Working with multipart uploads
You are here:

With the Hitachi API for Amazon S3, you can perform operations to create an individual object by uploading the object data in multiple parts. This process is called multipart upload.

This section of the Help starts with general information about and considerations for working with multipart uploads. Then, for each multipart upload operation you can request, this section:

• Describes the operation
• Shows the request line for the operation
• Describes the request headers for the operation
• Describes the response headers returned for a successful execution of the requested operation
• Shows the format of the request or response body, where applicable
• Explains the HTTP status codes that can be returned in response to requests for the operation
• Presents one or more examples of requests for the operation

For general information about S3 compatible requests and HCP responses to those requests, see Requests and responses.

For information about the examples, see Examples in this help.

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

About multipart uploads
You are here:

Multipart upload is the process of creating an object by breaking the object data into parts and uploading the parts to HCP individually. The result of a multipart upload is a single object that behaves the same as objects for which the data was stored by means of a single PUT object request. An object created by means of a multipart upload is called a multipart object.
Multipart uploads facilitate storing large objects. With a multipart upload:

• You can upload multiple parts of the object data to HCP concurrently, thereby speeding up the time it takes to store the whole object.

• You don't need to know the full size of the object data before you start uploading the data to HCP. Thus, multipart uploads support storing streaming data in real time.

• You can store an object over time. By spacing the uploads of the individual parts over time, you can reduce the use of bandwidth when other high-bandwidth operations are in progress.

• You can avoid the need to repeat a large upload operation when the connection is lost while the upload is in progress. Because each part you upload can be small, the time required to repeat the upload of a part can be short.

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

Multipart upload operations

You are here:

To create an object by means of a multipart upload, you perform these operations:

1. Initiate the multipart upload (see Initiating a multipart upload).

2. Upload the object data parts. To upload a part, you use either of these operations:
   a. Upload the part using data from outside HCP (see Uploading a part of a multipart upload)
   b. Upload the part using data copied from another object already stored in HCP (Uploading a part of a multipart upload)

3. Complete the multipart upload (see Completing a multipart upload).

You can also perform these operations related to multipart uploads:

• Abort a multipart upload (see Aborting a multipart upload)

• List the parts of an in-progress multipart upload (see Listing the parts of a multipart upload)

• For any given bucket, list the multipart uploads that are currently in progress in that bucket (see Listing the in-progress multipart uploads in a bucket)

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.
Considerations for working with multipart uploads

The following considerations apply to working with multipart uploads.

Support for multipart uploads

Whether a bucket supports multipart uploads depends on a bucket configuration setting. The tenant administrator determines the default for this setting for new buckets. This setting is not visible through the S3 compatible API.

If a bucket is empty, the tenant administrator can change it from supporting multipart upload to not supporting multipart upload. Whether the tenant administrator can change a bucket from not supporting multipart upload to supporting multipart upload depends on which other access protocols are enabled for the bucket.

To learn whether the bucket you're working with supports multipart upload, contact your tenant administrator.

Part size and count

The parts you upload for a multipart upload can be any size up to and including five gigabytes. However, the minimum size for the parts you include in a complete multipart upload request, except the last part, is one megabyte. The last part can be smaller than one megabyte.

A multipart upload can have at most ten thousand parts. Therefore, the maximum size for the object resulting from a multipart upload is five terabytes (ten thousand times five gigabytes).

Automatic abort of multipart uploads

If a multipart upload is never completed or aborted, the disjointed parts can remain in HCP indefinitely. Even though the multipart upload is incomplete, these parts count toward the storage used by the bucket where they were uploaded.

To prevent parts of multipart uploads from remaining in HCP indefinitely, the tenant administrator can set the maximum amount of time for which a multipart upload can remain incomplete before the multipart upload is automatically aborted. This time is counted from the time the multipart upload was initiated.

Each bucket has its own automatic abort time for multipart uploads. This time is not visible through the S3 compatible API. By default, the automatic abort time for new buckets is 30 days.

The tenant administrator can change the automatic abort time for a bucket at any time. When the tenant administrator changes the time, the new time applies to all subsequent multipart uploads as well as to multipart uploads that are already in progress.

For example, if you initiate a multipart upload on September 9th while the automatic abort time is set to 30 days, that multipart upload will be automatically aborted on October 9th. If the tenant administrator then changes the automatic abort time to 25 days, the date on which the multipart upload will be automatically aborted changes to October 4th.
Creation date and time for multipart objects

The creation date and time for the object resulting from a multipart upload is the time when the multipart upload was completed, regardless of the amount of time that has elapsed since the multipart upload was initiated.

Retention setting for multipart objects

The retention setting for the object resulting from a multipart upload is calculated based on the time when the multipart upload was completed, regardless of the amount of time that has elapsed since the multipart upload was initiated.

For example, suppose the bucket default retention setting for new objects is 10 days. If a multipart upload in the bucket is completed on September 9th, the resulting object will be under retention until September 19th, regardless of when the multipart upload was initiated.

Retention is not set for the individual parts of an in-progress multipart upload. Replaced parts and parts not used in completed multipart uploads are automatically deleted regardless of the retention setting for the completed multipart object.

For more information about retention, see Retention.

Multipart uploads and versioning

If versioning is enabled for a bucket, you can use multipart uploads to store new versions of objects in the bucket. However, HCP does not check whether versioning is enabled until you submit the request to complete a multipart upload. If versioning is disabled and an object with the same name already exists when you submit the complete request, the request fails with a 409 (Conflict) status code.

If versioning is disabled for a bucket and you are concerned about object conflicts, before you initiate a multipart upload for an object, you can check the bucket for an existing object with the same name. For information about checking for existing objects, see Checking the existence of an object or folder.

Tip: If a request to complete a multipart upload fails because versioning is disabled for the bucket and an object with the same name already exists, you can delete the existing object and then submit the complete request again with the same request body. A failed request to complete a multipart upload has no effect on the uploaded parts, so you don't need to upload them again.

Multiple multipart uploads for objects with the same name

Regardless of the bucket versioning status, you can initiate multiple concurrent multipart uploads with the same object name in a single bucket. These multipart uploads are distinguished by upload ID. Because you specify both the upload ID and the object name in each request to upload a part, each part you upload applies to only one of the multipart
uploads.

