーによって接続されている

接続するドメイン名が汎用ドメイン名である

アクセラレーションリージョン：中国本土、中国本土以外およびグローバルアクセラレーションをサポートしています。グローバルアクセラレーションとは、すべてのリージョン間でのバケットアクセラレーションをサポート

することを指します。

オリジンサーバーのタイプ：デフォルトでは**デフォルトオリジンサーバー**です。オリジンサーバーにしているバケットで静的ウェブサイトを有効にし、かつ静的ウェブサイトのアクセラレーションを行いたい場合は、オリジンサーバーのタイプを**静的ウェブサイトのオリジンサーバー**に設定することができます。具体的には、CDNアクセラレーションの概要をご参照ください。

認証：back-to-origin認証をオンにします。プライベート読み取りバケットに対しては、**back-to-origin認証を有効化**してオリジンサーバーを保護してください。

注意：

プライベート読み取りバケットに対して、back-to-origin認証とCDNサービス権限承認を同時に有効化した場合、CDNがオリジンサーバーへアクセスする際に署名を必要としないため、CDNキャッシュリソースがパブリックネットワーク配信を行い、データセキュリティに影響します。CDN認証を有効化することをお勧めします。

back-to-origin認証の有効化

back-to-origin認証は、CDNエッジノードのサービスIDを検証し、不正アクセスを阻止するために用いられます。具体的には次のとおりです。

パブリック読み取りバケット：CDNエッジノードに何の権限承認を行わなくても、バケットに直接アクセスできるため、back-to-origin認証を有効にする必要はありません。

プライベート読み取りバケット：CDNエッジノードはback-to-origin認証によるサービスIDの検証を受ける必要があり、検証に合格したCDNエッジノードのみがバケット内のオブジェクトにアクセスできます。**back-to-origin認証**の有効化を選択します。

back-to-origin認証を有効化した後、右側の**保存**をクリックし、約5分待つと、カスタムドメイン名の追加とCDNアクセラレーションのデプロイが完了します。

CDN認証の有効化

注意：

カスタムドメイン名でCDNアクセラレーションを有効化すると、このドメイン名によって誰でもオリジンサーバーに直接アクセスできるようになります。データに一定のプライバシーが存在する場合は、必ずCDN認証設定を有効にして、オリジンサーバーデータを保護してください。

カスタムドメイン名のデプロイが完了すると、CDN認証欄にCDN認証機能設定リンクが表示されます。**設定**をクリックすると、CDNコンソールに直接進んでCDN認証設定を行うことができます。具体的な操作方法については、認証設定をご参照ください。

CDNキャッシュの自動更新：有効化すると、COSバケットのファイル更新ルールをトリガーした際にCDNキャッシュを自動更新します。COSの関数の計算で設定できます。操作ガイドはCDNキャッシュ更新の設定をご参照ください。

HTTPS証明書：カスタムCDNアクセラレーションドメイン名に HTTPS証明書を追加する必要がある場合は、CDNコンソールで設定できます。

3. ドメイン名の解決

カスタムドメイン名でCDNにアクセスすると、システムから自動的にCNAMEドメイン名（ `.cdn.dnsv1.com` を拡張子とするもの）が割り当てられます。お客様はドメイン名サービスプロバイダにCNAME設定を完了するよう依頼する必要があります。具体的にはCNAME設定をご参照ください。

注意：

CNAMEドメイン名には直接アクセスできません。

4. 機能の無効化

上記の手順が完了すると、 カスタムドメイン名によってバケット内のリソースにアクセスできるようになります。カスタムCDNアクセラレーションを無効化したい場合は、次の方法で行うことができます。

カスタムCDNアクセラレーションドメイン名管理画面で、**編集**をクリックするとドメイン名のステータスを変更できます。ドメイン名のステータスを**オンライン**から**オフライン**に変更し、**保存**をクリックしてデプロイされるまで待ちます。約5分で機能が無効化されます。CDNコンソール上では対応するカスタムアクセラレーションドメイン名のステータスが**有効化済み**から**無効化済み**に変わります。

注意：

カスタムドメイン名を削除したい場合、カスタムドメイン名のステータスが**オンライン**の場合は、ドメイン名を直接削除することはできません。ステータスを必ず**オフライン**にしてからドメイン名を削除してください。無効化または削除の操作はCDNコンソールで行うことができます。詳細については、ドメイン名の操作をご参照ください。

単一リンクの速度制限

最終更新日：：2024-06-26 11:09:29

単一リンクの速度制限

COSではファイルのアップロード、ダウンロードの際のトラフィックを管理し、他のアプリケーションのためのネットワーク帯域幅を確保することができます。PutObject、PostObject、GetObject、UploadPartリクエストの際にパラメータx-cos-traffic-limitを付加し、速度制限値を設定すると、COSは設定された速度制限値に基づいて、そのリクエストのネットワーク帯域幅を制御します。

利用説明

ユーザーはPUT Object、POST Object、GET Object、Upload Partリクエストの際にx-cos-traffic-limit リクエストヘッダー（POST Objectリクエストについてはリクエストのフォームフィールド）を付加して、そのリクエストの速度制限値を指定します。このパラメータはheader、リクエストパラメータ内、またはフォームアップロードインターフェースを使用する場合はフォームドメイン内に設定することができます。

x-cos-traffic-limitパラメータ値は数字でなければならず、単位はデフォルトではbit/sです。

速度制限値の範囲は819200～838860800、すなわち100KB/s～100MB/sです。この範囲を超過するとエラーコード400が返されます。

説明：

単位換算公式：1MByte=1024KByte=1048576Byte=8388608bitです。

APIの使用例

シンプルアップロードのAPIの例を次に示します。速度制限値は1048576 bit/s、すなわち128KB/sです。

```
PUT /exampleobject HTTP/1.1
Host: examplebucket-1250000000.cos.ap-beijing.myqcloud.com
Content-Length: 13
Authorization: q-sign-algorithm=sha1&q-ak=AKID8A0fBVtYFrNm02oY1g1JQQF0c3JO****&q-si
x-cos-traffic-limit: 1048576
```

バッチ処理

バッチ処理タスクの管理

最終更新日：：2024-06-26 11:09:29

コンソールでバッチ処理タスクの管理を行うことができます。ここではバッチ処理タスクに関する管理操作について詳細にご説明します。

タスクのフィルタリング

List JobsAPIによって、過去90日間に作成されたバッチ処理タスクをリストアップすることができます。バッチ処理タスクリストには、例えばタスクID、タスクの説明、タスクの優先度、タスクのステータス、タスクの実行状況などの、各バッチ処理タスクの情報が含まれます。タスクのステータスによって、バッチ処理タスクリストから同一のステータスにあるタスクをフィルタリングすることができます。コンソール上でフィルタリング操作を行う場合は、タスクの説明またはタスクIDによってもフィルタリングが可能です。

タスクのステータスの照会

例えば、タスクに関するより多くの情報を取得したい場合は、DescribeJobAPIによって単一のタスクのすべての情報を取得することができます。このインターフェースは指定されたタスクの操作設定、オブジェクトリストの情報、タスクレポートなどの情報を返します。このインターフェースによって指定のタスクの詳細を把握することができます。

