ALIBABA CLOUD

Alibaba Cloud

对象存储 OSS

最佳實踐

Document Version: 20220118

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.網站與行動裝置 App

1.1. Web端直傳實踐

1.1.1. Web端直傳實踐簡介

本教程介紹如何在Web端通過表單上傳方式直接上傳數據到OSS。

背景

每個OSS的用戶都會用到上傳服務。Web端常見的上傳方法是用戶在瀏覽器或app端上傳檔案到應用伺服器，然後應用伺服器再把檔案上傳到OSS。

和數據直傳到OSS相比，以上方法有三個缺點：

- 上傳慢。先上傳到應用伺服器，再上傳到OSS，網路傳送比直傳到OSS多了一倍。如果直傳到OSS，不通過應用伺服器，速度將大大提升，而且OSS採用BGP頻寬，能保證各地各電訊廠商的速度。
- 擴充性差。如果後續用戶多了，應用伺服器會成為瓶頸。
- 費用高。需要準備多台應用伺服器。由於OSS上傳流量是免費的，如果數據直傳到OSS，不通過應用伺服器，那麼將能省下幾台應用伺服器。

目的

本教程的目的是通過以下三個例子介紹如何通過表單直傳數據到OSS：

- JavaScript客戶端簽名直傳講解在客戶端通過JavaScript程式碼完成簽名，然後通過表單直傳數據到OSS。
- 服務端簽名後直傳講解在服務端通過PHP程式碼完成簽名，然後通過表單直傳數據到OSS。
- 服務端簽名直傳並設定上傳回調講解在服務端通過PHP程式碼完成簽名，並且服務端設定了上傳後回調，然後通過表單直傳數據到OSS。OSS回調完成後，應用伺服器再返回結果給客戶端。

1.1.2. JavaScript客戶端簽名直傳

本樣本講解如何在客戶端通過JavaScript程式碼完成簽名，然後通過表單直傳數據到OSS。

Demo

PC瀏覽器測試樣例

本樣本採用Plupload 直接提交表單數據（即PostObject）到OSS，可以運行在PC瀏覽器、手機瀏覽器、微信等。您可以同時選擇多個檔案上傳，並設定上傳到指定目錄和設定上傳檔案名字是隨機檔案名還是本地檔案名。您還可以通過進度條查看上傳進度。

> **說明** 檔案上傳到一個測試的公共Bucket，會定時清理，所以不要傳一些敏感及重要數據。

步驟 1：下載並安裝Plugload

Plupload是一款簡單易用且功能強大的檔案上傳工具， 支援多種上傳方式，包括html5、flash、silverlight,、html4。它會智能檢測當前環境，選擇最適合的上傳方式，並且會優先採用Html5方式。請參見Plupload官網進行下載和安裝。

步驟 2：下載應用伺服器代碼

下載地址

步驟 3：修改設定檔

將下載包解壓後，修改upload.js檔案：

```
accessid= '<yourAccessKeyId>';
accesskey= <yourAccessKeySecrety';
host = 'http://post-test.oss-cn-hangzhou.aliyuncs.com';
```

- $id：您的AccessKeyId
- $key：您的AessKeySecret
- $host：格式為 `BucketName.Endpoint` ，例如 `post-test.oss-cn-hangzhou.aliyuncs.com`

> **說明** 關於Endpoint的介紹，請參見Endpoint（訪問網域名稱）。

步驟 4：設定CORS

HTML表單直接上傳到OSS會產生跨域請求。為了瀏覽安全，需要為Bucket設定跨域規則（CORS），支援Post方法。

具體操作步驟請參見設定跨域訪問。設定如下圖所示：

> **說明** 在低版本IE瀏覽器，Plupload會以flash方式執行。您需要設定crossdomain.xml，設定方法請參見OSS Web直傳—使用Flash上傳。

步驟 5：體驗JavaScript客戶端簽名直傳

1. 將應用伺服器代碼zip包解壓到Web根目錄下。
2. 在Web瀏覽器中輸入 `<Web應用伺服器位址>/oss-h5-upload-js-direct/index.html` ，例如 `http://abc.com:8080/oss-h5-upload-js-direct/index.html` 。
3. 選擇一個或多個檔案進行上傳。
4. 上傳成功後，通過控制台查看上傳結果。

核心代碼解析

因為OSS支援POST協議，所以只要在Plupload發送POST請求時帶上OSS簽名即可。核心代碼如下：

```
var uploader = new plupload.Uploader({
    runtimes : 'html5,flash,silverlight,html4',
    browse_button : 'selectfiles',
    //runtimes : 'flash',
    container: document.getElementById('container'),
    flash_swf_url : 'lib/plupload-2.1.2/js/Moxie.swf',
    silverlight_xap_url : 'lib/plupload-2.1.2/js/Moxie.xap',
    url : host,
    multipart_params: {
        'Filename': '${filename}',
        'key' : '${filename}',
        'policy': policyBase64,
        'OSSAccessKeyId': accessid,
        'success_action_status' : '200', //讓服務端返回200，不設定則預設返回204
        'signature': signature,
    },
  ....
}
```

上述代碼中，`'Filename': '${filename}'` 表示上傳後保持原來的檔案名。如果您想上傳到特定目錄如abc下，且檔案名不變，請修改代碼如下：

```
multipart_params: {
        'Filename': 'abc/' + '${filename}',
        'key' : '${filename}',
        'policy': policyBase64,
        'OSSAccessKeyId': accessid,
        'success_action_status' : '200', //讓服務端返回200，不設定則預設返回204
        'signature': signature,
    },
```

- 設定成隨機檔案名

  如果想在上傳時固定設定成隨機檔案名，尾碼保持跟客戶端檔案一致，可以將函數改為：

  ```
  function check_object_radio() {
      g_object_name_type = 'random_name';
  }
  ```

- 設定成用戶的檔案名

  如果想在上傳時固定設定成用戶的檔案名，可以將函數改為：

  ```
  function check_object_radio() {
      g_object_name_type = 'local_name';
  }
  ```

- 設定上傳目錄

  您可以將檔案上傳到指定目錄下。下面的代碼是將上傳目錄改成abc/，注意目錄必須以正斜線（/）結尾。

```
function get_dirname()
{
    g_dirname = "abc/";
}
```

- 上傳簽名

  上傳簽名主要是對policyText進行簽名，最簡單的例子如下：

```
var policyText = {
    "expiration": "2020-01-01T12:00:00.000Z", // 設定Policy的失效時間，如果超過失效時間，就無法通過此Policy上傳檔案
    "conditions": [
    ["content-length-range", 0, 1048576000] // 設定上傳檔案的大小限制，如果超過限制，檔案上傳到OSS會報錯
    ]
}
```

總結

在客戶端通過JavaScript程式碼完成簽名，無需過多配置，即可實現直傳，非常方便。但是客戶端通過JavaScript把AccesssKeyID 和AccessKeySecret寫在代碼裡面有泄露的風險，建議採用服務端簽名後直傳。

1.1.3. 服務端簽名後直傳

本樣本講解如何在服務端通過PHP程式碼完成簽名，然後通過表單直傳數據到OSS。

> **說明** 本樣本無法實現分區上傳與斷點續傳。

背景

採用JavaScript客戶端直接簽名（參見JavaScript客戶端簽名直傳）有一個嚴重的安全隱患：OSS AccessKey暴露在前端頁面，這是非常不安全的做法。因此，OSS提供了服務端簽名後直傳的方案。

Demo

您可以通過樣例體驗服務端簽名後直傳效果：PC瀏覽器測試樣例

原理介紹

服務端簽名後直傳的邏輯圖如下：

流程如下：

1. 用戶發送上傳Policy請求到應用伺服器。
2. 應用伺服器返回上傳Policy和簽名給用戶。
3. 用戶使用Plupload直接上傳數據到OSS。

步驟 1：下載並安裝Plugload

Plupload是一款簡單易用且功能強大的檔案上傳工具， 支援多種上傳方式，包括html5、flash、silverlight,、html4。它會智能檢測當前環境，選擇最適合的上傳方式，並且會優先採用Html5方式。請參見Plupload官網進行下載和安裝。

步驟 2：下載應用伺服器代碼

- PHP：下載地址
- Java：下載地址
- Python：下載地址
- Go：下載地址

步驟 3：修改設定檔

本樣本採用PHP編寫。將下載包解壓後，修改以下檔案：

- php/get.php檔案：

```
$id= '<yourAccessKeyId>';
 $key= '<yourAccessKeySecret>';
 $host = 'http://post-test.oss-cn-hangzhou.aliyuncs.com
```

  - $id：您的AccessKeyId
  - $key：您的AessKeySecret
  - $host：格式為 `BucketName.Endpoint` ，例如 `post-test.oss-cn-hangzhou.aliyuncs.com`

    > **說明** 關於Endpoint的介紹，請參見Endpoint（訪問網域名稱）。

- upload.js檔案

  將變數severUrl改成伺服器部署的地址，例如 `http://abc.com:8080/oss-h5-upload-js-php/get.php` 。

步驟 4：設定CORS

HTML表單直接上傳到OSS會產生跨域請求。為了瀏覽安全，需要為Bucket設定跨域規則（CORS），支援Post方法。

具體操作步驟請參見設定跨域訪問。設定如下圖所示：

> **說明** 在低版本IE瀏覽器，Plupload會以Flash方式執行。您需要設定crossdomain.xml，設定方法請參見OSS Web直傳—使用Flash上傳。

步驟 5：體驗服務端簽名後直傳

1. 將應用伺服器代碼zip包解壓到Web根目錄下。
2. 在Web瀏覽器中輸入 `<Web應用伺服器位址>/oss-h5-upload-js-php/index.html` ，例如 `http://abc.com:8080/oss-h5-upload-js-php/index.html` 。
3. 選擇一個或多個檔案進行上傳。
4. 上傳成功後，通過控制台查看上傳結果。