If versioning is enabled for the bucket, each multipart upload with the same object name, when completed, creates a separate version of the object. The order of the versions is determined by the order in which the multipart uploads are completed. The oldest version is the one resulting from the multipart upload that was completed first. The most recent version is the one resulting from the multipart upload that was completed last.

If versioning is disabled for the bucket, the object stored is the one resulting from the multipart upload that was completed first. Each subsequent completion of a multipart upload for the same object results in a 409 (Conflict) status code, and the resulting object is not stored. The parts of a multipart upload that cannot be completed remain in HCP until the multipart upload is aborted.

For more information about versioning, see Versioning.

Multipart uploads for objects under retention or on hold

You cannot store a new version of an object that’s under retention or on hold. However, HCP does not check the retention or hold status of the existing object until you submit the request to complete the multipart upload. If the object is under retention or on hold when you submit the complete request, the request fails with a 403 (Forbidden) status code.

Before you initiate a multipart upload to store a new version of an object, you should use the HTTP data access protocol to check the retention and hold status of the existing object. Because this status can change, you may also want to check the retention and hold status of the existing object periodically while the multipart upload is in progress. If the object is put under retention or on hold while the multipart upload is in progress, you can abort the multipart upload.

For more information about retention and hold, see Retention. For information about using the HTTP data access protocol to check retention status, see Checking the existence of an object or multiple versions of an object.

Multipart uploads and replicated buckets

All requests related to a given multipart upload must be sent to the same HCP system. That is, even if the bucket on which you initiate the multipart upload is replicated on other HCP systems, requests to upload parts for, list parts for, complete, or abort that multipart upload must be sent to the system on which you initiated the multipart upload.

For more information about replication, see Replication.

Initiating a multipart upload

You are here:

You use the HTTP POST method with the uploads query parameter to initiate a multipart upload in a bucket.
To initiate a multipart upload, you must be an authenticated user. Additionally, you need write permission for the target bucket.

When you initiate a multipart upload, you specify a name for the object you’re creating. For information about naming objects, see Object names.

In initiate the multipart upload request, you can specify custom metadata to be added to the object when the multipart upload is completed. To do this, you use x-amz-meta- headers. For information about custom metadata, see Custom metadata.

In the initiate multipart upload request, you can specify an ACL to be added to the object when the multipart upload is completed. To do this, you need to use ACL headers. You cannot use an ACL request body when initiating a multipart upload. For information about ACLs, see Access control lists.

If the ACL you specify in a request to initiate a multipart upload is invalid, HCP returns a 400 (Bad Request) or 501 (Not Implemented) status code and does not initiate the multipart upload.

You cannot change the custom metadata or ACL for a multipart object until the multipart upload used to create the object has been completed.

When you complete a multipart upload, you become the owner of the resulting object. For information about object ownership, see Object owners.

Upload IDs

In response to a request to initiate a multipart upload, HCP returns an upload ID. You use this ID in conjunction with the object name to identify the multipart upload in:

• Requests to upload parts for the multipart upload
• Requests to upload parts for the multipart upload by copying
• Requests to complete the multipart upload
• Requests to abort the multipart upload
• Requests to list the parts that have been uploaded for the multipart upload
• Responses to requests to list in-progress multipart uploads

Object encryption

The response headers returned in response to a successful request to initiate a multipart upload include x-amz-server-side-encryption, which indicates whether objects stored in HCP are encrypted. If stored objects are encrypted, the value of this header represents the encryption algorithm and key length HCP is using. If stored objects are not encrypted, the value of this header is None.
Request line (POST object initiate multipart upload)

You are here:

Depending on whether the bucket name is included in the hostname in the S3 compatible request, a request to initiate a multipart upload has either of these formats:

• With the bucket name included in the hostname:

  POST /object-name?uploads HTTP/1.1

• With the bucket name following the hostname:

  POST /bucket-name/object-name?uploads HTTP/1.1

The uploads query parameter is case sensitive.

Request headers (POST object initiate multipart upload)

You are here:

The table below describes the headers you can use in a request to initiate a multipart upload.

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>See Authentication.</td>
<td>Yes</td>
</tr>
<tr>
<td>Date</td>
<td>See Common request headers.</td>
<td>Date or x-amz-date</td>
</tr>
<tr>
<td>Host</td>
<td>See Common request headers.</td>
<td>Yes</td>
</tr>
<tr>
<td>x-amz-acl</td>
<td>Adds a canned ACL to the resulting object when the multipart upload is completed. This header is used only to add a canned ACL to a multipart object. If you’re using individual x-amz-grant- headers to add the ACL, the x-amz-acl header is invalid.</td>
<td>No</td>
</tr>
<tr>
<td>Request header</td>
<td>Description</td>
<td>Required</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td></td>
<td>For valid values for this header, see <a href="#">Canned ACLs</a>.</td>
<td></td>
</tr>
<tr>
<td>x-amz-date</td>
<td>See <a href="#">Common request headers</a>.</td>
<td>x-amz-date or Date</td>
</tr>
<tr>
<td>x-amz-grant-full-control</td>
<td>Grants full control over the resulting object to one or more specified grantees when the multipart upload is completed.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>If you’re using a canned ACL to add an ACL to the object resulting from a multipart upload, the x-amz-grant-full-control header is invalid.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For valid values for this and the following x-amz-grant-headers, see <a href="#">Specifying an ACL with headers</a>.</td>
<td></td>
</tr>
<tr>
<td>x-amz-grant-read</td>
<td>Grants the browse and read data access permissions for the resulting object to one or more specified grantees when the multipart upload is completed.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>If you’re using a canned ACL to add an ACL to the object resulting from a multipart upload, the x-amz-grant-read header is invalid.</td>
<td></td>
</tr>
<tr>
<td>x-amz-grant-read-acp</td>
<td>Grants the read ACL data access permission for the resulting object to one or more specified grantees when the multipart upload is completed.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>If you’re using a canned ACL to add an ACL to the object resulting from a multipart upload, the x-amz-grant-read-acp header is invalid.</td>
<td></td>
</tr>
<tr>
<td>x-amz-grant-write</td>
<td>Grants the write and delete data access permissions for the resulting object to one or more specified grantees when the multipart upload is completed.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>If you’re using a canned ACL to add an ACL to the object resulting from a multipart upload, the x-amz-grant-write</td>
<td></td>
</tr>
<tr>
<td>Request header</td>
<td>Description</td>
<td>Required</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>x-amz-grant-write-acp</td>
<td>Grants the write ACL data access permission for the resulting object to one or more specified grantees when the multipart upload is completed. If you’re using a canned ACL to add an ACL to the object resulting from a multipart upload, the x-amz-grant-write-acp header is invalid.</td>
<td>No</td>
</tr>
<tr>
<td>x-amz-meta-</td>
<td>Adds custom metadata to the resulting object when the multipart upload is completed. For information about using this header, see Storing custom metadata with the S3 compatible API.</td>
<td>No</td>
</tr>
<tr>
<td>x-hcp-pretty-print</td>
<td>Optionally, requests that the XML response body be formatted for readability. Valid values are:</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>•true — Format the XML response body for readability.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>•false — Do not apply any special formatting to the XML response body. The default is false.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The values true and false are not case sensitive.</td>
<td></td>
</tr>
</tbody>
</table>