タスクの優先度の割り当て

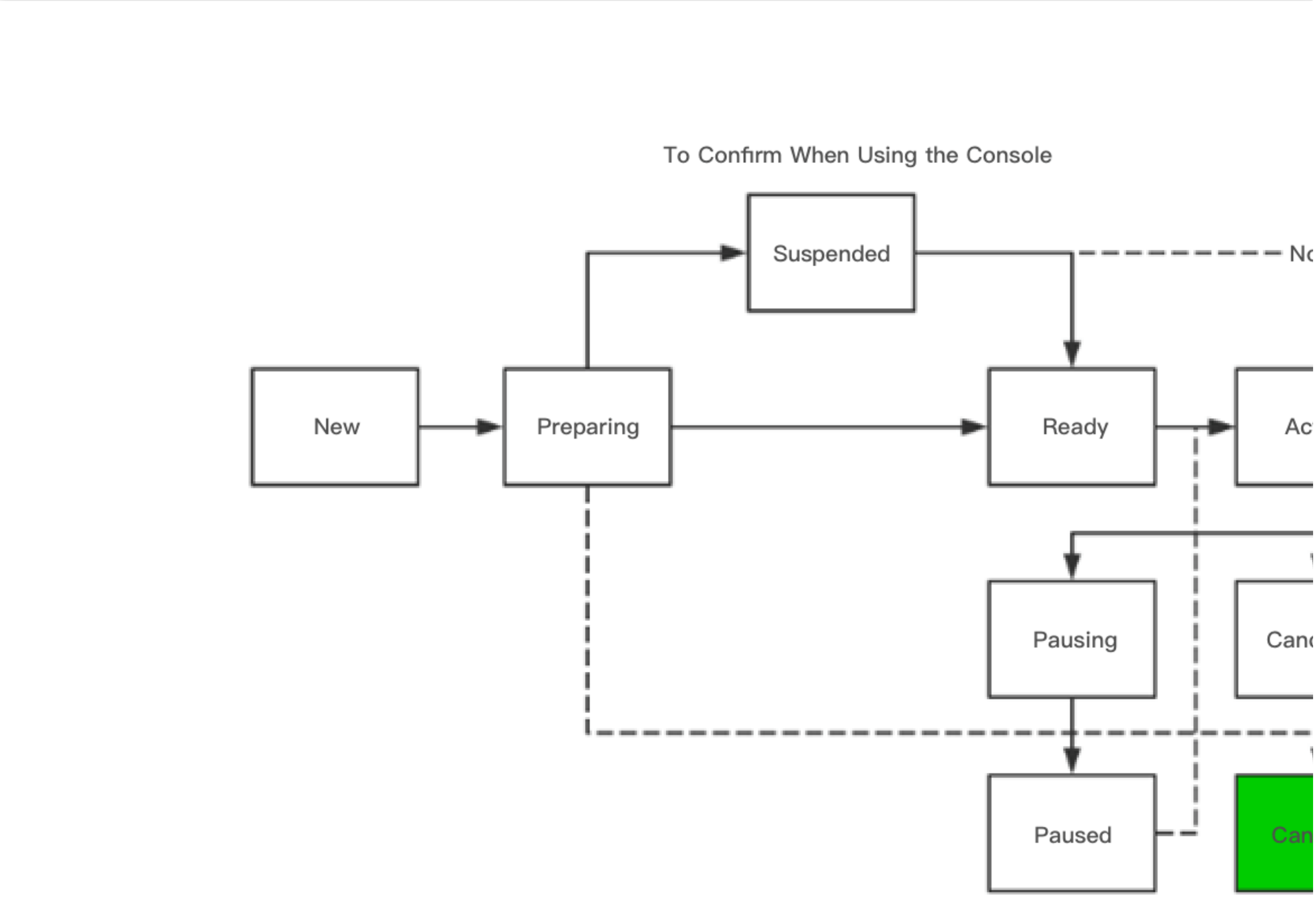
ご自身のタスクに対し、それに応じた優先度を設定することができます。COSはお客様の設定した優先度に応じてバッチ処理タスクを処理します。高優先度のタスクは優先的に実行されます。優先度は整数型の数値で表し、順位は降順、すなわち数値が大きいほど優先度が高いことを表します。優先度はタスクの実行中に変更することができます。タスクの実行中に優先度がより高いタスクが新たに追加された場合は、優先度の低いタスクを一時停止し、高優先度のタスクを実行することができます。

注意：

高優先度のバッチ処理タスクは通常、低優先度のタスクより優先的に実行されますが、優先度はシーケンシャルな実行における唯一の基準とはなりません。複数のバッチ処理タスクを順に実行するよう設定する必要がある場合は、それが実現されるよう、各タスクの実行状態をご自身で監視してください。

タスクのステータス

あるタスクを作成した後、タスクのステータスは次のいくつかのステータスの間で切り替わります。ステータス切り替えの図は次のとおりです。


To Confirm When Using the Console
Suspended
No
New
Preparing
Ready
Ac
Pausing
Canc
Paused
Can

タスクの各ステータスの意味について、次の表で詳しくご説明します。

タスクのステータス	説明	次のステータス
New	1つのタスクが作成されたばかりのとき、タスクのステータスはNewとなります。	Preparingに切り替え可能です。これはCOSがタスク内のリストの解析を開始したことを表します。

Preparing	COSがバッチ処理タスク内に設定されたリストおよびその他の情報の解析を開始した時点で、タスクのステータスはPreparingとなります。	ReadyまたはSuspendedステータスに切り替え可能です。Readyステータスへの切り替えは、バッチ処理タスクの設定情報のCOSによる解析がすでに完了し、COSはその設定に基づいてリスト内のオブジェクトに対し指定された操作を実行できる状態であることを表します。Suspendedステータスへの切り替えは、バッチ処理タスクの設定情報のCOSによる解析はすでに完了しているものの、COSが実行前にお客様による確認を必要としていることを表します。このステータスはコンソールでバッチ処理タスクの設定を行った場合に表示されます。
Suspended	バッチ処理タスクがお客様による確認を待ってから実行する必要がある場合、この確認待ちの状態をSuspendedと呼びます。このステータスはコンソール上でタスクを作成した場合に生成されます。お客様の確認を経て、COSがタスクの実行を開始した場合、このステータスは次のステータスに切り替わり、再びこのステータスに戻ることはありません。	お客様がタスクの実行を確認した時点で、タスクのステータスはReadyに切り替わります。
Ready	COSがバッチ処理タスク内のオブジェクトリストおよびタスク設定情報の解析を完了し、バッチ処理タスクをすぐに実行できる状態になった時点で、タスクのステータスはReadyとなります。	Activeステータスに切り替え可能です。その時点でCOSはタスクの実行を開始します。実行中の高優先度のタスクがある場合は、COSは高優先度のタスクのステータスがCompleteに切り替わるまで、タスクのステータスをReadyのままにします。
Active	COSがバッチ処理タスク内の設定情報に基づいてリスト内のオブジェクトに対する操作を実行中のとき、タスクのステータスはActiveとなります。コンソール上で、またはDescribeJobAPIの呼び出しなどの方法でタスクの進捗状況を照会できます。	Complete、Failing、PausingまたはCancellingなどのステータスに切り替え可能です。タスクが正常に終了または失敗した場合、または中止コマンドによって、例えばタスクのキャンセルなどがあった場合、ステータスは具体的な原因に応じて切り替わります。
Pausing	COSが処理中のバッチ処理タスクを中断した場合は、Pausedの中間ステータスであるPausingに切り替わります。	Pausedステータスに切り替え可能です。実行中だったバッチ処理タスクが中止されたことがCOSによって確認さ