核心代碼解析

- 設定成隨機檔案名

  如果想在上傳時固定設定成隨機檔案名，尾碼保持跟客戶端檔案一致，可以將函數改為：

```
function check_object_radio() {
    g_object_name_type = 'random_name';
}
```

- 設定成用戶的檔案名

  如果想在上傳時固定設定成用戶的檔案名，可以將函數改為：

```
function check_object_radio() {
    g_object_name_type = 'local_name';
}
```

- 設定上傳目錄

  上傳的目錄由服務端（即PHP）指定， 每個客戶端只能上傳到指定的目錄，實現安全隔離。下面的代碼是將上傳目錄改成abc/，注意目錄必須以正斜線（/）結尾。

```
$dir = 'abc/';
```

- 設定上傳過濾條件

  您可以利用Plupload的屬性filters設定上傳的過濾條件，如設定只能上傳圖片、上傳檔案的大小、不能有重複上傳等。

```
var uploader = new plupload.Uploader({
    ……
    filters: {
        mime_types : [ //只允許上傳圖片和zip檔案
        { title : "Image files", extensions : "jpg,gif,png,bmp" },
        { title : "Zip files", extensions : "zip" }
        ],
        max_file_size : '400kb', //最大隻能上傳400KB的檔案
        prevent_duplicates : true //不允許選取重複檔案
    },
```

  - mime_types：限制上傳的檔案尾碼
  - max_file_size：限制上傳的檔案大小
  - prevent_duplicates：限制不能重複上傳

  > **說明** filters過濾條件不是必須的。如果不想設定過濾條件，只要把該項注釋即可。

- 獲取上傳後的檔案名

  如果要知道檔案上傳成功後的檔案名，可以用Plupload調用FileUploaded事件獲取，如下所示：

```
FileUploaded: function(up, file, info) {
            if (info.status == 200)
            {
                document.getElementById(file.id).getElementsByTagName('b')[0].innerHTML =
'upload to oss success, object name:' + get_uploaded_object_name(file.name);
            }
            else
            {
                document.getElementById(file.id).getElementsByTagName('b')[0].innerHTML =
info.response;
            }
    }
```

可以利用如下函數，得到上傳到OSS的檔案名，其中file.name記錄了上傳本地檔案的名稱。

```
get_uploaded_object_name(file.name)
```

- 上傳簽名

  JavaScript可以從服務端獲取policyBase64、accessid、signature這三個變數，獲取這三個變數的核心代碼如下：

```
phpUrl = './php/get.php'
        xmlhttp.open( "GET", phpUrl, false );
        xmlhttp.send( null );
        var obj = eval ("(" + xmlhttp.responseText+ ")");
        host = obj['host']
        policyBase64 = obj['policy']
        accessid = obj['accessid']
        signature = obj['signature']
        expire = parseInt(obj['expire'])
        key = obj['dir']
```

  xmlhttp.responseText解析如下：

  > **說明** 以下僅為樣本，並不要求必須是相同的格式，但是必須有accessid、policy、signature這三個值。

```
{"accessid":"6MKOxxxxxx4AUk44",
"host":"http://post-test.oss-cn-hangzhou.aliyuncs.com",
"policy":"eyJleHBpcmF0aW9uIjoiMjAxNS0xMS0wNVQyMDoyMzoyM1oiLCJjxb25kaXRpb25zIjpbWyJjcb250ZW50LWxlbmd0aC1yYW5nZSIsMCwxMDQ4NTc2MDAwXSxbInN0YXJ0cy13aXRoIiwiJGtleSIsInVzZXItZGlyXC8iXV19",
"signature":"I2u57FWjTKqX/AE6doIdyff151E=",
"expire":1446726203,"dir":"user-dir/"}
```

  - accessid：用戶請求的accessid。
  - host：用戶要往哪個網域名稱發送上傳請求。
  - policy：用戶表單上傳的策略（Policy），是經過base64編碼過的字元串。
  - signature：對變數policy簽名後的字元串。

- expire：上傳策略失效時間，在PolicyText裡指定。在失效時間之前，都可以利用此Policy上傳檔案，所以沒有必要每次上傳都去服務端獲取簽名。

  > **說明** 為了減少服務端的壓力，設計思路是：初始化上傳時，每上傳一個檔案後，獲取一次簽名。然後再上傳時，比較目前時間與簽章時間，看簽章時間是否失效。如果失效了，就重新獲取一次簽名，如果沒有失效，就使用之前的簽名。這裡就用到了變數expire，核心代碼如下：

  ```
  now = timestamp = Date.parse(new Date()) / 1000;
  [color=#000000]//可以判斷當前expire是否超過了目前時間，如果超過了目前時間，就重新取一次簽名，緩衝時間為3[/color]
      if (expire < now + 3)
  {
          .....
          phpUrl = './php/get.php'
          xmlhttp.open( "GET", phpUrl, false );
          xmlhttp.send( null );
          ......
  }
  return .
  ```

解析Policy的內容如下：

```
{"expiration":"2015-11-05T20:23:23Z",
"conditions":[["content-length-range",0,1048576000],
["starts-with","$key","user-dir/"]]
```

> **說明** Policy的詳細資料請參見Policy基本元素。

上面Policy中增加了starts-with，用來指定此次上傳的檔案名必須以user-dir開頭，用戶可自行指定此字元串。增加starts-with的原因是：在很多場景下，一個應用對應一個Bucket，為了防止數字覆蓋，每個用戶上傳到OSS的檔案都可以有特定的首碼。這樣就存在一個問題，用戶獲取到這個Policy後，在失效期內都能修改上傳首碼，從而上傳到別人的目錄下。為了解決這個問題，可以設定應用伺服器在上傳時就指定用戶上傳的檔案必須是某個首碼。這樣如果用戶獲取到了Policy也沒有辦法上傳到別人的首碼上，從而保證了數據的安全性。

總結

本樣本中，web端向服務端請求籤名，然後直接上傳，不會對服務端產生壓力，而且安全可靠。但是這個樣本有個問題，就是用戶上傳了多少檔案，上傳了什麼檔案，服務端並不能馬上知道，如果想即時了解用戶上傳了什麼檔案，可以採用服務端簽名直傳並設定上傳回調。

1.1.4. 服务端签名直传并设置上传回调

1.1.4.1. 服務端簽名直傳並設定上傳回調

本文主要介紹如何基於POST Policy的使用規則在服務端通過各種語言程式碼完成簽名，並且設定上傳回調，然後通過表單直傳資料到OSS。

背景

採用服務端簽名後直傳方案有個問題：大多數情況下，使用者上傳資料後，應用伺服器需要知道使用者上傳了哪些檔案以及檔案名稱；如果上傳了圖片，還需要知道圖片的大小等。為此OSS提供了上傳回調方案。OSS回調完成後，應用伺服器再將結果返回給用戶端，以便服務端即時瞭解使用者上傳了什麼檔案。

流程介紹

流程如下：

1. 使用者嚮應用伺服器請求上傳Policy和回調。
2. 應用伺服器返回上傳Policy和回調設定。
3. 使用者直接向OSS傳送檔案上傳請求。
4. OSS根據使用者的回調設定，發送回調請求給應用伺服器。
5. 應用伺服器返迴響應給OSS。
6. OSS將應用伺服器返回的內容返回給使用者。

當使用者要上傳一個檔案到OSS，而且希望將上傳的結果返回給應用伺服器，這時就需要設定一個回呼函數，將請求告知應用伺服器。使用者上傳完檔案後，不會直接得到返回結果，而是先通知應用伺服器，再把結果轉達給使用者。

流程解析

服務端簽名直傳並設定上傳回調提供了PHP、Java、Python、Go、Ruby語言版本，流程如下：

1. 使用者嚮應用伺服器請求上傳Policy和回調。
2. 應用伺服器返回上傳Policy和回調設定代碼。

   應用伺服器側的簽名直傳服務會處理用戶端發過來的GET請求訊息，您可以設定對應的代碼讓應用伺服器能夠給用戶端返回正確的訊息。各個語言版本的配置文檔中都有明確的說明供您參考。
3. 使用者直接向OSS傳送檔案上傳請求。
4. OSS根據使用者的回調設定，發送回調請求給應用伺服器。

   用戶端上傳檔案到OSS結束後，OSS解析用戶端的上傳回調設定，發送POST回調請求給應用伺服器。
5. 應用伺服器返迴響應給OSS。

   應用伺服器根據OSS發送訊息中的 `authorization` 來進行驗證，如果驗證通過，則向OSS返回如下json格式的成功訊息。

   ```
   String value: OK
   Key: Status
   ```
6. OSS將應用伺服器返回的訊息返回給使用者。

用戶端源碼解析

用戶端源碼下載地址：aliyun-oss-appserver-js-master.zip

> **說明** 用戶端JavaScript代碼使用的是Plupload組件。Plupload是一款簡單易用且功能強大的檔案上傳工具，支援多種上傳方式，包括html5、flash、silverlight、html4。它會智能檢測當前環境，選擇最適合的上傳方式，並且會優先採用html5方式。詳情請參見Plupload官網。

下面舉例介紹幾個關鍵功能的代碼實現：

- 設定成隨機檔案名稱

若上傳時採用固定格式的隨機檔案名稱，且尾碼跟用戶端檔案名稱保持一致，可以將函數改為：

```
function check_object_radio() {
    g_object_name_type = 'random_name';
}
```

- 設定成使用者的檔案名稱

  如果想在上傳時設定成使用者的檔案名稱，可以將函數改為：

```
function check_object_radio() {
    g_object_name_type = 'local_name';
}
```