Response headers (POST object initiate multipart upload)

The table below describes the headers returned in response to a successful request to initiate a multipart upload.
<table>
<thead>
<tr>
<th>Response header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the response body. For a request to initiate a multipart upload, the value of this header is always application/xml;charset=UTF-8.</td>
</tr>
<tr>
<td>Date</td>
<td>See <a href="#">Common response headers</a>.</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>Indicates that HCP could not determine the size of the response body before formulating the response. For a request to initiate a multipart upload, the value of this header is always chunked.</td>
</tr>
</tbody>
</table>
| x-amz-abort-date        | Specifies the date and time when the multipart upload will be automatically aborted if it has not already been completed. The date and time are expressed in this format:  

DDD, dd MMM yyyy hh:mm:ss GMT

For example:

Sat, 25 Mar 2017 17:19:26 GMT

For information about automatic abort of multipart uploads, see [Considerations for working with multipart uploads](#). |
| x-amz-server-side-encryption | Specifies whether objects stored in HCP are encrypted. Possible values are:  

- If objects are encrypted, AES256  
- If objects are not encrypted, None |

---

**Response body (POST object initiate multipart upload)**

You are here:

HCP returns information about the multipart upload that results from a successful initiate request in an XML response body, in this format:

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```
   <Bucket>bucket-name</Bucket>
   <Key>object-name</Key>
   <UploadId>upload-id</UploadId>
</InitiateMultipartUploadResult>

The table below describes the XML elements in the response body returned in response to a request to initiate a multipart upload. The elements are listed in alphabetical order.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bucket</td>
<td>Child of the InitiateMultipartUploadResult element.</td>
</tr>
<tr>
<td></td>
<td>The Bucket element specifies the name of the bucket where the multipart upload was initiated.</td>
</tr>
<tr>
<td>InitiateMultipartUploadResult</td>
<td>Root element.</td>
</tr>
<tr>
<td>Key</td>
<td>Child of the InitiateMultipartUploadResult element.</td>
</tr>
<tr>
<td></td>
<td>The Key element specifies the name of the object for which the multipart upload was initiated.</td>
</tr>
<tr>
<td>UploadId</td>
<td>Child of the InitiateMultipartUploadResult element.</td>
</tr>
<tr>
<td></td>
<td>The UploadId element specifies the upload ID for the multipart upload initiated by the request.</td>
</tr>
</tbody>
</table>

*Trademarks and Legal Disclaimer*

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

**Status codes (POST object initiate multipart upload)**

*You are here:*

The table below describes HTTP status codes that can be returned in response to a request to initiate a multipart upload. For more information about HTTP status codes and the error codes that can accompany them, see Error codes.
<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>HCP successfully initiated the multipart upload.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified object name is too long.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• An ACL grant header specifies an invalid grantee.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request specifies anonymous access. Only an authenticated user can initiate a multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request are invalid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You do not have permission to initiate multipart uploads in the specified bucket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified bucket does not support multipart uploads.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The S3 compatible API is currently disabled for the specified bucket.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The specified bucket does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
<td>An internal error occurred.  If this error persists, contact your tenant administrator.</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented</td>
<td>The request includes the x-amz-acl header with an invalid value.</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
<td>HCP is temporarily unable to handle the request, probably due to system overload, maintenance, or upgrade.  Try the request again, gradually increasing the delay between each successive attempt.  If this error persists, contact your tenant administrator.</td>
</tr>
</tbody>
</table>
Example: Initiating a multipart upload

You are here:

Here’s a sample POST request that initiates a multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket.

Request with s3curl command line

```
./s3curl.pl --id=lgreen --post -- -k
   "https://finance.europe.hcp.example.com...ns.pdf?uploads"
   -H "x-hcp-pretty-print: true"
```

Request headers

```
POST /acctg/RulesAndRegulations.pdf?uploads HTTP/1.1
Host: finance.europe.hcp.example.com
Date: Thu, 10 Jan 2019 17:19:26 +0000
Authorization: AWS bGdyZWVu:u0eqPDwPw1SR7kdZn87dnGACzaM=
   x-hcp-pretty-print: true
```

Response headers

```
HTTP/1.1 200 OK
Date: Thu, 10 Jan 2019 17:19:26 GMT
x-amz-server-side-encryption: None
x-amz-abort-date: Sat, 25 Mar 2017 17:19:26 GMT
Content-Type: application/xml;charset=UTF-8
Transfer-Encoding: chunked
```