		れた場合、ステータスはPausedに切り替わります。
Paused	高優先度のバッチ処理タスクを作成すると、その時点で実行中だったバッチ処理タスクはPausedステータスに切り替わります。	高優先度のタスクが完了、失敗または確認待ちとなった時点で、Pausedステータスのタスクは自動的にActiveステータスに切り替わります。
Complete	バッチ処理タスクのリスト内の全オブジェクトに対するバッチ操作が正常に完了または失敗した場合、タスクのステータスはCompleteに切り替わります。 タスクレポートを生成するよう設定している場合は、COSがタスクのステータスをCompleteに切り替えた時点で、タスクレポートが指定されたバケットに送信されます。	Completeステータスは最終のステータスであり、あるタスクがCompleteステータスに切り替わった場合、それからさらに他のステータスに切り替わることはありません。
Cancelling	COSが処理中のバッチ処理タスクをキャンセルした場合は、Cancelledの中間ステータスであるCancellingに切り替わります。	Cancelledステータスに切り替え可能です。実行中だったバッチ処理タスクがキャンセルされたことがCOSによって確認された場合、ステータスはCancelledに切り替わります。
Cancelled	バッチ処理タスクが正常にキャンセルされた場合、タスクのステータスはCancelledとなり、その時点でタスクのステータスにはいかなる変更もできなくなります。	Cancelledステータスは最終のステータスであり、あるタスクがCancelledステータスに切り替わった場合、それからさらに他のステータスに切り替わることはありません。
Failing	COSがFailedに切り替える中間ステータスがFailedです 。	Failedステータスに切り替え可能です。
Failed	タスクが失敗すると、タスクのステータスはFailedとなります。タスク失敗の詳細情報に関しては、失敗したタスクの追跡をご参照ください。	Failedステータスは最終のステータスであり、あるタスクがFailedステータスに切り替わった場合、それからさらに他のステータスに切り替わることはありません。

失敗したタスクの追跡

バッチ処理タスクの実行中に、例えばオブジェクトリストの解析が正常に行えないなどの問題が発生した場合は、バッチ処理タスクの実行は失敗となり、COSは対応するエラーコードおよびエラーの原因を返します。COS

はタスク失敗の原因を保存し、お客様はDescribeJobAPIによってタスク失敗の詳細情報を取得することができます。タスクレポートによってタスク失敗の原因およびその他の関連情報を取得することもできます。
COSは各バッチ処理タスクにつき操作失敗閾値を設けることで、作成したタスクに大量の失敗操作が発生することを防止します。1つのタスクに1000以上の操作が存在する場合、COSは操作の失敗率を監視します。任意の時刻における操作失敗率（現在の実行における失敗操作を実行済みの全操作数で割ったもの）が50%の閾値を超えた場合、COSはタスクを終了し、失敗のステータスを返します。操作失敗率が閾値を超過した原因をお客様ご自身で調べることも可能です。例えばオブジェクトリストに大量の存在しないオブジェクト情報が含まれていた場合などは、エラーを修復した後に再度タスクを作成することができます。

注意：

COSのバッチ処理タスクは非同期方式によって実行され、かつオブジェクトに対する操作は必ずしもリストに列記されたオブジェクトの順序に従って行われるわけではありません。このため、オブジェクトリスト内のオブジェクトの順序によって、タスクがどのオブジェクトまで操作を行ったかを判断し、それによって操作の成功または失敗を判断することはできません。ただし、タスクレポートから操作の成功または失敗の情報を取得することはできます。

タスクレポート

タスクを作成する際に、タスクレポートを出力するかどうかを設定できます。タスクレポートを出力するよう設定した場合、COSはタスクの成功、失敗またはタスクがキャンセルされた際にレポートを出力し、お客様はそのレポート上で、タスクが関与したすべての成功または失敗操作の状況を確認することができます。
タスクレポートにはタスクの指定された操作の設定パラメータ、実行ステータスなどの情報が含まれるほか、処理対象のオブジェクト名、バージョンID、操作ステータスコードおよびエラーの説明などの内容も含まれます。

バッチ処理の概要

最終更新日：：2024-06-26 11:09:29

Cloud Object Storage（COS）のバッチ処理機能では、バケット内の指定のオブジェクトリストに対し、指定の操作を実行することが可能です。リスト機能によって生成したオブジェクトリストを指定のオブジェクトリストとするか、または処理したいオブジェクトをリストファイルの形式で、CSV形式のファイルに記録し、COSのバッチ処理機能によってこれらのオブジェクトリストファイルに対しバッチ処理を行います。

リスト機能についてより詳しくお知りになりたい場合は、リスト機能の概要をご参照ください。

現在、COSのバッチ処理機能は次の指定操作のみサポートしています。

オブジェクトの一括コピー

アーカイブオブジェクトの一括復元

COSコンソールでCOSのバッチ処理機能を使用できます。詳細については、バッチ処理をご参照ください。

原理

バッチ処理操作を実行したい場合は、まず初めにバッチ処理タスクを作成する必要があります。バッチ処理タスクにはオブジェクトリストに対し指定の操作を実行するために必要な全ての情報が含まれます。リスト機能によって生成したリストをオブジェクトリストとすることができます。

オブジェクトリストを提供し、バッチ処理タスクを作成してタスクを起動すると、バッチ処理機能がリスト内のそれぞれのオブジェクトに対し、指定された操作を実行します。タスクの実行中は、COSコンソールでそのタスクの実行状況を監視することができ、またタスクの完了後に対応するタスクレポートを出力するよう指定することもできます。タスクレポートには、そのタスクで実行された各操作のステータスが詳細に記述されます。

注意：

バッチ処理タスクは現在のバケット内のオブジェクトに対してのみ有効です。他のバケットのオブジェクトデータのバッチ処理を行いたい場合は、該当するバケット内でバッチ処理機能を有効にしてください。

オブジェクトリスト

オブジェクトリストは処理対象のすべてのオブジェクトを記録したリストです。バッチ処理タスクを作成する場合は先にオブジェクトリストを提供することで、処理したいオブジェクトをCOSに通知する必要があります。このオブジェクトリストファイルをバケット内に保存し、このファイルの名前、ETag、VersionID（もしあれば）などの情報を提供する必要があります。オブジェクトリストは次の2つの方法で作成することができます。

COSリスト機能：この機能はCSV形式でオブジェクトリストを出力するものです。オブジェクトリストの詳細情報に関しては、リスト機能の概要をご参照ください。オブジェクトリストにオブジェクトのバージョンID情報が含まれる場合は、COSによるバッチ処理の実行時に対応するバージョンIDのオブジェクトの処理を行います。