- 設定上傳目錄

  上傳的目錄由服務端指定， 每個用戶端只能上傳到指定的目錄，實現安全隔離。下面的代碼以PHP為例，將上傳目錄改成abc/，注意目錄必須以正斜線（/）結尾。

```
$dir ='abc/';
```

- 設定上傳過濾條件

  您可以利用Plupload的屬性filters設定上傳的過濾條件，如設定只能上傳圖片、上傳檔案的大小、不能有重複上傳等。

```
var uploader = new plupload.Uploader({
    ……
    filters: {
        mime_types : [ //只允許上傳圖片和zip檔案
        { title : "Image files", extensions : "jpg,gif,png,bmp" },
        { title : "Zip files", extensions : "zip" }
        ],
        max_file_size : '400kb', //最大隻能上傳400KB的檔案
        prevent_duplicates : true //不允許選取重複檔案
    },
```

  - mime_types：限制上傳的檔案尾碼。
  - max_file_size：限制上傳的檔案大小。
  - prevent_duplicates：限制不能重複上傳。

- 擷取上傳後的檔案名稱

  如果要知道檔案上傳成功後的檔案名稱，可以用Plupload調用FileUploaded事件擷取，如下所示：

```
FileUploaded: function(up, file, info) {
            if (info.status == 200)
            {
                document.getElementById(file.id).getElementsByTagName('b')[0].innerHTML =
'upload to oss success, object name:' + get_uploaded_object_name(file.name);
            }
            else
            {
                document.getElementById(file.id).getElementsByTagName('b')[0].innerHTML =
info.response;
            }
    }
```

可以利用如下函數，得到上傳到OSS的檔案名稱，其中file.name記錄了本地檔案上傳的名稱。

```
get_uploaded_object_name(file.name)
```

- 上傳簽名

  JavaScript可以從服務端擷取policyBase64、accessid、signature這三個變數，核心代碼如下：

```
function get_signature()
{
    // 判斷expire的值是否超過了目前時間，如果超過了目前時間，就重新擷取簽名，緩衝時間為3秒。
    now = timestamp = Date.parse(new Date()) / 1000;
    if (expire < now + 3)
    {
        body = send_request()
        var obj = eval ("(" + body + ")");
        host = obj['host']
        policyBase64 = obj['policy']
        accessid = obj['accessid']
        signature = obj['signature']
        expire = parseInt(obj['expire'])
        callbackbody = obj['callback']
        key = obj['dir']
        return true;
    }
    return false;
};
```

從服務端返回的訊息解析如下：

> 說明 以下僅為樣本，並不要求相同的格式，但必須有accessid、policy、signature三個值。

```
{"accessid":"6MKO******4AUk44",
"host":"http://post-test.oss-cn-hangzhou.aliyuncs.com",
"policy":"eyJleHBpcmF0aW9uIjoiMjAxNS0xMS0wNVQyMDoyMzoyM1oiLCJjxb25kaXRpb25zIjpbWyJjcb250ZW50LWxlbmd0aC1yYW5nZSIsMCwxMDQ4NTc2MDAwXSxbInN0YXJ0cy13aXRoIiwiJGtleSIsInVzZXItZGlyXC8iXV19",
"signature":"I2u57FWjTKqX/AE6doIdyff151E=",
"expire":1446726203,"dir":"user-dir/"}
```

  - accessid：使用者請求的accessid。

- host：使用者要往哪個網域名稱發送上傳請求。
- policy：使用者表單上傳的策略（Policy），是經過base64編碼過的字串。詳情請參見Post Policy。
- signature：對變數policy簽名後的字串。
- expire：上傳策略Policy失效時間，在服務端指定。失效時間之前都可以利用此Policy上傳檔案，無需每次上傳都去服務端擷取簽名。

> **說明** 為了減少服務端的壓力，設計思路是：初始化上傳時，每上傳一個檔案，擷取一次簽名。再上傳時，比較目前時間與簽章時間，看簽章時間是否失效。如果失效，就重新擷取一次簽名，如果沒有失效，就使用之前的簽名。

解析Policy的內容如下：

```
{"expiration":"2015-11-05T20:23:23Z",
"conditions":[["content-length-range",0,1048576000],
["starts-with","$key","user-dir/"]]
```

上面Policy中增加了starts-with，用來指定此次上傳的檔案名稱必須以user-dir開頭，使用者也可自行指定。增加starts-with的原因是：在很多情境下，一個應用對應一個Bucket，為了防止資料覆蓋，每個使用者上傳到OSS的檔案都可以有特定的首碼。但這樣存在一個問題，使用者擷取到這個Policy後，在失效期內都能修改上傳首碼，從而上傳到別人的目錄下。解決方案為，在應用伺服器端就指定使用者上傳檔案的首碼。如果使用者擷取了Policy也沒有辦法上傳到別人的目錄，從而保證了資料的安全性。

- 設定應用伺服器的地址

  在用戶端源碼 `upload.js` 檔案中，如下程式碼片段的變數 `serverUrl` 的值可以用來設定簽名伺服器的URL，設定好之後，用戶端會向該 `serverUrl` 發送GET請求來擷取需要的資訊。

```
// serverUrl是 使用者擷取簽名和Policy等資訊的應用伺服器的URL，請將下面的IP和Port配置為您自己的真實資訊。
serverUrl = 'http://88.88.88.88:8888'
```

1.2. 行動裝置 App端直傳實踐

1.2.1. 快速搭建行動裝置 App直傳服務

本文介紹如何在30分鐘內搭建一個基於OSS的行動裝置 App數據直傳服務。直傳指的是行動裝置 App數據的上傳和下載直接連接OSS，只有控制流程串連自己的伺服器。

背景

在移動互聯的時代，手機app上傳的數據越來越多。作為開發人員，您可以利用OSS處理各種資料存放區需求，從而更加專註於自己的應用邏輯。

基於OSS的行動裝置 App數據直傳服務具有以下優勢：

- 數據安全：使用靈活的授權和鑒權方式進行數據的上傳和下載，更加安全。
- 成本低廉：您不需要準備很多伺服器。行動裝置 App直接連接雲端儲存OSS，只有控制流程串連應用伺服器。
- 高並發：支援海量用戶並發訪問。
- 彈性擴容：無限擴容的儲存空間。
- 資料處理：和圖片處理以及音視頻轉碼搭配使用，方便靈活地進行資料處理。

下載並安裝行動裝置 App

行動裝置 App源碼的下載地址如下：

- Android：下載地址
- iOS：下載地址

您可以通過此行動裝置 App上傳圖片到OSS。上傳的方法支援普通上傳和斷點續傳上傳。在網路環境差的情況下，推薦使用斷點續傳上傳。您還可以利用圖片處理服務，對要上傳的圖片進行縮略和加浮水印處理。樣本應用的最終效果圖如下：

- 應用伺服器：該行動裝置 App對應的後台應用伺服器。請使用步驟2：下載應用伺服器程式碼範例中提供的應用伺服器程式碼範例，部署自己的應用伺服器。
- 上傳Bucket：該行動裝置 App要把數據上傳到哪個Bucket。
- 區域：上傳Bucket所在的地域。

原理介紹

行動裝置 App直傳服務的架構圖如下所示：

- Android/iOS 行動裝置 App：即終端使用者手機上的app。
- OSS：即阿里雲對象儲存，負責儲存app上傳的數據。
- RAM/STS：負責生成臨時上傳憑證，即Token。
- 應用伺服器：即提供該Android/iOS應用的開發人員開發的app後台服務，用於管理app上傳和下載的Token，以及用戶通過app上傳數據的元資訊。

實現步驟如下：

1. 行動裝置 App嚮應用伺服器申請一個臨時上傳憑證。

   > **說明** Android和iOS應用不能直接儲存AccessKey，這樣會存在數據泄露的風險。所以應用必須向用戶的應用伺服器申請一個Token。這個Token是有時效性的，如果Token的過期時間是30分鐘（由應用伺服器指定），那麼在這30分鐘裡，該Android/iOS應用可以使用此Token從OSS上傳和下載數據， 30分鐘後需要重新獲取Token。

2. 應用伺服器檢測上述請求的合法性，然後返回Token給行動裝置 App。
3. Android/iOS行動裝置 App使用此Token將數據上傳到OSS，或者從OSS下載數據。

以下介紹應用伺服器如何生成Token以及Android/iOS行動裝置 App如何獲取Token。

步驟1：建立Bucket並開通STS服務

1. 開通OSS，並且建立Bucket。
2. 開通STS服務。
   i. 登入OSS管理主控台。
   ii. 在OSS**概覽**頁中找到基礎配置區域，單擊**安全性權杖**。

   iii. 進入到**安全性權杖快捷配置**頁面，單擊**開始授權**。

   > **說明** 如果沒有開通RAM，會彈出開通的對話方塊。單擊**開通**。開通完成後單擊**開始授權**。

iv. 系統進行自動授權。請保存AccessKeyID、AccessKeySecret和RoleArn三個參數。

- 如果您未建立過AccessKey，系統自動建立AccessKey。請保存AccessKeyId、AccessKeySecret和RoleArn，如下圖所示。

- 如果您之前已經建立過AccessKey，也可以單擊如下圖所示的查看建立新的AccessKey。

步驟2：下載應用伺服器程式碼範例

- PHP：下載地址
- Java：下載地址
- Ruby：下載地址
- Node.js：下載地址

步驟3：修改設定檔

以下例子採用PHP編寫。每個語言的範例程式碼下載後，都會有一個設定檔config.json，如下所示：

```
{
"AccessKeyID" : "",
"AccessKeySecret" : "",
"RoleArn" : "",
"TokenExpireTime" : "900",
"PolicyFile": "policy/all_policy.txt"
}
```