Response body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <Bucket>finance</Bucket>
   <Key>acctg/RulesAndRegulations.pdf</Key>
   <UploadId>94837746087105</UploadId>
</InitiateMultipartUploadResult>
```

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.
Uploading a part of a multipart upload

You use the HTTP PUT method with the uploadId and partNumber query parameters to upload a part of a multipart upload. The bucket where you upload the part must be the same bucket as the one where the multipart upload was initiated.

To upload a part, you must be an authenticated user. You need to use the same user account as the account used in the request to initiate the multipart upload. Additionally, you need write permission for the bucket where you’re uploading the part.

For a request to upload a part, the request body consists of the data in a specified file.

You use the uploadId query parameter to specify the upload ID of the multipart upload for which you're uploading the part. If the upload ID you specify doesn't match the upload ID of an in-progress multipart upload or isn't associated with the object specified in the request, HCP returns a 404 (Not Found) status code and does not upload the part.

When you upload a part, you use the partNumber query parameter to specify the part number of the data you're uploading. The part number must be an integer in the range 1 (one) through 10,000.

The parts of the data for an object are ordered, but the part numbers do not need to start at one and do not need to be consecutive numbers. For example, if you're uploading the data for an object in three parts, you could number the parts 1, 2, and 3, but you could also number them 2, 7, and 19.

You can upload the parts of a multipart upload in any order. However, in the request to complete the multipart upload, you need to list the parts in ascending numeric order.

You can upload multiple parts of a multipart upload concurrently.

If a request to upload a part includes any custom metadata (x-amz-meta-) headers, HCP returns a 400 (Bad Request) status code and does not upload the part.

HCP ignores ACL headers in requests to upload a part.

Part size

The parts you upload for a multipart upload can be any size up to and including five gigabytes. However, the minimum size for the parts you include in a complete multipart upload request, except the last part, is one megabyte. The last part can be smaller than one megabyte.

The maximum size for the object resulting from a multipart upload is five terabytes.

Replacing parts of a multipart upload
When you upload a part of a multipart upload, if you use a part number that was already used for a part in the same multipart upload, the new part replaces the part that was previously uploaded. The part that was replaced is automatically deleted.

You can replace a part multiple times. The part for which the upload finished last is the one that is used for the object created by the multipart upload.

ETags

When you upload a part of a multipart upload, HCP returns the ETag for the part in the ETag response header. The ETag is an identifier for the part data. In a request to complete a multipart upload, you need to specify the ETag along with the part number for each part you want to include in the resulting object.

Ensuring data integrity

When you upload a part of a multipart upload, you can use the Content-MD5 request header to ensure the integrity of the part data. The value you specify for this header must be the Base64-encoded MD5 hash of the original part data.

When you include the Content-MD5 header in a request to upload a part, HCP calculates the Base64-encoded MD5 hash of the data it receives and compares that to the header value. If the values don’t match, HCP returns a 400 (Bad Request) status code and does not upload the part.

Object encryption

The response headers returned in response to a successful request to upload a part of a multipart upload include x-amz-server-side-encryption, which indicates whether objects stored in HCP are encrypted. If stored objects are encrypted, the value of this header represents the encryption algorithm and key length HCP is using. If stored objects are not encrypted, the value of this header is None.

Saving network bandwidth

You can use the Expect request header in a request to upload a part to tell the application not to send the request body (the data) to HCP if the request headers are rejected. This prevents unnecessary network bandwidth usage.

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

Request line (PUT object upload part)

You are here:

Depending on whether the bucket name is included in the hostname in the S3 compatible request, a request to upload a part of a multipart upload has either of these formats:

• With the bucket name included in the hostname:
PUT /object-name?uploadId=upload-id&partNumber=part-number
HTTP/1.1

• With the bucket name following the hostname:

PUT /bucket-name/object-name?uploadId=upload-id
&partNumber=part-number HTTP/1.1

The uploadId and partNumber query parameters are not case sensitive.

Request headers (PUT object upload part)

You are here:

The table below describes the headers you can use in a request to upload a part of a multipart upload.

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>See Authentication.</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Length</td>
<td>Specifies the size, in bytes, of the data being uploaded.</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-MD5</td>
<td>Directs HCP to check the integrity of the data it receives by comparing a Base64-encoded MD5 hash of that data to the value specified by this header. The valid value for this header is the Base64-encoded MD5 hash of the data in the request body.</td>
<td>No</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the data being uploaded. Valid values are Internet media types (for example, text/plain, application/xml, or image/jpeg).</td>
<td>No</td>
</tr>
<tr>
<td>Date</td>
<td>See Common request headers.</td>
<td>Date or x-amz-date</td>
</tr>
<tr>
<td>Expect</td>
<td>Tells the application not to send the request body if the request headers are rejected. The only valid value is 100-continue. This value is not case sensitive.</td>
<td>No</td>
</tr>
<tr>
<td>Host</td>
<td>See Common request headers.</td>
<td>Yes</td>
</tr>
<tr>
<td>x-amz-date</td>
<td>See Common request headers.</td>
<td>x-amz-date or Date</td>
</tr>
</tbody>
</table>
Response headers (PUT object upload part)

You are here:

The table below describes the headers returned in response to a successful request to upload a part of a multipart upload.

<table>
<thead>
<tr>
<th>Response header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-length</td>
<td>Specifies the size, in bytes, of the response body. In response to a successful request to upload a part of a multipart upload, the value of this header is always 0 (zero).</td>
</tr>
<tr>
<td>Date</td>
<td>See Common response headers.</td>
</tr>
<tr>
<td>ETag</td>
<td>Specifies the ETag for the uploaded part.</td>
</tr>
<tr>
<td>x-amz-server-side-encryption</td>
<td>Specifies whether objects stored in HCP are encrypted. Possible values are:</td>
</tr>
<tr>
<td></td>
<td>• If objects are encrypted, AES256</td>
</tr>
<tr>
<td></td>
<td>• If objects are not encrypted, None</td>
</tr>
<tr>
<td>x-amz-version-id</td>
<td>Specifies a version ID for the uploaded part. This header is returned only while versioning is enabled for the bucket.</td>
</tr>
</tbody>
</table>

Status codes (PUT object upload part)

You are here:

The table below describes HTTP status codes that can be returned in response to a request to upload a part of a multipart upload. For more information about HTTP status codes and the error codes that can accompany them, see Error codes.
<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>HCP successfully uploaded the part.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified part number is not an integer in the range 1 (one) through 10,000.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The value specified by the Content-MD5 header does not match the Base64-encoded MD5 hash of the data HCP received.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request includes a custom metadata (x-amz-meta-) header.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request specifies anonymous access. Only an authenticated user can upload parts for a multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request are invalid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request do not match the credentials used to initiate the multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You do not have permission to upload parts in the specified bucket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The S3 compatible API is currently disabled for the specified bucket.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>One of these:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified multipart upload does not exist. Either the upload ID is invalid, or the multipart upload was aborted or completed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In the case of completion, if the part being uploaded is a replacement for an existing part, the existing part is used for completing the multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified bucket does not exist.</td>
</tr>
<tr>
<td>411</td>
<td>Length Required</td>
<td>Either the request does not include a Content-Length header, or the request includes a Content-Length header with no value.</td>
</tr>
</tbody>
</table>
### Code 413: Request Entity Too Large

- **Meaning**: Request entity too large.
- **Description**: The part you are trying to upload is too big for the amount of space left in the bucket.

### Code 500: Internal Server Error

- **Meaning**: Internal server error.
- **Description**: An internal error occurred. If this error persists, contact your tenant administrator.

### Code 503: Service Unavailable

- **Meaning**: Service unavailable.
- **Description**: HCP is temporarily unable to handle the request, probably due to system overload, maintenance, or upgrade. Try the request again, gradually increasing the delay between each successive attempt. If this error persists, contact your tenant administrator.

---

**Example: Uploading a part**

You are here:

Here’s a sample PUT request that uploads a part of the multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket. The data for the part is in a file named /file-parts-temp/RulesAndRegs-Acctg-1.

The example shows the response headers HCP returns while versioning is enabled for the bucket and while versioning is disabled for the bucket.

Request with `s3curl` command line