CSVファイルの設定：このファイルには各行に必ずバケット名、処理対象のオブジェクト名を含めなければなりません。バケットで同時にバージョン管理を有効にしている場合は、オブジェクトのバージョンIDも含める必要があります。バージョン管理機能を有効にしていない場合は、オブジェクトのバージョンID情報は無視することができます。CSVファイル設定の形式は次のようになります。

```
examplebucket-appid, exampleobject, PZ9ibn9D5lP6p298B7S9_ceqx1n5EJ0p
examplebucket-appid, exampleobject, jbo9_jhdPEyB4RrmOxWS0kU0EoNrU_oI
```

注意：

バケットのバージョン管理を有効にしている、あるいは以前に有効にしたことがあり、なおかつ指定したオブジェクトのバージョンに対し一括処理を行いたい場合は、オブジェクトリストにオブジェクトのバージョンID情報を必ず記載しなければなりません。

バケットのバージョン管理を有効にしている、あるいは以前に有効にしたことがあっても、オブジェクトリストでバージョンIDを指定していない場合は、COSはデフォルトで最新バージョンのオブジェクトに対し操作を行います。

タスクの作成前に、処理対象のオブジェクトと同名のオブジェクトファイルをアップロードしていた場合、COSはデフォルトで、オブジェクトリスト作成時のオブジェクトではなく、最新バージョンのオブジェクトを処理します。このような操作が行われることを避ける方法は、バージョン管理機能を有効にし、オブジェクトリストでバージョンIDを指定しておくことです。

オブジェクトリストにバケット内のすべてのオブジェクトを含めることは可能ですが、COSが大量のオブジェクトを処理する際、タスクの実行プロセスに長時間かかる可能性があることに注意が必要です。

バッチ処理タスク

ここでは、バッチ処理タスクの作成方法、ならびにバッチ処理タスクの作成完了後のシステムのフィードバック状況について詳細にご説明します。

バッチ処理タスクを作成するには、次の情報を提供する必要があります。

タイプ	説明
操作	リスト内の処理対象のオブジェクトに対し、どのような操作を行うのかを明確にする必要があります。各操作について対応するパラメータを設定することができ、COSはこれらの操作の設定情報に基づいて、リスト内のオブジェクトを順に処理します。
オブジェクトリスト	オブジェクトリストは処理対象のすべてのオブジェクトを記録したファイルです。オブジェクトリストはリスト機能によって作成することができます。詳細については、リスト機能の概要をご参照ください。あるいはお客様ご自身で、処理対象のオブジェクトをリストファイルの形式でCSV形式のファイルに記録し、それをオブジェクトリストとすることもできます。
優先度	優先度を使用して、現在のバッチ処理タスクの、他のバッチ処理タスクに対する優先状態を識別することができます。タスクの優先度はタスク完了の順序を直接決定づけるものではありません。複数のタスクの実行順序を管理したい場合は、タスクの実行状態を自らチェックし、1つのタスクの終了後に次のタスクを開始するようにする必要があります。
ルール権限	バッチ処理タスクを設定後は、アカウントがバッチ処理操作の実行に必要なIAM権限を持つことを保証する必要があります。例えば、 `PUT Object-copy` 操作を一括実行するバッチ処理タスクを設定した場合、ソースバケットが `Get Object` の権限を持つことを確実にすると同時に、ターゲットバケットが `PUT Object` の権限を持つことも確実にしなければなりません。また、どのバッチ処理タスクについても、オブジェクトリストの読み取りおよび

	タスクレポートの書き込み権限を有することを常に保証する必要があります。権限設定に関する詳細情報については、権限設定およびバケットアクセスポリシーをご参照ください。
タスクレポート	タスク完了後にタスクレポートを出力するよう設定することができます。タスクレポートの出力が必要な場合、バッチ処理タスクの作成時に対応するパラメータを入力することで、システムに指定のバケットへタスクレポートを正しく出力させることができます。入力必須情報には、タスクレポートを保存するバケット、タスクレポートの形式、タスクレポートにすべてのタスク情報を含めるかどうかなどがあります。タスクレポートのファイルプレフィックスはオプションです。
タスク説明（オプション）	作成したバッチ処理タスクに256バイトのタスク説明を追加し、タスクの追跡と監視に役立てることができます。タスク説明の詳細情報についてはCOSコンソール上に表示しています。タスク説明によって、作成したタスクの並べ替えやフィルタリングを便利に行うことが可能です。タスク説明は内容が重複していてもよく、類似したタスクに同一のタスク説明（例えば、毎週のログデータ同期コピーなど）を設定することで、同じ種類のタスクの管理を実現することができます。

バッチ処理操作

オブジェクトの一括コピー

最終更新日：：2024-06-26 11:09:29

オブジェクトの一括コピー操作は、リスト内のオブジェクトのコピーに用いることができます。指定したオブジェクトをソースバケットからターゲットバケットに一括コピーすることが可能です。ターゲットバケットの所属リージョンは同一であっても異なっていてもかまいません。オブジェクトの一括コピー操作は `PUT Object-copy` の関連パラメータのカスタム設定をサポートしており、これらの設定情報はレプリカのメタデータ情報またはストレージタイプなどの情報に影響する場合があります。PUT Object-copyに関するその他の説明については、PUT Object-copyをご参照ください。

関連の制限

処理対象のオブジェクトはすべて同一のバケット内にある必要があります。

1つのバッチ処理タスクでは、1つのターゲットバケットのみ設定可能です。

ソースバケットからのオブジェクト読み取り権限およびターゲットバケットへのオブジェクト書き込み権限が承認されている必要があります。

処理対象のオブジェクトは5GB以下とします。

ETagsチェックおよびユーザーのカスタムキーを使用したサーバー側の暗号化はサポートしていません。

ターゲットバケットでバージョン管理が有効になっておらず、なおかつ同名のオブジェクトファイルが存在する場合、COSはオブジェクトファイルの上書き操作を実行します。

リスト内のオブジェクトに複数のバージョンがある場合は、1つのバージョンのみコピーされます。バージョン番号を指定しない場合、デフォルトでは最新バージョンがコピーされます。

アーカイブオブジェクトの一括復元

最終更新日：：2024-06-26 11:09:29

アーカイブの一括復元操作は、リスト内のアーカイブタイプのオブジェクトの復元に用いることができます。この操作はPOST Object restoreの関連パラメータのカスタム設定をサポートしており、これらの設定情報はレプリカの復元時間および有効期限に影響します。POST Object restoreに関するその他の説明については、POST Object restoreをご参照ください。

復元操作は、アーカイブストレージタイプとディープアーカイブストレージタイプの2種類のアーカイブタイプでサポートされています。この2種類のタイプに関するその他の説明については、ストレージタイプの概要をご参照ください。

アーカイブの一括復元タスクを作成するには、次の2つのパラメータを指定する必要があります。

リカバリモード：標準取得モードと一括取得モードがあります。リカバリモードに関するその他の情報については、アーカイブオブジェクトの復元をご参照ください。

レプリカの有効期限：アーカイブタイプのオブジェクトは復元された後、一時的なレプリカを生成します。このレプリカは指定された時間が過ぎると自動的に削除されます。レプリカの有効期限に関するその他の情報については、アーカイブオブジェクトの復元をご参照ください。