相關參數說明如下：

- AccessKeyID：填寫之前記錄的AccessKeyID。
- AccessKeySecret：填寫之前記錄的AccessKeySecret。
- RoleArn：填寫之前記錄的RoleArn。
- TokenExpireTime：指Android/iOS應用獲取到的Token的失效時間，最小值和預設值均為900s。
- PolicyFile：該Token所擁有的許可權列表的檔案，可使用預設值。

> **說明**
>
> 程式碼範例中提供了三種最常用的Token許可權檔案，位於policy目錄下面。分別是：
>
> - all_policy.txt：指定該Token擁有該帳號下的所有許可權，包括：
>   - 建立Bucket
>   - 刪除Bucket
>   - 上傳檔案
>   - 下載檔案
>   - 刪除檔案
> - bucket_read_policy.txt：指定該Token擁有該帳號下指定Bucket的讀許可權。
> - bucket_read_write_policy.txt：指定了該Token擁有該帳號下指定Bucket的讀寫權限。
>
> 如果您想要指定該Token只對指定的Bucket有讀寫權限， 請把bucket_read_policy.txt和bucket_read_write_policy.txt檔案裡的$BUCKET_NAME替換成指定的Bucket名稱。

步驟4：運行範例程式碼

程式碼範例的運行方法如下：

- 對於PHP版本，將包下載解壓後，修改config.json檔案，直接運行php sts.php即能生成Token，將程式部署到指定的應用伺服器位址。
- 對於Java版本（依賴於java 1.7），將包下載解壓後，修改config.json檔案，運行java -jar oss-token-server.jar (port)。如果不指定port（通信埠），直接運行java -jar oss-token-server.jar，程式會監聽7080通信埠。

返回的資料格式解析如下：

```
//正確返回
{
    "StatusCode":200,
    "AccessKeyId":"STS.3p***dgagdasdg",
    "AccessKeySecret":"rpnwO9***tGdrddgsR2YrTtI",
   "SecurityToken":"CAES+wMIARKAAZhjH0EUOIhJMQBMjRywXq7MQ/cjLYg80Aho1ek0Jm63XMhr9Oc5s'∂'∂3q
aPer8p1YaX1NTDiCFZWFkvlHf1pQhuxfKBc+mRR9KAbHUefqH+rdjZqjTF7p2m1wJXP8S6k+G2MpHrUe6TYBkJ43Ghh
TVFMuM3BZajY3VjZWOXBIODRIR1FKZjIiEjMzMzE0MjY0NzM5MTE4NjkxMSoLY2xpZGSSDgSDGAGESGTETqOio6c2Rr
LWRlbW8vKgoUYWNzOm9zczoqOio6c2RrLWRlbW9KEDExNDg5MzAxMDcyNDY4MThSBTI2ODQyWg9Bc3N1bWVkUm9sZVV
zZXJgAGoSMzMzMTQyNjQ3MzkxMTg2OTExcglzZGstZGVtbzI=",
   "Expiration":"2017-12-12T07:49:09Z",
}
//錯誤返回
{
    "StatusCode":500,
    "ErrorCode":"InvalidAccessKeyId.NotFound",
    "ErrorMessage":"Specified access key is not found."
}
```

正確返回的五個變數構成一個Token：

- StatusCode：獲取Token的狀態，獲取成功時，傳回值是200。
- AccessKeyId：Android/iOS行動裝置 App初始化OSSClient獲取的 AccessKeyId。
- AccessKeySecret：Android/iOS行動裝置 App初始化OSSClient獲取AccessKeySecret。
- SecurityToken：Android/iOS行動裝置 App初始化的Token。
- Expiration：該Token失效的時間。Android SDK會自動判斷Token是否失效，如果失效，則自動獲取Token。

錯誤返回說明如下：

- StatusCode：表示獲取Token的狀態，獲取失敗時，傳回值是500。
- ErrorCode：表示錯誤原因。
- ErrorMessage：表示錯誤的具體資訊描述。

步驟5：體驗行動裝置 App直傳服務

1. 部署程式後，記下應用伺服器位址如 `http://abc.com:8080` ，將樣本程式裡面的應用伺服器修改成上述地址。
2. 選擇數據要上傳到哪個Bcuket及地域，修改樣本程式裡相應的上傳Bucket及區域。
3. 單擊**設定**，載入配置。

4. 選擇圖片，設定上傳OSS檔案名，上傳圖片。
5. 上傳成功後，通過控制台查看上傳結果。

核心代碼解析

OSS初始化的代碼解析如下：

- Android版本

```
// 推薦使用OSSAuthCredentialsProvider，token過期後會自動刷新。
String stsServer = "應用伺服器位址，例如http://abc.com:8080"
OSSCredentialProvider credentialProvider = new OSSAuthCredentialsProvider(stsServer);
//config
ClientConfiguration conf = new ClientConfiguration();
conf.setConnectionTimeout(15 * 1000); // 連接逾時時間，預設15秒
conf.setSocketTimeout(15 * 1000); // Socket逾時時間，預設15秒
conf.setMaxConcurrentRequest(5); // 最大並發請求數，預設5個
conf.setMaxErrorRetry(2); // 失敗後最大重試次數，預設2次
OSS oss = new OSSClient(getApplicationContext(), endpoint, credentialProvider, conf);
```

- iOS版本

```
OSSClient * client;
...
// 推薦使用OSSAuthCredentialProvider，token過期後會自動刷新。
id<OSSCredentialProvider> credential = [[OSSAuthCredentialProvider alloc] initWithAuthSer
verUrl:@"應用伺服器位址，例如http://abc.com:8080"];
client = [[OSSClient alloc] initWithEndpoint:endPoint credentialProvider:credential];
```

1.2.2. 快速搭建行動裝置 App上傳回調服務

背景

快速搭建行動裝置 App直傳服務介紹了如何在30分鐘內中搭建一個基於OSS的行動裝置 App數據直傳服務。移動端開發場景流程圖如下：

角色分析如下所示：

- 應用伺服器負責為Android/iOS行動裝置 App生成STS憑證。
- Android/iOS行動裝置 App負責從應用伺服器申請及使用STS憑證。
- OSS負責處理行動裝置 App的數據請求。

對於Android/iOS移動應來說，行動裝置 App只需要執行上圖中操作1（申請STS憑證），就能調用多次5(使用該STS憑證上傳數據到OSS)，導致應用伺服器根本不知道用戶都上傳了哪些數據，作為該APP的開發人員，就沒法對應用上傳數據進行管理。有什麼問題能讓應用伺服器感知到Android/iOS行動裝置 App上傳的數據呢？

您可以通過使用OSS的上傳回調服務，就能解決上述問題，如下圖所示：

OSS在收到Android/iOS移動的數據（上圖中操作5）和在返回用戶上傳結果（上圖中操作6）之間，觸發一個上傳回調工作。即第上圖中操作5.5，先回調使用者服務器，得到應用伺服器返回的內容，將這個內容返回給Android/iOS行動裝置 App。可以參考Callback API文檔。

上傳回調的作用

- 通過上傳回調可以讓用戶應用伺服器知道當前上傳檔案的基本資料。

  基本資料如下表。返回下述變數的一個或者多個，返回內容格式形式在Android/iOS上傳時指定。

系統變數	含義
bucket	行動裝置 App上傳到哪個儲存空間
object	行動裝置 App上傳到OSS保存的檔案名
etag	該上傳的檔案的etag，即返回給用戶的etag欄位
size	該上傳的檔案的大小
mimeType	資源類型
imageInfo.height	圖片高度
imageInfo.width	圖片寬度
imageInfo.format	圖片格式，如jpg、png，只以識別圖片

- 通過上傳回調設定自訂參數，達到資訊傳遞目的。

  假如您是一個開發人員，您想知道目前使用者所使用的APP版本、目前使用者所在的作業系統版本、用戶的GPS資訊、用戶的手機型號。您可以在Android/iOS端上傳檔案時，指定上述自訂參數，如下所示：

  - x:version指定APP版本
  - x:system指定作業系統版本
  - x:gps指定GPS資訊
  - x:phone指定手機型號

    上述這些值會在Android/iOS行動裝置 App上傳到OSS時附帶上，OSS會把這些值放到CallbackBody裡面一起發給應用伺服器。這樣應用伺服器就能收到這些資訊，達到資訊傳遞的目的。

在行動裝置 App端設定上傳回調

要讓OSS在接收上傳請求時，觸發上傳回調，行動裝置 App在構造上傳請求時必須把如下兩個內容指定到上傳請求裡面：

- 要回調到哪個伺服器callbackUrl，如 `http://abc.com/callback.php` ，這個地址必須是公網能夠訪問的。
- 上傳回調給應用伺服器的內容callbackBody，可以是上述OSS返回應用伺服器系統變數的一個或者多個。

假如您的使用者服務器上傳回調地是 `http://abc.com/callback.php` 。您想獲取手機上傳的檔案名字、檔案的大小，並且定義了photo變數是指手機型號，system是指作業系統版本。

上傳回調樣本分以下兩種：

- iOS指定上傳回調樣本：

```
OSSPutObjectRequest * request = [OSSPutObjectRequest new];
request.bucketName = @"<bucketName>";
request.objectKey = @"<objectKey>";
request.uploadingFileURL = [NSURL fileURLWithPath:@<filepath>"];
// 設定回調參數
request.callbackParam = @{
                          @"callbackUrl": @"http://abc.com/callback.php",
                          @"callbackBody": @"filename=${object}&size=${size}&photo=${x:ph
oto}&system=${x:system}"
                          };
// 設定自訂變數
request.callbackVar = @{
                        @"x:photo": @"iphone6s",
                        @"x:system": @"ios9.1"
                        };
```