```
./s3curl.pl --id=lgreen --put=/file-parts-temp/RulesAndRegs-Acctg-1 -- -k "https://finance.europe.hcp.example.com/RulesAndRegulations.pdf?uploadId=94837746087105&partNumber=1"
```

Request headers

```
PUT /acctg/RulesAndRegulations.pdf?uploadId=94837746087105&partNumber=1 HTTP/1.1
Host: finance.europe.hcp.example.com
Date: Thu, 10 Jan 2019 17:19:26 +0000
Authorization: AWS bGdyZWVu:oVRZgtbngViycDs+i2p2NP5Y3zY=
```

Response headers with versioning enabled

---

**Trademarks and Legal Disclaimer**

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

---

https://knowledge.hitachivantara.com/Documents/Storage/Content_Platform/8.2/Developer_documentation/Using_the_Hitachi...
Uploading a part of a multipart object by copying

You are here:

You can upload a part of a multipart upload by copying all or part of the data for an existing object. To do this, you use the HTTP PUT method with the uploadId and partNumber query parameters and the x-amz-copy-source header. The bucket where you upload the part must be the same bucket as the one where the multipart upload was initiated.

You use the x-amz-copy-source header to specify the object you’re copying (the source object) and the bucket that contains that object (the source bucket). The source bucket can be the same bucket as the one to which you’re uploading the part. Alternatively, the source bucket can be a different bucket that’s owned by the same tenant.

To upload a part by copying, you must be an authenticated user. You need to use the same user account as the account used in the request to initiate the multipart upload. Additionally, you need write permission for the bucket where you're uploading the part and read permission for the source bucket or for the source object itself.

You use the uploadId query parameter to specify the upload ID of the multipart upload for which you're uploading the part. If the upload ID you specify doesn't match the upload ID of an in-progress multipart upload or isn't associated with the object specified in the request, HCP returns a 404 (Not Found) status code and does not upload the part.

When you upload a part by copying, you use the partNumber query parameter to specify the part number of the data you're uploading. The part number must be an integer in the range 1 (one) through 10,000.

The parts of the data for an object are ordered, but the part numbers do not need to start at one and do not need to be consecutive numbers. For example, if you’re uploading the data for an object in three parts, you could number the parts 1, 2, and 3, but you could also number them 2, 7, and 19.
You can upload the parts of a multipart upload in any order. However, in the request to complete the multipart upload, you need to list the parts in ascending numeric order.

You can upload multiple parts of a multipart upload concurrently.

If a request to upload a part by copying includes any custom metadata (x-amz-meta-) headers, HCP returns a 400 (Bad Request) status code and does not upload the part.

HCP ignores ACL headers in requests to upload a part by copying.

Source object version

By default, when you upload a part by copying, HCP copies data from the current version of the source object. However, while versioning is enabled for the source bucket, you can use the versionId query parameter with the source object specification to copy data from a specific version of the object. If the version identified by the versionId parameter does not exist or is a delete marker, HCP returns a 404 (Not Found) status code.

Copying part of the data for an object

To upload a part by copying only part of the data for the source object, you use the x-amz-copy-source-range request header. The value of the x-amz-copy-source-range header is the range of bytes you want to copy. The first byte of the data for an object is in position 0 (zero), so a range of one to five specifies the second through sixth bytes, not the first through fifth.

To specify a byte range in an x-amz-copy-source-range header, use this format:

```
bytes=start-position-end-position
```

For example, to copy 6,000,000 bytes, starting from the 1,000,001st byte, you would use this x-amz-copy-source-range header:

```
x-amz-copy-source-range: bytes=1000000-6999999
```

These considerations apply to x-amz-copy-source-range header values:

- If you specify a valid range in which the start position is less than the size of the source object data, HCP uploads the part and returns a 200 (OK) status code.

- If you specify a valid range in which the start position is greater than or equal to the size of the source object data, HCP returns a 400 (Bad Request) status code and does not upload a part.

- If you specify an invalid range (for example, a range in which the start position is greater than the end position, HCP returns a 400 (Bad Request) status code and does not upload a part.

A request to upload a part by copying a range of bytes in a multipart object is most efficient when the start and end bytes for the requested range are aligned with the start and end bytes for one or more of the parts uploaded to create the object.
Part size

The parts you upload for a multipart upload can be any size up to and including five gigabytes. However, the minimum size for the parts you include in a complete multipart upload request, except the last part, is one megabyte. The last part can be smaller than one megabyte.

The maximum size for the object resulting from a multipart upload is five terabytes.

Replacing parts of a multipart upload

When you upload a part of a multipart upload by copying, if you use a part number that was already used for a part in the same multipart upload, the new part replaces the part that was previously uploaded. The part that was replaced is automatically deleted.

You can replace a part multiple times. The part for which the upload finished last is the one that is used for the object created by the multipart upload.

ETags

When you upload a part of a multipart upload by copying, HCP returns the ETag for the part in the ETag response header. The ETag is an identifier for the part data. In a request to complete a multipart upload, you need to specify the ETag along with the part number for each part you want to include in the resulting object.

Conditionally uploading a part by copying

You can choose to upload a part by copying only if the ETag or last modification date and time of the source object meet certain criteria. To specify these criteria, you use request headers:

• The x-amz-copy-source-if-match and x-amz-copy-source-if-none-match request headers compare the ETag for the source object or object version to one or more values that you specify. Typically, each value is the ETag for an object or object version of interest.

• The x-amz-copy-source-if-modified-since and x-amz-copy-source-if-unmodified-since request headers compare the date and time the source object or object version was last modified to a date and time that you specify.

If the source object or object version meets all the conditions specified by the conditional headers included in the request, HCP uploads the part by copying the object data. If the source object or object version does not meet the condition specified by:

• An x-amz-copy-source-if-match or x-amz-copy-source-if-unmodified-since header, HCP returns a 412 (Precondition Failed) status code and does not upload the part

• An x-amz-copy-source-if-none-match or x-amz-copy-source-if-modified-since header, HCP returns a 304 (Not Modified) status code and does not upload the part

If a request includes multiple different conditional headers, HCP processes any x-amz-copy-source-if-match and x-amz-copy-source-if-none-match headers before any x-amz-copy-source-if-modified-since or x-amz-copy-source-if-
unmodified-since headers. If a request includes more than one of any given header, HCP processes only the first one of those headers and ignores the rest.

Note: The x-amz-copy-source-if-modified-since and x-amz-copy-source-if-unmodified-since request headers are not compatible with s3curl.

Client timeouts

Uploading a part of a multipart upload by copying a large amount of data from another object can take some time. If a client times out because an upload part copy operation is taking too long, HCP continues the operation in the background.

Because the connection to the client is broken, HCP cannot report the completion of the upload part copy operation to the client. To see whether the operation has finished successfully, use a GET request with the multipart upload ID to list the parts of the multipart upload. If the operation finished successfully, the part appears in the list. If the operation is still in progress or has failed, the part does not appear in the list.

If upload part copy operations are causing a client to time out, consider increasing the client timeout interval.

Request line (PUT object upload part copy)

You are here:

Depending on whether the bucket name is included in the hostname in the S3 compatible request, for a request to upload a part of a multipart upload by copying has either of these formats:

• With the bucket name included in the hostname:

PUT /object-name?uploadId=upload-id&pageNumber=page-number
   HTTP/1.1

• With the bucket name following the hostname:

PUT /bucket-name/object-name?uploadId=upload-id&pageNumber=page-number HTTP/1.1

The uploadId and pageNumber query parameters are not case sensitive.

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.
## Request headers (PUT object upload part copy)

You are here:

The table below describes the headers you can use in a request to upload a part of a multipart upload by copying.

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>See <a href="#">Authentication</a>.</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the data being copied. Valid values are Internet media types (for example, text/plain, application/xml, or image/jpeg).</td>
<td>No</td>
</tr>
<tr>
<td>Date</td>
<td>See <a href="#">Common request headers</a>.</td>
<td>Date or x-amz-date</td>
</tr>
<tr>
<td>Host</td>
<td>See <a href="#">Common request headers</a>.</td>
<td>Yes</td>
</tr>
<tr>
<td>x-amz-copy-source</td>
<td>Specifies the source bucket and object or object version, in this format: /bucket-name/source-object-name [?versionId=source-object-version-id] The initial forward slash (/) is required. Valid values for source-object-version-id are the IDs of versions of the source object specified in the request. The versionId query parameter is not case sensitive. If you include the versionId query parameter in the x-amz-copy-source header with an invalid value while versioning is enabled, HCP returns a 404 (Not Found) status code and does not perform the upload part copy operation. If you include the versionId query parameter in the x-amz-copy-source header while versioning is disabled, the parameter is ignored, and the current version of the specified object is used as the source for the upload part copy operation.</td>
<td>Yes</td>
</tr>
<tr>
<td>x-amz-copy-source-if-match</td>
<td>Specifies one or more values for comparison with the ETag of the object to be copied.</td>
<td>No</td>
</tr>
<tr>
<td>Request header</td>
<td>Description</td>
<td>Required</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>x-amz-copy-source-if-match</td>
<td>the specified source object or object version. If the ETag matches one of the specified values, HCP continues processing the request. If the ETag doesn't match any of the specified values, HCP returns a 412 (Precondition Failed) status code and does not upload the part. To specify the values for this header, use this format: &quot;value&quot;, &quot;value&quot;... In this format, each value can be any string of one or more characters and must be enclosed in double quotation marks (&quot;). Alternatively, you can specify a single asterisk (*) as the value for the x-amz-copy-source-if-match header. All ETags match an asterisk in an x-amz-copy-source-if-match header.</td>
<td></td>
</tr>
<tr>
<td>x-amz-copy-source-if-</td>
<td>Specifies a date and time, in Greenwich Mean Time (GMT), for comparison with the date and time the specified source object or object version was last modified. If the modification date and time is later than the specified date and time, HCP continues processing the request. If the modification date and time is equal to or earlier than the specified date and time, HCP returns a 412 (Precondition Failed) status code and does not upload the part. To specify the date and time for this header, use one of these formats: • DDD,ddMMyyyyHH:mm:ss(+0000)GMT For example: Tue, 07 Feb 2017 14:27:05 +0000 • DDDD,dd-MMM-yyyyHH:mm:ss(+0000)GMT For example: Tuesday, 07-Feb-17 14:27:05 +0000 • DDDMMmdHH:mm:ssyyyy For example: Tue Feb 7 14:27:05 2017</td>
<td></td>
</tr>
<tr>
<td>modified-since</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Request header</td>
<td>Description</td>
<td>Required</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>In these formats:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• DDD is the three-letter abbreviation for the day of the week, with an uppercase first letter (for example, Mon).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• DDDD is the day of the week fully spelled out, with an uppercase first letter (for example, Monday).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• d is the one- or two-digit day of the month, as applicable.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• dd is the two-digit day of the month.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• MMM is the three-letter abbreviation for the month, with an uppercase first letter (for example, Feb).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• yy is the last two digits of the year. HCP assumes that the intended year is within 80 years before or 20 years after the current year.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• yyyy is the four-digit year.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• HH is the hour on a 24-hour clock.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• mm is the number of minutes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ss is the number of seconds.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If the value specified by the x-amz-copy-source-if-modified-since header doesn’t conform to one of the formats shown above, the header is ignored.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specifies one or more values for comparison with the ETag of the specified source object or object version. If the ETag doesn’t match any of the specified values, HCP continues processing the request. If the ETag matches any of the specified values, HCP returns a 412 (Precondition Failed) status code and does not upload the part.</td>
<td>No</td>
</tr>
<tr>
<td>x-amz-copy-source-if-none-match</td>
<td>To specify the values for this header, use this format: &quot;value&quot;[], &quot;value&quot;[]...</td>
<td></td>
</tr>
</tbody>
</table>
## Request header

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-amz-copy-source-if-unmodified-since</td>
<td>Specifies a date and time, in Greenwich Mean Time (GMT), for comparison with the date and time the specified source object or object version was last modified. If the modification date and time is equal to or earlier than the specified date and time, HCP continues processing the request. If the modification date and time is later than the specified date and time, HCP returns a 412 (Precondition Failed) status code and does not upload the part. For valid values for this header, see the description of the x-amz-copy-source-if-modified-since header above.</td>
<td>No</td>
</tr>
<tr>
<td>x-amz-copy-source-range</td>
<td>Copies part of the source object. For valid values for this header, see <a href="#">Uploading a part of a multipart object by copying</a>.</td>
<td>No</td>
</tr>
<tr>
<td>x-amz-date</td>
<td>See <a href="#">Common request headers</a>.</td>
<td>x-amz-date or Date</td>
</tr>
</tbody>
</table>

---

**Response headers (PUT object upload part copy)**

You are here:

The table below describes the headers returned in response to a successful request to upload a part of a multipart upload by copying.

<table>
<thead>
<tr>
<th>Response header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the response body. For a</td>
</tr>
</tbody>
</table>
### Response header | Description
---|---
| | request to upload a part by copying, the value of this header is always application/xml;charset=UTF-8.
| Date | See [Common response headers](#).
| ETag | Specifies the ETag for the uploaded part.
| Transfer-Encoding | Indicates that HCP could not determine the size of the response body before formulating the response. For a request to upload a part by copying, the value of this header is always chunked.
| x-amz-copy-source-version-id | Specifies the version ID of the source object. This header is returned only while versioning is enabled for the source bucket.
| x-amz-server-side-encryption | Specifies whether objects stored in HCP are encrypted. Possible values are:
  - If objects are encrypted, AES256
  - If objects are not encrypted, None
| x-amz-version-id | Specifies a version ID for the resulting part of the multipart upload. This header is returned only while versioning is enabled for the bucket where you're performing the multipart upload.

**Response body (PUT object upload part copy)**

You are here:

HCP returns information about the part of a multipart upload that results from a successful upload part copy request in an XML response body, in this format:

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <LastModified>date-and-time-last-modified</LastModified>
  <ETag>"etag"</ETag>
</CopyPartResult>
```
The table below describes the XML elements in the response body returned in response to a request to upload a part by copying. The elements are listed in alphabetical order.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CopyPartResult</td>
<td>Root element.</td>
</tr>
<tr>
<td>ETag</td>
<td>Child of the CopyPartResult element.</td>
</tr>
<tr>
<td></td>
<td>The ETag element specifies the ETag for the uploaded part.</td>
</tr>
<tr>
<td>LastModified</td>
<td>Child of the CopyPartResult element.</td>
</tr>
<tr>
<td></td>
<td>The LastModified element specifies the date and time when the uploaded part was last modified, in Greenwich Mean Time (GMT). The date and time are expressed in this format: yyyy-MM-ddTHH:mm:ss.SSSZ</td>
</tr>
<tr>
<td></td>
<td>For example: 2019-02-18T19:46:03.856Z</td>
</tr>
</tbody>
</table>