注意事項

1. アーカイブの一括復元タスクに、すでに復元が完了したオブジェクトのレプリカが含まれる場合は、アーカイブの一括復元タスクを起動すると、これらのオブジェクトのレプリカの有効期限が更新され、タスク内のすべてのオブジェクトレプリカの有効期限が同じになるよう保証されます。
2. 一括復元タスクはオブジェクトの復元リクエストの送信にのみ用いられます。すべてのリクエストの送信が完了すると、バッチ処理ページでそのタスクの進捗状況が完了となったことが表示されます。ただし、Cloud Object Storage（COS）からは、オブジェクトの復元がいつ完了したかは通知されません。イベント通知を設定すると通知を受信することができます。その他の情報に関しては、イベントの通知をご参照ください。

グローバルアクセラレーション

グローバルアクセラレーションの概要

最終更新日：：2024-06-26 11:09:29

Cloud Object Storage（COS）のグローバルアクセラレーション機能は、トラフィックをグローバルにスケジューリングするTencentのロードバランサシステムを利用して、ユーザーのリクエストをインテリジェントルーティングによって解決し、最適なネットワークアクセスリンクを選択して、リクエストの最寄りのサーバーへのアクセスを実現します。グローバルに分布するデータセンターを利用して、世界各地のユーザーがお客様のバケットにスピーディーにアクセスできるようサポートすることで、ビジネスにおけるアクセス成功率を向上させ、お客様のビジネスの安定と体験の向上をさらに保障します。また、COSのグローバルアクセラレーション機能は、データのアップロードおよびダウンロードのアクセラレーションも実現可能です。

注意：

グローバルアクセラレーション機能はすでにフルリリースされており、中国本土および中国本土以外のパブリッククラウドリージョンでサポートされています。

グローバルアクセラレーション機能をご利用の際は、リクエスト情報がTencent Cloudプライベートネットワークの専用回線経由でアクセラレーションされて転送されるため、料金は有料となります。具体的な価格の詳細については、製品価格をご参照ください。

利用方法

COSコンソール、APIなどの方式でグローバルアクセラレーション機能を有効化できます。

COSコンソールの使用

COSコンソールでバケットのグローバルアクセラレーション機能を有効化することができます。グローバルアクセラレーションの有効化のコンソールドキュメントをご参照ください。

REST APIの使用

次のAPIによってグローバルアクセラレーション機能を直接有効化できます。

PUT Bucket Accelerate

GET Bucket Accelerate

アクセスドメイン名

グローバルアクセラレーション機能を有効にすると、2つのドメイン名でCOSのファイルにアクセスすることができるようになります。

バケットのデフォルトドメイン名：形式は `<BucketName-APPID>.cos.<Region>.myqcloud.com` です。詳細については、リージョンとアクセスドメイン名をご参照ください。
グローバルアクセラレーションドメイン名：形式は `<BucketName-APPID>.cos.accelerate.myqcloud.com` です。
広州リージョンにあるバケット `examplebucket-1250000000` を例にとると、例えばこのバケットでグローバルアクセラレーション機能を有効化しており、業務上、北京からこのバケットにファイル `exampleObject.txt` をアップロードする必要がある場合、次の2つのアップロード方式によって行うことができます。
グローバルアクセラレーションドメイン名を使用したアクセス：アップロードする際にドメイン名を `exampleBucket-1250000000.cos.accelerate.myqcloud.com` に指定する必要があります。このドメイン名によってオブジェクトをアップロードすると、COSサービスはネットワーク状況に応じてインテリジェントに解決し、最寄りのサーバーにアクセスさせます。例えば、リクエストを北京のアクセス層に転送した後、さらにプライベートネットワークの専用回線で広州のストレージ層に転送することでアクセラレーション効果を実現します。
バケットのデフォルトドメイン名を使用したアクセス：アップロードする際にドメイン名を `examplebucket-125000000.cos.ap-guangzhou.myqcloud.com` に指定する必要があります。このドメイン名によってオブジェクトをアップロードすると、リクエストは広州のアクセス層に直接転送され、そこから広州のストレージ層に届きます。この場合、パブリックネットワークリンクが長いため、転送が不安定になる可能性があります。

注意：

グローバルアクセラレーション機能を利用すると別途料金が発生するため、業務の実際の状況に応じて、次の点を慎重に評価されることをお勧めします。

1. 業務上、読み取りより書き込みが多く、かつ遠隔地域からデータをデータセンターにアップロードする必要がある（例えばPUT Object、POST Object、Multipart Uploadなどの操作）場合は、グローバルアクセラレーションドメイン名のご使用をお勧めします。
2. 業務上、書き込みより読み取りが多く、主なユースケースがファイルのダウンロードである（GET Objcet操作など）場合は、CDNアクセラレーションの使用のソリューションを総合的に評価し、コストメリットの最も高いソリューションを選択することをお勧めします。
3. 業務上、設定操作やファイル検索が主である場合は、バケットのデフォルトドメイン名のご使用をお勧めします。
4. 業務上、同一リージョンのプライベートネットワーク環境によって同一リージョンのバケットにアクセスする場合、または専用回線経由でのアクセスの場合は、バケットのデフォルトドメイン名のご使用をお勧めします。

注意事項

グローバルアクセラレーションドメイン名を使用する際の注意事項は次のとおりです。
グローバルアクセラレーションドメイン名を有効にした後、実際に有効になるまでに15分前後かかると見込まれます。ドメイン名が有効になるまでしばらくお待ちください。

グローバルアクセラレーションドメイン名を有効にすると、単一のバケットのアクセラレーションドメイン名を使用したアクセスにおける最大の帯域幅は、ネットワーク全体の業務量に応じて割り当てられます。

グローバルアクセラレーションドメイン名を有効にした後、アクセラレーション効果が得られるのはアクセラレーションドメイン名を使用したリクエストのみとなります。バケットのデフォルトドメイン名は引き続き通常どおり使用できます。

アクセラレーションドメイン名を使用する場合、リクエストリンクがアクセラレーションリンクに該当する場合にのみアクセラレーション料金が発生します。例えば、アクセラレーションドメイン名を使用してデータを北京から北京のバケットにアップロードする場合、リンクがアクセラレーションされていないため、このリクエストにアクセラレーション料金は発生しません。

アクセラレーションドメイン名を使用する場合は、HTTP/HTTPS転送プロトコルを指定することができます。リクエスト情報をプライベートネットワークの専用回線上で転送している場合、COSは、データ転送の安全性確保のためにHTTPSプロトコルで転送を行う必要があるかどうかを、状況に応じて選択します。

課金の例

グローバルアクセラレーションドメイン名を使用したデータアップロードまたはバケットへのアクセスには、追加のアクセラレーション料金が必要となります。アクセラレーション料金は**日次**決済となります。課金項目に関する説明および価格については、課金概要および製品価格をご参照ください。アクセラレーションドメイン名とバケットのデフォルトドメイン名を使用する場合の料金比較を次に示します。