- Android指定上傳回調樣本：

```
PutObjectRequest put = new PutObjectRequest(testBucket, testObject, uploadFilePath);
ObjectMetadata metadata = new ObjectMetadata();
metadata.setContentType("application/octet-stream");
put.setMetadata(metadata);
put.setCallbackParam(new HashMap<String, String>() {
    {
        put("callbackUrl", "http://abc.com/callback.php");
        put("callbackBody", "filename=${object}&size=${size}&photo=${x:photo}&system=${x:
system}");
    }
});
put.setCallbackVars(new HashMap<String, String>() {
     {
         put("x:photo", "IPOHE6S");
         put("x:system", "YunOS5.0");
     }
});
```

上傳回調對應用伺服器的要求

- 您必須部署一個可以接收POST請求的服務，這個服務必須有公網地址
  如 `www.abc.com/callback.php` （或者外網IP也可以），不然OSS沒有辦法訪問到這個地址。
- 您要給OSS正確的返回，返回格式必須是JSON格式，內容自訂。因為OSS會把應用伺服器返回的內容，原封不動地返回給Android/iOS行動裝置 App。（切記，返回給OSS的Response Header一定要加上Content-Length這個頭部）。

本教程在最後為大家準備了多個語言版本的樣本、下載及運行方法。

應用伺服器收到的回調請求

應用伺服器收到OSS的請求，抓包的請求如下（這個結果根據設定的不同URL和回調內容會有不同）：

```
POST /index.html HTTP/1.0
Host: 121.43.113.8
Connection: close
Content-Length: 81
Content-Type: application/x-www-form-urlencoded
User-Agent: ehttp-client/0.0.1
authorization: kKQeGTRccDKyHB3H9vF+xYMSrmhMZjzzl2/kdD1ktNVgbWEfYTQG0G2SU/RaHBovRCE8OkQDjC3u
G33esH2txA==
x-oss-pub-key-url: aHR0cDovL2dvc3NwdWJsaWMuYWxpY2RuLmNvbS9jYWxsYmFja19wdWJfa2V5X3YxLnBlbQ==
filename=test.txt&size=5&photo=iphone6s&system=ios9.1
```

更多內容請參考Callback API文檔。

應用伺服器判斷回調請求是否來自OSS

如果您的回調伺服器被人惡意攻擊了，例如惡意回調您的應用伺服器，導致應用伺服器收到一些非法的請求，影響正常邏輯，此時您就需要判斷回調請求是否來自OSS。

判斷的方法主要是利用OSS給應用伺服器返回的頭部內容中，`x-oss-pub-key-url`和`authorization`這兩個參數進行RSA校驗。只有通過RSA校驗才能說明這個請求是來自OSS，本教程提供的樣本程式都有實現的樣本供您參考。

應用伺服器收到回調請求後的處理

應用伺服器在校驗這個請求是來自OSS後，指定回調給應用伺服器的內容格式，如

```
filename=test.txt&size=5&photo=iphone6s&system=ios9.1
```

應用伺服器就可以根據OSS的返回內容，解析得到自己想要得到的數據。得到這個數據後，應用伺服器可以把數據存放起來，方便後續管理。

應用伺服器收到回調請求後如何返回給OSS

- 返回狀態碼是200；
- 返回必須是json格式的內容；
- 返回的頭部必須帶有Content-Length這個頭部。

OSS如何處理應用伺服器的返回內容

有兩種情況：

- OSS將回調請求發送給應用伺服器，但是應用伺服器接收失敗或者訪問不通，OSS會返回給Android/iOS行動裝置 App203的狀態碼，但是數據已經存放到OSS上了。
- 應用伺服器接收到OSS的回調請求，並且正確返回了，OSS會返回給Android/iOS行動裝置 App狀態碼是200, 並把應用伺服器給OSS的內容，原封不動地返回給Android/iOS行動裝置 App。

上傳回調伺服器樣本程式下載

樣本程式只是完成了如何檢查應用伺服器收到的簽名， 用戶要自行增加對應用伺服器收到回調的內容的格式解析 。

- Java版本：
  - 下載地址：單擊這裡。

- 運行方法，解壓包運行 `java -jar oss-callback-server-demo.jar 9000` （9000是啟動並執行通信埠，可以自己指定）。

  > **說明** 這個jar例子在java 1.7運行通過，如果有問題可以自己依據提供的代碼進行修改。這是一個maven項目。

- PHP版本：
  - 下載地址：單擊這裡
  - 運行方法：部署到Apache環境下，因為PHP本身語言的特點，取一些數據頭部會依賴於環境。所以可以參考例子根據自己所在環境進行修改。
- Python版本：
  - 下載地址：單擊這裡。
  - 運行方法：解壓包直接運行python callback_app_server.py即可，程式自實現了一個簡單的http server，運行該程式可能需要安裝rsa的依賴。
- Ruby版本：
  - 下載地址：單擊這裡。
  - 運行方法： ruby aliyun_oss_callback_server.rb。

1.3. 通過crc64校驗資料轉送的完整性

背景

數據在客戶端和伺服器之間傳輸時有可能會出錯。OSS現在支援對各種方式上傳的Object返回其crc64值，客戶端可以和本地計算的crc64值做對比，從而完成資料完整性的驗證。

- OSS對新上傳的Object進行crc64的計算，並將結果儲存為Object的元資訊儲存，隨後在返回的response header中增加 `x-oss-hash-crc64ecma` 頭部，表示其crc64值，該64位CRC根據ECMA-182標準計算得出。
- 對於crc64上線之前就已經存在於OSS上的Object，OSS不會對其計算crc64值，所以獲取此類Object時不會返回其crc64值。

操作說明

- Put Object / Append Object / Post Object / Multipart upload part 均會返回對應的crc64值，客戶端可以在上傳完成後拿到伺服器返回的crc64值和本地計算的數值進行校驗。
- Multipart Complete時，如果所有的Part都有crc64值，則會返回整個Object的crc64值；否則，比如有crc64上線之前就已經上傳的Part，則不返回crc64值。
- Get Object / Head Object / Get ObjectMeta 都會返回對應的crc64值（如有）。客戶端可以在Get Object完成後，拿到伺服器返回的crc64值和本地計算的數值進行校驗。

  > **說明** range get請求返回的將會是整個Object的crc64值。

- Copy相關的操作，如Copy Object / Upload Part Copy，新生成的Object/Part不保證具有crc64值。

應用樣本

以下為完整的Python範例程式碼，示範如何基於crc64值驗證資料轉送的完整性。

1. 計算crc64。

```
import oss2
from oss2.models import PartInfo
import os
import crcmod
import random
import string
do_crc64 = crcmod.mkCrcFun(0x142F0E1EBA9EA3693L, initCrc=0L, xorOut=0xffffffffffffffffL
, rev=True)
def check_crc64(local_crc64, oss_crc64, msg="check crc64"):
if local_crc64 != oss_crc64:
print "{0} check crc64 failed. local:{1}, oss:{2}.".format(msg, local_crc64, oss_crc64)
return False
else:
print "{0} check crc64 ok.".format(msg)
return True
def random_string(length):
return ''.join(random.choice(string.lowercase) for i in range(length))
bucket = oss2.Bucket(oss2.Auth(access_key_id, access_key_secret), endpoint, bucket_name
)
```

2. 驗證Put Object。

```
content = random_string(1024)
 key = 'normal-key'
 result = bucket.put_object(key, content)
 oss_crc64 = result.headers.get('x-oss-hash-crc64ecma', '')
 local_crc64 = str(do_crc64(content))
 check_crc64(local_crc64, oss_crc64, "put object")
```

3. 驗證Get Object。

```
result = bucket.get_object(key)
 oss_crc64 = result.headers.get('x-oss-hash-crc64ecma', '')
 local_crc64 = str(do_crc64(result.resp.read()))
 check_crc64(local_crc64, oss_crc64, "get object")
```

4. 驗證Upload Part 和 Complete。

```
part_info_list = []
 key = "multipart-key"
 result = bucket.init_multipart_upload(key)
 upload_id = result.upload_id
 part_1 = random_string(1024 * 1024)
 result = bucket.upload_part(key, upload_id, 1, part_1)
 oss_crc64 = result.headers.get('x-oss-hash-crc64ecma', '')
 local_crc64 = str(do_crc64(part_1))
 #check 上傳的 part 1數據是否完整
 check_crc64(local_crc64, oss_crc64, "upload_part object 1")
 part_info_list.append(PartInfo(1, result.etag, len(part_1)))
 part_2 = random_string(1024 * 1024)
 result = bucket.upload_part(key, upload_id, 2, part_2)
 oss_crc64 = result.headers.get('x-oss-hash-crc64ecma', '')
 local_crc64 = str(do_crc64(part_2))
 #check 上傳的 part 2數據是否完整
 check_crc64(local_crc64, oss_crc64, "upload_part object 2")
 part_info_list.append(PartInfo(2, result.etag, len(part_2)))
 result = bucket.complete_multipart_upload(key, upload_id, part_info_list)
 oss_crc64 = result.headers.get('x-oss-hash-crc64ecma', '')
 local_crc64 = str(do_crc64(part_2, do_crc64(part_1)))
 #check 最終oss上的object和本地檔案是否一致
 check_crc64(local_crc64, oss_crc64, "complete object")
```

OSS SDK支援

部分OSS SDK已經支援上傳、下載使用crc64進行數據校驗，用法見下表中的樣本：

SDK	是否支援CRC	樣本
Java SDK	是	CRCSample.java
Python SDK	是	object_check.py
PHP SDK	否	無
C# SDK	否	無
C SDK	是	oss_crc_sample.c
JavaScript SDK	否	無
Go SDK	是	crc_test.go
Ruby SDK	否	無
iOS SDK	否	OSSCrc64Tests.m
Android SDK	否	CRC64Test.java