HTTP status codes (PUT object upload part copy)

You are here:

The table below describes HTTP status codes that can be returned in response to a request to upload a part of a multipart upload by copying. For more information about HTTP status codes and the error codes that can accompany them, see Error codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>HCP successfully uploaded the part.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td>Code</td>
<td>Meaning</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 403  | Forbidden | Possible reasons include:  
• The request specifies anonymous access. Only an authenticated user can upload parts for a multipart upload.  
• The credentials provided with the request are invalid.  
• The credentials provided with the request do not match the credentials used to initiate the multipart upload.  
• You do not have permission to upload parts in the specified bucket.  
• The S3 compatible API is currently disabled for the specified bucket.  
• The source object exists, but the HCP system does not have the source object data. Try the request again later.  
• The source object is in the process of being deleted. |
| 404  | Not Found | One of these:  
• The specified multipart upload does not exist. Either the upload ID is invalid, or the multipart upload was aborted or completed.  
In the case of completion, if the part being uploaded is a |
<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>replacement for an existing part, the existing part is used for completing the multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified bucket does not exist.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict</td>
<td>The specified multipart upload was completed or aborted while the upload part copy operation was in progress.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>One of these:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request included an x-amz-copy-source-if-match header, and the ETag for the specified source object or object version does not match any of the values specified by the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request included an x-amz-copy-source-if-none-match header, and the ETag for the specified source object or object version matched a value specified by the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request included an x-amz-copy-source-if-modified-since header, and the specified source object or object version was not modified after the date and time specified by the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request included an x-amz-copy-source-if-unmodified-since header, and the specified source object or object version was modified after the date and time specified by the header.</td>
</tr>
<tr>
<td>412</td>
<td>Precondition Failed</td>
<td>An internal error occurred. If this error persists, contact your tenant administrator.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The source object exists but cannot be read.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HCP is temporarily unable to handle the request, probably due to system overload, maintenance, or upgrade.</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
<td>Try the request again, gradually increasing the delay between each successive attempt.</td>
</tr>
<tr>
<td>Code</td>
<td>Meaning</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this error persists, contact your tenant administrator.</td>
</tr>
</tbody>
</table>

**Examples (PUT object part copy)**

You are here:

This section of the Help contains sample PUT requests for uploading a part of a multipart upload by copying data from an existing object.