業務シナリオ1

ユーザーの業務が主にビデオファイルのCOSへのアップロードであり、高い転送成功率が求められるケースで、バケットが広州リージョンに設置され、ユーザーは毎日新疆およびシンガポールエリアからそれぞれ1GBのビデオデータを広州のバケットにアップロードする必要があるとします。この場合、30日間の料金は次のようになります。

アクセラレーションドメイン名を使用したアップロードの際に消費するアップロード料金：30 x 1GB x （0.07米ドル/GB + 0.18米ドル/GB） = 7.5米ドル

バケットのデフォルトドメイン名を使用したアップロードの際に消費するアップロード料金：30 x 1GB x （0米ドル/GB） = 0米ドル

説明：

中国本土のアップロードアクセラレーション料金の単価は0.07米ドル/GB、中国本土以外のアップロードアクセラレーション料金の単価は0.18米ドル/GBです。バケットのデフォルトドメイン名を使用してファイルをアップロードする場合、アップロードトラフィック料金は課金されません。

業務シナリオ2

ユーザーの業務が主にアクセラレーションファイルの越境ダウンロードであり、高い転送成功率が求められるケースで、バケットが広州リージョンに設置され、ユーザーは毎日シンガポールエリアから1GBのビデオデータをダウンロードする必要があるとします。この場合、30日間の料金は次のようになります。

アクセラレーションドメイン名を使用したダウンロードの際に消費するアクセラレーショントラフィック料金：30 x 1GB x 0.18米ドル/GB = 5.4米ドル

アクセラレーションドメイン名を使用したダウンロードの際に消費するパブリックネットワークダウンストリームトラフィック料金：30 x 1GB x 0.1米ドル/GB = 3米ドル

上記を総合すると、ダウンロードトラフィック料金の合計は5.4 + 3 = 8.4米ドルとなります。

説明：

越境ダウンロードのアクセラレーション料金単価は0.18米ドル/GBです。グローバルアクセラレーションドメイン名を使用してファイルをダウンロードする際は、**パブリックネットワークダウンストリームトラフィック料金**と**グローバルアクセラレーションダウンストリームトラフィック料金**が必要です。

プライベートネットワークグローバルアクセラレーション

最終更新日：：2024-06-26 11:09:29

Tencent Cloud Object Storage（COS）サービスのプライベートネットワークグローバルアクセラレーション機能は、Tencent Cloudのグローバルトラフィックスケジューリング機能を利用して、ユーザーのリクエストをインテリジェントルーティングによって解決し、最適なネットワークアクセスリンクを選択することで、異なるリージョン間のプライベートネットワークアクセス機能をワンクリックで実装したいユーザーをサポートします。プライベートネットワークグローバルアクセラレーション機能の利用によって、Tencent Cloudプライベートネットワークの基幹回線の安定した伝送品質を活用し、異なるリージョン間でのデータ伝送の安定性とパフォーマンスを向上させることができます。

多くの業務には、異なるリージョンのプライベートネットワーク間でデータをプルしたいというニーズがあります。例えばコンテナイメージ配信のシナリオでは、イメージウエアハウスは集中型のホスティングストレージ方式であるリージョンに保存されている一方、コンテナクラスターは業務上の必要性により異なるリージョンにデプロイされていることがしばしばあります。デプロイの適用の際、コンテナクラスターは異なるリージョンからイメージをプルする必要があり、大量のクロスリージョンリクエストが発生します。プライベートネットワークグローバルアクセラレーションドメイン名を使用すると、このようなシナリオ向けに安定したプライベートネットワーク都市間専用回線による伝送経路を提供でき、イメージ配信の安定性を向上させることができます。

注意：

プライベートネットワークグローバルアクセラレーション機能をご利用の際は、リクエスト情報がTencent Cloudプライベートネットワークの専用回線経由でアクセラレーションされて転送されるため、料金は有料となります。具体的な価格の詳細については、製品価格をご参照ください。

利用方法

COSコンソール、APIなどの方式でグローバルアクセラレーション機能を有効化できます。

COSコンソールの使用

COSコンソールでバケットのグローバルアクセラレーション機能を有効化することができます。グローバルアクセラレーションの有効化のコンソールドキュメントをご参照ください。

REST APIの使用

次のAPIによってグローバルアクセラレーション機能を直接有効化できます。

PUT Bucket Accelerate

GET Bucket Accelerate

アクセスドメイン名

グローバルアクセラレーション機能を有効化すると、プライベートネットワークグローバルアクセラレーションのドメイン名で、プライベートネットワークによってCOSファイルにアクセスできるようになります。プライベートネットワークグローバルアクセラレーションドメイン名の形式は、 `<BucketName-APPID>.cos-internal.accelerate.tencentcos.cn` などとなります。
ある業務を例にとります。この業務に用いるイメージは広州リージョンのバケット `examplebucket-1250000000` に保存され、コンテナクラスターは広州、北京、上海リージョンにそれぞれデプロイされています。この業務チームはコンテナイメージ配信のアクセラレーションのため、バケット `examplebucket-1250000000` のグローバルアクセラレーション機能を有効化しています。そのため、北京、上海、広州リージョンのコンテナクラスターはプライベートネットワークグローバルアクセラレーションドメイン名 `examplebucket-1250000000.cos-internal.accelerate.tencentcos.cn` によって、プライベートネットワーク経由でイメージパッケージをプルすることができます。
広州のコンテナクラスターがこのドメイン名によってファイルのアクセラレーションプルを行う際、COSはリクエストを広州リージョンのプライベートネットワークアクセス層にインテリジェントに解決し、広州リージョンのストレージクラスターから直接ファイルをプルします。
北京/上海リージョンのコンテナクラスターがこのドメイン名によってファイルのアクセラレーションプルを行う際、COSはリクエストを北京/上海リージョンのプライベートネットワークアクセス層にインテリジェントに解決し、プライベートネットワークアクセス層のデバイスによって、都市間基幹ネットワーク専用回線を通じて、広州リージョンのストレージクラスターからファイルをプルします。

注意：

グローバルアクセラレーション機能を利用すると別途料金が発生するため、業務の実際の状況に応じて慎重に評価されることをお勧めします。
プライベートネットワークグローバルアクセラレーションドメイン名はTencent Cloudのプライベートネットワーク環境でのみ使用できます。リクエスト元がTencent Cloudのプライベートネットワーク環境下にない場合は接続できません。その場合はデフォルトのオリジンサーバードメイン名またはデフォルトのグローバルアクセラレーションドメイン名の使用を検討することができます。
他のTencent Cloud製品（CVM、TKE、SCFなど）を介してプライベートネットワークグローバルアクセラレーションドメイン名を使用する場合は、クラウド製品の所属リージョンがCOSのアベイラビリティリージョンの範囲内になければ使用することはできません。COSのアベイラビリティリージョンに関しては、リージョンとアクセスドメイン名をご参照ください。