2.資料湖與資料分析

2.1. 資料處理與分析

2.1.1. 基於OSS+MaxCompute構建資料倉儲

背景

本文重點介紹基於OSS並使用MaxCompute構建PB資料倉儲的方法。

- Object Storage Service

  Object Storage Service提供標準、低頻、Archive Storage類型，能夠覆蓋從熱到冷的不同儲存場景。同時，OSS能夠與Hadoop開源社區及EMR、批次運算、MaxCompute、機器學習服務PAI、DatalakeAnalytics、Function Compute等阿里雲計算產品進行深度結合。

  用戶可以打造基於OSS的資料分析應用，如MapReduce、HIVE/Pig/Spark等批處理（如日誌離線計算）、互動式查詢分析（Imapla、Presto、DataLakeAnalytics）、深度學習訓練（阿里雲PAI）、基因渲染計算交付（批次運算）、大數據應用（MaxCompute）及串流（Function Compute）等。

- MaxCompute

  MaxCompute是一項大數據計算服務，能夠提供快速且完全託管的資料倉儲解決方案，並可以與OSS結合，高效並經濟地分析處理海量數據。MaxCompute的處理性能達到了全球領先水平，被Forrester評為全球雲端資料倉儲領導者。

- OSS外部表格查詢功能

  MaxCompute重磅推出了一項重要特性：OSS外表查詢功能。該功能可以幫助您直接對OSS中的海量檔案進行查詢，而不必將數據載入到MaxCompute表中，既節約了數據搬遷的時間和人力，也節省了多地儲存的成本。

  使用MaxCompute+OSS的方案，您可以獲得以下優勢：

  - MaxCompute是一個無伺服器的分散式運算架構，無需用戶對伺服器基礎設施進行額外的維護和管理，能夠根據OSS用戶的需求方便及時地提供臨時查詢服務，幫助企業大幅節省成本。
  - OSS為海量的對象儲存服務。用戶將數據統一儲存在OSS，可以做到計算和儲存分離，多種計算應用和業務都可以訪問使用OSS的數據，數據只需要儲存一份。
  - 支援處理OSS上開源格式的結構化檔案，包括：Avro、CSV、ORC、Parquet、RCFile、RegexSerDe、SequenceFile和TextFile，同時支援gzip壓縮格式。

典型應用場景

互連網金融應用每天都需要將大量的金融數據分頁檔存放在OSS上，並需要進行超大文字檔的結構化分析。通過MaxCompute的OSS外部表格查詢功能，用戶可以直接用外部表格的方式將OSS上的大檔案載入到MaxCompute進行分析，從而大幅提升整個鏈路的效率。

操作步驟

下面我們通過兩個簡單的樣本，介紹如何通過MaxCompute外表功能實現對OSS數據的分析和處理。

- 場景一：物聯網採集資料分析
  i. 步驟1：準備工作
     a. 開通OSS和MaxCompute服務。

您可以通過官網分別開通OSS、MaxCompute服務，並建立OSS bucket、MaxCompute Project。

b. 採集數據到OSS。

您可以使用任何資料集來執行測試，以驗證我們在這篇文章中概述的最佳實踐。本樣本在OSS上準備了一批 CSV 數據，endpoint為oss-cn-beijing-internal.aliyuncs.com，bucket為oss-odps-test，資料檔案的存放路徑為 **/demo/vehicle.csv**。

c. 授權MaxCompute訪問OSS。

MaxCompute需要直接存取OSS的數據，因此需要將OSS的數據相關許可權賦給MaxCompute的訪問帳號。您可以在直接登入阿里雲帳號後，點擊完成授權。

ii. 步驟2：通過MaxCompute建立外部表格

建立外部表格，語句如下：

```
CREATE EXTERNAL TABLE IF NOT EXISTS ambulance_data_csv_external
(
    vehicleId int,
    recordId int,
    patientId int,
    calls int,
    locationLatitute double,
    locationLongtitue double,
    recordTime string,
    direction string
    )
    STORED BY 'com.aliyun.odps.CsvStorageHandler'
    LOCATION 'oss://oss-cn-beijing-internal.aliyuncs.com/oss-odps-test/Demo/';
```

iii. 步驟3：通過MaxCompute查詢外部表格

成功建立外部表格後，便可如普通表一樣使用該外部表格。

假設 **/demo/vehicle.csv** 的數據如下：

```
1,1,51,1,46.81006,-92.08174,9/14/2014 0:00,S
1,2,13,1,46.81006,-92.08174,9/14/2014 0:00,NE
1,3,48,1,46.81006,-92.08174,9/14/2014 0:00,NE
1,4,30,1,46.81006,-92.08174,9/14/2014 0:00,W
1,5,47,1,46.81006,-92.08174,9/14/2014 0:00,S
1,6,9,1,46.81006,-92.08174,9/14/2014 0:00,S
1,7,53,1,46.81006,-92.08174,9/14/2014 0:00,N
1,8,63,1,46.81006,-92.08174,9/14/2014 0:00,SW
1,9,4,1,46.81006,-92.08174,9/14/2014 0:00,NE
1,10,31,1,46.81006,-92.08174,9/14/2014 0:00,N
```

執行如下 SQL 語句：

```
select recordId, patientId, direction from ambulance_data_csv_external where patientI
d > 25;
```

輸出結果如下：

```
+------------+------------+-----------+
| recordId   | patientId  | direction |
+------------+------------+-----------+
| 1          | 51         | S         |
| 3          | 48         | NE        |
| 4          | 30         | W         |
| 5          | 47         | S         |
| 7          | 53         | N         |
| 8          | 63         | SW        |
| 10         | 31         | N         |
+------------+------------+-----------+
```

- 場景二：阿里雲產品消費賬單分析
  i. 步驟1：準備工作
     a. 開通OSS和MaxCompute服務。

        您可以通過官網分別開通OSS和MaxCompute服務，並建立OSS bucket、MaxCompute Project。

     b. 授權MaxCompute訪問OSS。

        MaxCompute需要直接存取OSS的數據，因此需要將OSS的數據相關許可權賦給MaxCompute的訪問帳號。您可以在直接登入阿里雲帳號後，點擊完成授權。

  ii. 步驟2：通過費用中心同步賬單數據到OSS
     a. 服務開通後，每天會將增量的實例消費詳細資料組建檔案，並同步儲存到指定的bucket中，如下圖所示：

  iii. 步驟3：通過MaxCompute註冊賬單處理類
     a. 點擊以下連結下載自訂代碼：odps-udf-example-0.30.0-SNAPSHOT-jar-with-dependencies.jar
     b. 運行以下命令，將自訂代碼編譯打包，並上傳到 MaxCompute。

        ```
        add jar odps-udf-example-0.30.0-SNAPSHOT-jar-with-dependencies.jar
        ```

  iv. 步驟4：通過MaxCompute建立外部表格

     以建立5月4日的賬單消費表為例，外部表格如下：

```
CREATE EXTERNAL TABLE IF NOT EXISTS oms_oss_0504
(
月份 string,
資源擁有者 string,
消費時間 string,
消費類型 string,
賬單編號 string,
商品 string,
計費方式 string,
服務開始時間 string,
服務結束時間 string,
服務時長 string,
財務核算單元 string,
資源id string,
資源暱稱 string,
TAG string,
```

```
地域 string,
可用區 string,
公網ip string,
內網ip string,
資源配置 string,
原價 string,
優惠金額 string,
應付金額 string,
計費項1 string,
使用量1 string,
資源套件扣除1 string,
原價1 string ,
應付金額1 string,
計費項2 string,
使用量2 string,
資源套件扣除2 string,
原價2 string,
應付金額2 string,
計費項3 string,
使用量3 string,
資源套件扣除3 string,
原價3 string,
應付金額3 string,
計費項4 string,
使用量4 string,
資源套件扣除4 string,
原價4 string,
應付金額4 string,
計費項5 string,
使用量5 string,
資源套件扣除5 string,
原價5 string,
應付金額5 string,
計費項6 string,
使用量6 string,
資源套件扣除6 string,
原價6 string,
應付金額6 string,
計費項7 string,
使用量7 string,
資源套件扣除7 string,
原價7 string,
應付金額7 string,
計費項8 string,
使用量8 string,
資源套件扣除8 string,
原價8 string,
應付金額8 string,
計費項9 string,
使用量9 string,
資源套件扣除9 string,
原價9 string,
應付金額9 string
)
STORED BY 'com.aliyun.odps.udf.example.text.TextStorageHandler' -- STORED BY 指定自訂 S
```

```
STORED BY 'com.aliyun.odps.udf.example.text.TextStorageHandler' --STORED BY 指定自訂 S
torageHandler 的類名。
with SERDEPROPERTIES (
'odps.text.option.complex.text.enabled'='true',
'odps.text.option.strict.mode'='false'
--遇到列數不一致的情況不會拋異常，如果實際列數少於schema列數，將所有列按順序匹配，剩下的不足的列
補NULL。
)
LOCATION 'oss://oss-cn-beijing-internal.aliyuncs.com/oms-yl/2018-05-04/'
USING 'text_oss.jar'; --同時需要指定賬單中的文本處理類定義所在的 jar 包。
```

v. 步驟5：通過MaxCompute查詢外部表格

以查詢ECS快照消費賬單明細為例，命令如下：

```
select  月份,原價,優惠金額,應付金額,計費項1,使用量 from oms_oss where  商品=快照(SNAPSHOT);
```

總結

上述樣本說明了如何通過MaxCompute對OSS上的海量數據進行分析，將您的大資料分析工作效率提升至分鐘級，幫助您更高效、更低成本的挖掘海量數據價值。

3.資料備份

3.1. 備份儲存空間

針對存放在OSS上的數據，阿里雲提供多種資料備份方式，以滿足不同場景的備份需求。

OSS雲上備份方式有如下2種：

- 跨區域複製（提供控制台配置操作以及基於API/SDK兩種模式）
- 基於OssImport工具

通過跨區域複製進行備份

使用場景

參見管理跨區域複製開發指南。

控制台設定作業

參見設定跨區域複製操作指南。

常見問題

參見Bucket之間資料同步常見問題。

特別說明

- 源Bucket和目標Bucket屬於同一個用戶，且分屬不同的區域。
- 源Bucket、目標Bucket儲存類型都不是歸檔類型。
- 若要在同一區域的Bucket之間進行資料同步，則可以通過OSS的SDK/API編碼實現。

通過使用OssImport工具進行備份

OssImport工具可以將本地、其它雲端儲存的資料移轉到OSS，它有以下特點：

- 支援豐富的資料來源，有本地、七牛、百度BOS、AWS S3、Azure Blob、又拍雲、騰訊雲COS、金山KS3、HTTP、OSS等，並可根據需要擴充。
- 支援斷點續傳。
- 支援流量控制。
- 支援遷移指定時間後的檔案、特定首碼的檔案。
- 支援並行數據下載、上傳。
- 支援單機模式和分布式模式。單機模式部署簡單使用方便，分布式模式適合大規模資料移轉

使用場景

參見資料移轉。

安裝部署

參見說明及配置 、單機部署、分布式部署。

常見問題

參見常見問題。

特別說明

- 不同賬戶的不同Bucket之間，數據量很大，10TB等級以上的情況，建議使用分布式的版本。
- 利用增量模式在OSS之間同步檔案修改操作，OssImport只能同步檔案的修改操作

(put/apppend/multipart)，不能同步讀取和刪除操作，資料同步的及時性沒有具體的 SLA 保證， 慎重選擇。

- 如果開通了跨區域複製的地域，則推薦使用跨區域複製進行地域之間的資料同步。

4.成本管理

5.資料移轉

5.1. Amazon S3資料移轉到OSS

OSS提供了S3 API的相容性，可以讓您的數據從Amazon S3無縫遷移到阿里雲OSS上。從Amazon S3遷移到OSS後，您仍然可以使用S3 API訪問OSS，僅需要對S3的客戶端應用進行如下改動：

1. 獲取OSS主帳號或子帳號的AccessKeyId和AccessKeySecret，並在您使用的客戶端和SDK中配置您申請的AccessKeyId與AccessKeySecret。
2. 設定客戶端串連的endpoint為OSS endpoint。OSS endpoint列表請參見訪問網域名稱和資料中心。

遷移步驟

從第三方儲存遷移到OSS的具體操作步驟，請您參見使用OssImport遷移數據。

遷移後用S3 API訪問OSS

從S3遷移到OSS後，您使用S3 API訪問OSS時，需要注意以下幾點：

- Path style和Virtual hosted style

  Virtual hosted style是指將Bucket放入host header的訪問方式。OSS基於安全考慮，僅支援virtual hosted訪問方式，所以在S3到OSS遷移後，客戶端應用需要進行相應設定。部分S3工具預設使用Path style，也需要進行相應配置，否則可能導致OSS報錯禁止訪問。

- OSS對許可權的定義與S3不完全一致，遷移後如有需要，可對許可權進行相應調整。二者的主要區別可參考下表。

  > 說明
  > - 更詳細的區別請參見ACL驗證流程。
  > - OSS僅支援S3中的私有、公共讀和公共讀寫三種ACL模式。

對象	Amazon S3許可權	Amazon S3	阿里雲OSS
Bucket	READ	擁有Bucket的list許可權	對於Bucket下的所有Object，如果某Object沒有設定Object許可權，則該Object可讀
	WRITE	Bucket下的Object可寫入或覆蓋	○ 對於Bucket下不存在的Object，可寫入 ○ 對於Bucket下存在的Object，如果該Object沒有設定Object許可權，則該Object可被覆蓋 ○ 可以initiate multipart upload
	READ_ACP	讀取Bucket ACL	讀取Bucket ACL，僅Bucket owner和授權子帳號擁有此許可權

對象	Amazon S3許可權	Amazon S3	阿里雲OSS
	WRITE_ACP	設定Bucket ACL	設定Bucket ACL，僅Bucket owner和授權子帳號擁有此許可權
Object	READ	Object可讀	Object可讀
	WRITE	N/A	Object可以被覆蓋
	READ_ACP	讀取Object ACL	讀取Object ACL，僅Bucket owner和授權子帳號擁有此許可權
	WRITE_ACP	設定Object ACL	設定Object ACL，僅Bucket owner和授權子帳號擁有此許可權

- 儲存類型

  OSS支援標準（Standard）、低頻訪問（IA）和Archive Storage（Archive）三種儲存類型，分別對應Amazon S3中的STANDARD、STANDARD_IA和GLACIER。

  與Amazon S3不同的是，OSS不支援在上傳object時直接指定儲存類型，object的儲存類型由bucket的儲存類型指定。OSS支援標準、低頻訪問和歸檔三種Bucket類型，Object的儲存類型還可以通過lifecycle進行轉換。

  Archive Storage類型的object在讀取之前，要先使用Restore請求進行解凍操作。與S3不同，OSS會忽略S3 API中的解凍天數設定，解凍狀態預設持續1天，用戶可以延長到最多7天，之後，Object又回到初始時的冷凍狀態。

- ETag
  - 對於PUT方式上傳的Object，OSS Object的ETag與Amazon S3在大小寫上有區別。OSS為大寫，而S3為小寫。如果客戶端有關於ETag的校驗，請忽略大小寫。
  - 對於分區上傳的Object，OSS的ETag計算方式與S3不同。

相容的S3 API

- Bucket操作：
  - Delete Bucket
  - Get Bucket （list objects）
  - Get Bucket ACL
  - Get Bucket lifecycle
  - Get Bucket location
  - Get Bucket logging
  - Head Bucket
  - Put Bucket
  - Put Bucket ACL
  - Put Bucket lifecycle
  - Put Bucket logging

- Object操作：
  - Delete Object
  - Delete Objects
  - Get Object
  - Get Object ACL
  - Head Object
  - Post Object
  - Put Object
  - Put Object Copy
  - Put Object ACL
- Multipart操作：
  - Abort Multipart Upload
  - Complete Multipart Upload
  - Initiate Multipart Upload
  - List Parts
  - Upload Part
  - Upload Part Copy

5.2. 使用OssImport遷移數據

本文向您介紹如何使用OssImport將數據從第三方儲存（或OSS）遷移到OSS。

工具選擇：單機模式和分布式模式

OssImport有單機模式和分布式模式兩種部署方式。一般建議使用分布式模式。您參考OssImport官網指導文檔，即可完成遷移過程。本文介紹您在整體遷移方案中可能會關注的內容，以及可以參考的官網文檔資源。

遷移方案（從第三方儲存遷移到OSS）

以下步驟可以完成從其它儲存到OSS的無縫切換（參考官網支援的第三方儲存類型OssImport 架構及配置）：

具體步驟如下：

1. 全量遷移T1前的曆史數據，請參考：OssImport 架構及配置。
   - 記錄遷移開始時間T1（注意為Unix時間戳記，即自1970年1月1日UTC零點以來的秒數，通過命令 `date +%s` 獲取）。
   - 遷移指導說明參考OssImport官網文檔，請參考遷移工具-分布式。
2. 開啟OSS鏡像回源，並將讀寫切換到OSS，遷移源不再新增數據。
   - 步驟1遷移完成後，在OSS控制台開啟OSS鏡像回源功能，回源地址為遷移源（第三方儲存）。
   - 在業務系統讀寫切換到OSS，假設業務系統修改好的時間為T2。
   - 此時T1前的數據從OSS讀取，T1後的數據，OSS利用鏡像回源從第三方服務讀取，而新數據完全寫入OSS。
3. 快速遷移T1~T2到數據。
   - 在步驟2完成後，第三方儲存不會再新增數據，數據讀寫已切到OSS。

- 修改設定檔 `job.cfg` 的配置項 `importSince=T1` ，新發起遷移job，遷移T1~T2數據。

4. 步驟3完成後，即完成遷移全過程。
   - 步驟3完成後，您業務的所有的讀寫都在OSS上，第三方儲存只是一份曆史數據，您可以根據需要決定保留或刪除。
   - OssImport負責數據的遷移和校驗，不會刪除任何數據。

遷移成本

遷移過程涉及到成本一般有ECS費用、流量費用、儲存費用、時間成本。其中，多數情況下，比如數據超過TB等級，儲存成本和遷移時間成正比，而ECS費用相對流量、儲存費用較小。

環境準備

假設您有如下遷移需求：

需求項	需求情況
遷移源	AWS S3東京
遷移目的	OSS香港
數據量	500TB
遷移時間要求	1周內完成遷移

您需要準備的環境：

環境項	說明
開通OSS	開通OSS步驟如下： 1. 使用您的帳號建立香港地域的OSS Bucket。 2. 在RAM控制台建立子帳號，並授權該子帳號可訪問OSS。保存AccessKeyID和AccessKeySecret。
購買ECS	購買OSS同區域(香港)的ECS，一般普通的2核4G機型即可，如果遷移後ECS需釋放，建議按需購買ECS。正式遷移時的ECS數量，請參考關於遷移用的ECS數量。
配置OssImport	**說明** destDomain參數，即OSS目的endpoint，建議設定為OSS內網的endpoint，避免從ECS數據上傳到OSS時產生外網流量費用。具體的endpoint，請參考訪問網域名稱和資料中心。
遷移過程	1. 在ECS上搭建OssImport分布式環境。 2. 使用OssImport從東京AWS S3下載數據到ECS（香港），建議使用外網。 3. 使用OssImport從ECS（香港）將數據上傳到OSS（香港），建議使用內網。