注意事項

グローバルアクセラレーションドメイン名を使用する際の注意事項は次のとおりです。

グローバルアクセラレーションドメイン名を有効にした後、実際に有効になるまでに15分前後かかると見込まれます。ドメイン名が有効になるまでしばらくお待ちください。

グローバルアクセラレーションドメイン名を有効にすると、単一のバケットのアクセラレーションドメイン名を使用したアクセスにおける最大の帯域幅は、ネットワーク全体の業務量に応じて割り当てられます。

グローバルアクセラレーションドメイン名を有効にした後、アクセラレーション効果が得られるのはアクセラレーションドメイン名を使用したリクエストのみとなります。バケットのデフォルトドメイン名は引き続き通常どおり使用できます。

アクセラレーションドメイン名を使用する場合、リクエストリンクがアクセラレーションリンクに該当する場合にのみアクセラレーション料金が発生します。例えば、アクセラレーションドメイン名を使用してデータを北京から北京のバケットにアップロードする場合、リンクがアクセラレーションされていないため、このリクエストにアクセラレーション料金は発生しません。

アクセラレーションドメイン名を使用する場合は、HTTP/HTTPS転送プロトコルを指定することができます。リクエスト情報をプライベートネットワークの専用回線上で転送している場合、COSは、データ転送の安全性確保のためにHTTPSプロトコルで転送を行う必要があるかどうかを、状況に応じて選択します。

監視とアラーム

最終更新日：：2024-06-26 11:09:29

概要

COSの読み取り/書き込みリクエスト量、トラフィックなどのデータはBCM によって集計され、表示されます。COSコンソールまたはBCMのコンソール上で、COSの読み取り/書き込みリクエスト量、トラフィックなどの詳細な監視データを確認することができます。

説明：

ここでは主に、COSコンソールで統計データを取得するケースについて述べます。データインターフェースを使用してより詳細な情報を取得する方法について知りたい場合は、CMインターフェースをご利用ください。詳細については、CM製品ドキュメントをご参照ください。

現在、CMにレポートされるすべての指標は、全COSリージョンでサポートされています。詳細については、リージョンとアクセスドメイン名をご参照ください。

基本機能

BCMはCOSに次のポータルを提供し、監視およびアラート機能を実現します。

モジュール	機能	主な機能
監視概要	製品の現在のステータスを表示	全体状況、アラートの状況、全体監視情報一覧を提供します
アラート管理	アラートの管理と設定をサポート	COSのアラートポリシー、カスタムメッセージおよびトリガー条件テンプレートの新規追加をサポートします
監視プラットフォーム	トラフィックの監視およびユーザー定義の監視指標データの確認	ユーザーの帯域幅全体情報を確認します。ユーザーは監視指標および報告するデータをあらかじめ定義することができます
クラウド製品監視	COS下のバケットの監視ビューを確認	現在の各バケット下の読み取り/書き込みリクエスト量、トラフィックなどの監視ビューおよびデータを照会します

シナリオ

日常管理のケース：BCMコンソールにログインし、COSの実行ステータスをリアルタイムに確認します。

異常処理のケース：監視データがアラート閾値に達した時点でアラート情報が発出されるため、異常通知を迅速に取得し、異常の原因を照会するとともに、異常状況の処理を速やかに行うことができます。

コンソールからの設定と照会

BCMコンソールからCOSのアラートポリシーを作成することができます。監視指標が設定値に達するとアラートを受信します。関連の操作ガイドについては、モニタリングアラートの設定をご参照ください。
監視データを確認したい場合は、**クラウド製品監視>COS**ページで、すべてのバケットの監視データ、ヘルスステータス、アラートポリシー数を確認することができます。またCOSコンソールでも確認できます。操作ガイドについては、データ概要の確認およびデータモニタリングの照会をご参照ください。

インターフェース呼び出しによる確認

インターフェースを呼び出してCOSの監視データを確認することができます。COSの監視項目は次のとおりです。COSの監視インターフェースの詳細についてお知りになりたい場合は、COS監視指標のドキュメントをご参照ください。

監視指標

説明：
COSは共通リージョンを使用するため、バケットがどのリージョンにあるかにかかわらず、COS監視指標データをプルする際、Regionは常に「広州」リージョンを選択してください。
API Explorerを使用してデータをプルする際は、Regionフィールドは常に「華南リージョン(広州)」を選択します。
SDKを使用してデータをプルする際は、Regionフィールドには常に「ap-guangzhou」と入力します。

リクエストクラス

指標の英語名	指標の日本語名	指標の意味	単位	ディメンション
StdReadRequests	標準ストレージ読み取りリクエスト	標準ストレージタイプの読み取りリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
StdWriteRequests	標準ストレージ書き込み	標準ストレージタイプの書き込み	回	appid、

	リクエスト	リクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます		bucket
IaReadRequests	低頻度ストレージ読み取りリクエスト	低頻度ストレージタイプの読み取りリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
IaWriteRequests	低頻度ストレージ書き込みリクエスト	低頻度ストレージタイプの書き込みリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
DeepArcReadRequests	ディープアーカイブストレージ読み取りリクエスト	ディープアーカイブストレージタイプの読み取りリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
DeepArcWriteRequests	ディープアーカイブストレージ書き込みリクエスト	ディープアーカイブストレージタイプの書き込みリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
ItReadRequests	INTELLIGENT_TIERINGストレージ読み取りリクエスト	INTELLIGENT_TIERINGストレージタイプの読み取りリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
ItWriteRequests	INTELLIGENT_TIERINGストレージ書き込みリクエスト	INTELLIGENT_TIERINGストレージタイプの書き込みリクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
TotalRequests	総リクエスト数	すべてのストレージタイプの読み取り/書き込み総リクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket
GetRequests	GETクラス総リクエスト数	すべてのストレージタイプのGETクラスの総リクエスト回数です。リクエスト回数はリクエストコマ	回	appid、bucket

		ンドの送信回数に基づいて計算されます		
PutRequests	PUTクラス総リクエスト数	すべてのストレージタイプのPUTクラスの総リクエスト回数です。リクエスト回数はリクエストコマンドの送信回数に基づいて計算されます	回	appid、bucket

ストレージクラス

指標の英語名	指標の日本語名	単位	ディメンション
StdStorage	標準ストレージ-ストレージ容量	MB	appid、bucket
SiaStorage	低頻度ストレージ-ストレージ容量	MB	appid、bucket
ItFreqStorage	INTELLIGENT_TIERINGストレージ-高頻度階層ストレージ容量	MB	appid、bucket
ItInfreqStorage	INTELLIGENT_TIERINGストレージ-低頻度階層ストレージ容量	MB	appid、bucket
ArcStorage	アーカイブストレージ-ストレージ容量	MB	appid、bucket
DeepArcStorage	ディープアーカイブストレージ-ストレージ容量	MB	appid、bucket
StdObjectNumber	標準ストレージ-オブジェクト数	個	appid、bucket
IaObjectNumber	低頻度ストレージ-オブジェクト数	個	appid、bucket
ItFreqObjectNumber	INTELLIGENT_TIERINGストレージ_高頻度階層オブジェクト数	個	appid、bucket
ItInfreqObjectNumber	INTELLIGENT_TIERINGストレージ_低頻度階層オブジェクト数	個	appid、bucket
ArcObjectNumber	アーカイブストレージオブジェクト数	個	appid、bucket

DeepArcObjectNumber	ディープアーカイブストレージオブジェクト数	個	appid、bucket
StdMultipartNumber	標準ストレージ-ファイルフラグメント数	個	appid、bucket
IaMultipartNumber	低頻度ストレージ-ファイルフラグメント数	個	appid、bucket
ItFrequentMultipartNumber	INTELLIGENT_TIERING-高頻度ファイルフラグメント数	個	appid、bucket
ArcMultipartNumber	アーカイブストレージ-ファイルフラグメント数	個	appid、bucket
DeepArcMultipartNumber	ディープアーカイブストレージ-ファイルフラグメント数	個	appid、bucket

トラフィッククラス

指標の英語名	指標の日本語名	指標の意味	単位	ディメンション
InternetTraffic	パブリックネットワークダウンストリームトラフィック	データをインターネット経由でCOSからクライアントにダウンロードする際に発生するトラフィック	B	appid、bucket
InternetTrafficUp	パブリックネットワークアップストリームトラフィック	データをインターネット経由でクライアントからCOSにアップロードする際に発生するトラフィック	B	appid、bucket
InternalTraffic	プライベートネットワークダウンストリームトラフィック	データをTencent Cloudプライベートネットワーク経由でCOSからクライアントにダウンロードする際に発生するトラフィック	B	appid、bucket
InternalTrafficUp	プライベートネットワークアップストリームトラフィック	データをTencent Cloudプライベートネットワーク経由でクライアントからCOSにアップロードする際に発生するトラフィック	B	appid、bucket
CdnOriginTraffic	CDN back-to-originトラフィック	データをCOSからTencent Cloud CDNエッジノードに転送する際に発生するトラフィック	B	appid、bucket

InboundTraffic	パブリックネットワーク、プライベートネットワークアップロード総トラフィック	データをインターネット、Tencent Cloudプライベートネットワーク経由でクライアントからCOSにアップロードする際に発生するトラフィック	B	appid、bucket
CrossRegionReplicationTraffic	地域間コピートラフィック	データをあるリージョンのバケットから別のリージョンのバケットに転送する際に発生するトラフィック	B	appid、bucket

戻りコードクラス

指標の英語名	指標の日本語名	指標の意味	単位	ディメンション
2xxResponse	ステータスコード2xx	ステータスコード2xxが返されたリクエストの回数	回	appid、bucket
3xxResponse	ステータスコード3xx	ステータスコード3xxが返されたリクエストの回数	回	appid、bucket
4xxResponse	ステータスコード4xx	ステータスコード4xxが返されたリクエストの回数	回	appid、bucket
5xxResponse	ステータスコード5xx	ステータスコード5xxが返されたリクエストの回数	回	appid、bucket
2xxResponseRate	ステータスコード2xx比率	ステータスコード2xxが返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
3xxResponseRate	ステータスコード3xx比率	ステータスコード3xxが返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
4xxResponseRate	ステータスコード4xx比率	ステータスコード4xxが返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
5xxResponseRate	ステータスコード5xx比率	ステータスコード5xxが返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
400Response	ステータス	ステータスコード400が返されたリクエストの回	回	appid、

	コード400	数		bucket
403Response	ステータスコード403	ステータスコード403が返されたリクエストの回数	回	appid、bucket
404Response	ステータスコード404	ステータスコード404が返されたリクエストの回数	回	appid、bucket
400ResponseRate	ステータスコード400比率	ステータスコード400が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
403ResponseRate	ステータスコード403比率	ステータスコード403が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
404ResponseRate	ステータスコード404比率	ステータスコード404が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
500ResponseRate	ステータスコード500比率	ステータスコード500が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
501ResponseRate	ステータスコード501比率	ステータスコード501が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
502ResponseRate	ステータスコード502比率	ステータスコード502が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket
503ResponseRate	ステータスコード503比率	ステータスコード503が返されたリクエストの回数が総リクエスト回数に占める割合	%	appid、bucket

説明：

1. ステータスコード3xx、4xx、5xxの具体的な詳細についてはエラーコードリストをご確認ください。
2. 各指標の統計粒度（Period）の入手可能な値はそれぞれ異なります。DescribeBaseMetricsインターフェースによって、各指標のサポートする統計粒度を取得することができます。

データ読み取りクラス

指標の英語名	指標の日	指標の意味	単位	ディメンション

	本語名			
StdRetrieval	標準データ読み取り量	標準データ読み取りの際に発生するトラフィックであり、標準ストレージのパブリックネットワークダウンストリームトラフィック、プライベートネットワークダウンストリームトラフィック、CDN back-to-originトラフィックの総和	B	appid、bucket
IaRetrieval	低頻度データ読み取り量	低頻度データ読み取りの際に発生するトラフィックであり、低頻度ストレージのパブリックネットワークダウンストリームトラフィック、プライベートネットワークダウンストリームトラフィック、CDN back-to-originトラフィックの総和	B	appid、bucket

データ処理クラス

インターフェースを呼び出してCloud Infiniteの監視データを確認することができます。Cloud Infiniteの監視インターフェースの詳細について知りたい場合は、Cloud Infinite監視指標のドキュメントをご参照ください。

各ディメンションに対応するパラメーター覧

パラメータ名	ディメンション名	ディメンションの説明	形式
&Instances.N.Dimensions.0.Name	appid	ルートアカウントAPPIDのディメンション名	Stringタイプのディメンション名を入力：appid
&Instances.N.Dimensions.0.Value	appid	ルートアカウントの具体的なAPPID	ルートアカウントAPPIDを入力（例：1250000000）
&Instances.N.Dimensions.1.Name	bucket	バケットのディメンション名	Stringタイプのディメンション名を入力：bucket
&Instances.N.Dimensions.1.Value	bucket	具体的なバケット名	具体的なバケット名を入力（例：examplebucket-1250000000）

入力パラメータの説明

COS監視データを照会する際の入力パラメータ値は次のとおりです。

&Namespace=QCE/COS
&Instances.N.Dimensions.0.Name=appid
&Instances.N.Dimensions.0.Value=ルートアカウントのAPPID
&Instances.N.Dimensions.1.Name=bucket
&Instances.N.Dimensions.1.Value=バケット名

監視の説明

監視間隔：BCMは、リアルタイム、直近24時間、直近7日間、カスタマイズした日付などの様々な統計区間におけるデータ監視を行うことができます。時間粒度は1分、5分、1時間、1日をサポートしています。

データストレージ：粒度が1分間隔の監視データは15日間、粒度が5分間隔の監視データは31日間保存されます。粒度が1時間間隔の監視データは93日間、粒度が1日間隔の監視データは186日間保存されます。

アラート表示：BCMはCOSの監視データを統合し、データを見やすいチャート形式で表示します。製品ごとにあらかじめ定義したアラート指標に基づいてアラートを発出することができ、ユーザーが全体的な実行状況を把握するのに役立ちます。

アラート設定：監視指標の閾値を設定します。監視データがアラート条件をトリガーした時点で、CMが当該グループに速やかにアラート情報を送信することができます。詳細については、アラートの概要およびモニタリングアラートの設定をご参照ください。