關於遷移用的ECS數量

您需根據遷移需求，計算您需要用來做遷移的ECS數量：

- 假設您需要遷移的數據量是X TB，要求遷移完成時間Y天，單台遷移速度Z Mbps（每天遷移約Z/100 TB數據）。
- 則正式遷移時，ECS大致需要X/Y/(Z/100)台。

假設單台ECS遷移速度達到200Mbps(即每天約遷移2TB數據)。則上面Case中，ECS約需36台（500/7/2）。

OssImport遷移步驟

配置參考

您可以閱讀官網指導文檔，了解配置定義OssImport 架構及配置、操作步驟分布式部署，在開始前請關注如下資訊：

- OssImport下載：在Master下載OssImport分布式版，且master、workers都使用同樣的ssh帳號、密碼。（worker上不用單獨下載OssImport，運行 `deploy` 命令時，OssImport會分發到worker上，請參考分布式部署。）
- Java環境：確認Master和worker都已安裝。

  > **說明** 作為worker的ECS也需要安裝Java。

- 設定workdir：通過 `conf/sys.properties` 的配置項 `workingDir` 指定，請參考分布式部署。

  > **說明** workdir的設定請參考官網文檔，不要設定成OssImport包所在的路徑，同時盡量不要設定為已有內容的目錄。

- 並發控制： `conf/job.cfg` 的配置項 `taskObjectCountLimit` ， `conf/sys.properties` 的配置項 `workerTaskThreadNum` ，請參考OssImport 架構及配置。

  如果小檔案比較多，單台ECS遷移速度上不去且CPU load不高，可以參考調高 `workerTaskThreadNum` 、調低 `taskObjectCountLimit` 查看效果。

- 其他動作過程遇到問題，一般都是配置問題，請參考分布式部署，並可以查看master和worker上 `workdir/Logs` 的記錄檔。

前期測試

建議您先搭建小型環境（比如2~3台ECS），遷移少量數據以驗證配置正確與否。單台ECS遷移頻寬能否達到預期，比如200Mbps，如您對遷移時間明確無要求，可不關注。

頻寬查看：您可使用 `iftop` 或 `Nload` （需先安裝，如 `yum install ***` ），ECS控制台頻寬統計有延時。

> **說明** 如果測試時檔案數目較少，可能無法驗證並發性，可以減少參數 `taskObjectCountLimit` ，比如減少到：檔案數/workerTaskThreadNum/worker總數。

遷移步驟

1. 遷移曆史數據。
   - 清任務、清配置。
   - 操作過程請參考分布式部署。

- 開始遷移。
- 您可在OSS控制台OSS Bucket中查看實際遷移完成的數據。

2. 設定鏡像回源，客戶業務系統讀寫切到OSS。
   - 在OSS控制台開啟OSS鏡像回源，回源地址為遷移源（第三方儲存）。
   - 客戶業務系統讀寫切換到OSS，假設業務系統修改好的時間為T2，則T2後不再有新數據寫到遷移源。
3. 遷移剩餘的數據。

   修改 `job.cfg` 配置項 `importSince=T1` ，請參考OssImport 架構及配置，遷移剩餘數據（T1~T2）特殊情況下，也可以使用 `job.cfg` 中 `importSince = 0, isSkipExistFile=true` 進行再次遷移，請參考OssImport 架構及配置。

關於遷移速度

- 單台遷移速度：如單台遷移速度不理想（比如小於200Mbps、且CPU load不高時），可參考官網文檔和上文，優化並發控制參數，即 `conf/job.cfg` 的配置項 `taskObjectCountLimit` ， `conf/sys.properties` 的配置項 `workerTaskThreadNum` 。
- 遷移ECS數量：參考ECS數量估算（相對於流量、儲存、時間成本，ECS費用，在遷移總成本中佔比較少）。加大ECS數量，會減少遷移時間。

OssImport分布式相關官網文檔

序號	官網文檔
1	OssImport 分布式部署
2	OssImport 架構及配置
3	OssImport 最佳實踐
4	OssImport 常見問題

5.3. 如何將HDFS容災備份到OSS

背景

當前業界有很多公司是以Hadoop技術構建資料中心，而越來越多的公司和企業希望將業務順暢地遷移到雲上。

在阿里雲上使用最廣泛的儲存服務是Object Storage Service。OSS的資料移轉工具ossimport2可以將您本地或第三方雲端儲存體服務上的檔案同步到OSS上，但這個工具無法讀取Hadoop檔案系統的數據，從而發揮Hadoop分布式的特點。並且，該工具只支援本地檔案，需要將HDFS上的檔案先下載到本地，再通過工具上傳，整個過程耗時又耗力。

阿里雲E-MapReduce團隊開發的Hadoop資料移轉工具**emr-tools**，能讓您從Hadoop叢集隨即轉移數據到OSS上。

本文介紹如何快速地將Hadoop檔案系統（HDFS）上的資料移轉到OSS。

前提條件

確保當前機器可以正常訪問您的Hadoop叢集，即能夠用Hadoop命令訪問HDFS。

```
hadoop fs -ls /
```

Hadoop資料移轉到OSS

1. 下載emr-tools。

   > 說明 emr-tools相容Hadoop 2.4.x、2.5.x、2.6.x、2.7.x版本，如果有其他Hadoop版本相容性的需求，請提交工單。

2. 解壓縮工具到本地目錄。

   ```
   tar jxf emr-tools.tar.bz2
   ```

3. 複製HDFS數據到OSS上。

   ```
   cd emr-tools
   ./hdfs2oss4emr.sh /path/on/hdfs oss://accessKeyId:accessKeySecret@bucket-name.oss-cn-hangzhou.aliyuncs.com/path/on/oss
   ```

   參數說明如下。

參數	說明
accessKeyId	訪問OSS API的密鑰。 獲取方式請參見如何獲取如何獲取AccessKeyId和AccessKeySecret。
accessKeySecret	
bucket-name.oss-cn-hangzhou.aliyuncs.com	OSS的訪問網域名稱，包括bucket名稱和endpoint地址。

   系統將啟動一個Hadoop MapReduce任務（DistCp）。

4. 運行完畢之後會顯示本次資料移轉的資訊。資訊內容類別似如下所示。

```
17/05/04 22:35:08 INFO mapreduce.Job: Job job_1493800598643_0009 completed successfully
17/05/04 22:35:08 INFO mapreduce.Job: Counters: 38
 File System Counters
        FILE: Number of bytes read=0
        FILE: Number of bytes written=859530
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=263114
        HDFS: Number of bytes written=0
        HDFS: Number of read operations=70
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=14
        OSS: Number of bytes read=0
        OSS: Number of bytes written=258660
        OSS: Number of read operations=0
        OSS: Number of large read operations=0
        OSS: Number of write operations=0
 Job Counters
        Launched map tasks=7
        Other local map tasks=7
        Total time spent by all maps in occupied slots (ms)=60020
        Total time spent by all reduces in occupied slots (ms)=0
        Total time spent by all map tasks (ms)=30010
        Total vcore-milliseconds taken by all map tasks=30010
        Total megabyte-milliseconds taken by all map tasks=45015000
 Map-Reduce Framework
        Map input records=10
        Map output records=0
        Input split bytes=952
        Spilled Records=0
        Failed Shuffles=0
        Merged Map outputs=0
        GC time elapsed (ms)=542
        CPU time spent (ms)=14290
        Physical memory (bytes) snapshot=1562365952
        Virtual memory (bytes) snapshot=17317421056
        Total committed heap usage (bytes)=1167589376
 File Input Format Counters
        Bytes Read=3502
 File Output Format Counters
        Bytes Written=0
 org.apache.hadoop.tools.mapred.CopyMapper$Counter
        BYTESCOPIED=258660
        BYTESEXPECTED=258660
        COPY=10
copy from /path/on/hdfs to oss://accessKeyId:accessKeySecret@bucket-name.oss-cn-hangzho
u.aliyuncs.com/path/on/oss does succeed !!!
```

5. 您可以用osscmd等工具查看OSS上數據情況。

```
osscmd ls oss://bucket-name/path/on/oss
```

OSS資料移轉到Hadoop

如果您已經在阿里雲上搭建了Hadoop叢集，可以使用如下命令把數據從OSS上遷移到新的Hadoop叢集。

```
./hdfs2oss4emr.sh oss://accessKeyId:accessKeySecret@bucket-name.oss-cn-hangzhou.aliyuncs.com/path/on/oss /path/on/new-hdfs
```

更多使用場景

除了線下的叢集，在ECS上搭建的Hadoop叢集也可以使用emr-tools，將自建叢集迅速地遷移到E-MapReduce服務上。

如果您的叢集已經在ECS上，但是在經典網路中，無法和VPC中的服務做很好的互操作，所以想把叢集遷移到VPC中。可以按照如下步驟遷移：

1. 使用emr-tools遷移數據到OSS上。
2. 在VPC環境中新建一個叢集（自建或使用E-MapReduce服務）。
3. 將數據從OSS上遷移到新的HDFS叢集中。

如果你使用E-MapReduce服務，還可以直接在Hadoop叢集中通過Spark、MapReduce和Hive等組件訪問OSS，這樣不僅可以減少一次數據複製（從OSS到HDFS），還可以極大的降低儲存成本。有關降低成本的詳細資料，請參見EMR+OSS：計算與儲存分離。

6.IaC自動化