**Example 1: Conditionally creating a part from an object**

You are here:

Here’s a sample PUT request that conditionally uploads a part of the multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket by copying the data from an existing object named acctg/AcctgRR-Summary in the same bucket.

For this example, assume that part 6 of the multipart upload has already been uploaded and has an ETag of 7914d874df2c1d55cfab4fa82088ff56. The request directs HCP to create a new part 6 from the copied data only if the ETag of the copied data is different from the ETag of the data previously uploaded for part 6.

The example shows the response headers HCP returns while versioning is enabled for the bucket.

Request with s3curl command line

```
./s3curl.pl --id=lgreen --copysrc=finance/acctg/AcctgRR-Summary -- -k
  "https://finance.europe.hcp.example.com/acctg/LinksAndRegulations.pdf
   ?uploadId=94837746087105&partNumber=6"
  -H "x-amz-copy-source-if-none-match:7914d874df2c1d55cfab4fa82088ff56"
  -H "x-hcp-pretty-print: true"
```

Request headers

```
PUT /acctg/RulesAndRegulations.pdf?uploadId=94837746087105&partNumber=6 HTTP/1.1
Host: finance.europe.hcp.example.com
```
Example 2: Creating a part from part of an object

You are here:

Here's a sample PUT request that uploads a part of the multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket by copying part of the data for an existing object named AcctgBestPractices.doc in the same bucket.

The example shows the response headers HCP returns while versioning is enabled for the bucket.

Request with s3curl command line

```
./s3curl.pl --id=lgreen --copysrc=finance/AcctgBestPractices.doc -- -k
  "https://finance.europe.hcp.example.com/RulesAndRegulations.pdf?uploadId=94837746087105&partNumber=5"
  -H "x-amz-copy-source-range: bytes=2800000-9499999"
  -H "x-hcp-pretty-print: true"
```


Updated: Mon, 16 Dec 2019 08:34:08 GMT
Powered by
Request headers

PUT /acctg/RulesAndRegulations.pdf?uploadId=94837746087105&partNumber=5 HTTP/1.1
Host: finance.europe.hcp.example.com
Date: Thu, 10 Jan 2019 17:19:26 +0000
Authorization: AWS bGdyZWVu:a0wC/DuV+w2Agq6Fagnle6cShaU=
x-amz-copy-source: finance/AcctgBestPractices.doc
x-amz-copy-source-range: bytes=2800000-9499999
x-hcp-pretty-print: true

Response headers

HTTP/1.1 200 OK
Date: Thu, 10 Jan 2019 17:19:26 GMT
ETag: "bbe438b2a0376f08dc37867a82ea87e6"
x-amz-version-id: 9487113833377
x-amz-copy-source-version-id: 94860356828929
Content-Type: application/xml;charset=UTF-8
x-amz-server-side-encryption: None
Transfer-Encoding: chunked

Response body

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <LastModified>2017-02-23T17:19:26.000Z</LastModified>
  <ETag>"bbe438b2a0376f08dc37867a82ea87e6"</ETag>
</CopyPartResult>

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

Listing the parts of a multipart upload

You are here:

You use the HTTP GET method with the uploadId query parameter to list the parts that have been uploaded for an in-progress multipart upload. An in-progress multipart upload is one that has been initiated but not yet completed or aborted. You cannot list the parts of a completed or aborted multipart upload.

To list the parts of a multipart upload, you must be an authenticated user. You need to use either the same user account as the account used in the request to initiate the multipart upload or the user account for the owner of the bucket where the multipart upload was initiated. Additionally, you need read permission for the bucket.
You use the uploadId query parameter to specify the upload ID of the multipart upload for which you're requesting the part listing. If the upload ID you specify doesn't match the upload ID of an in-progress multipart upload or isn't associated with the object specified in the request, HCP returns a 404 (Not Found) status code and does not return a part listing.

A part listing is returned in an XML response body. The parts are listed in numeric order. For each part, the listing contains the part number, the date and time when the part was uploaded, the ETag for the part, and the size of the part.

A part listing does not include parts that are in the process of being uploaded or parts that have been replaced.

Request line (GET object list parts)

You are here:

Depending on whether the bucket name is included in the hostname in the S3 compatible request, a request to list the parts of a multipart upload has either of these formats:

• With the bucket name included in the hostname:

GET /object-name?uploadId=upload-id[&query-parameters] HTTP/1.1

• With the bucket name following the hostname:

GET /bucket-name/object-name?uploadId=upload-id[&query-parameters] HTTP/1.1

The uploadId query parameter is case sensitive.

query-parameters can be none, one, or more of:

• max-parts

• part-number-marker

• encoding-type

For information about these query parameters, see Limiting the part listing.
Limiting the part listing

You are here:

You can use the max-parts, part-number-marker, and encoding-type query parameters to limit the parts included in a part listing.

max-parts query parameter

By default, a part listing for a multipart upload can include at most one thousand parts. However, you can use the max-parts query parameter to specify a smaller maximum number of parts to be included in the returned listing. If more than the maximum number of parts satisfy the criteria for a request, you can use query parameters to retrieve the parts in groups.

max-parts parameter rules

Valid values for the max-parts query parameter are integers in the range zero through one thousand. If you specify an integer greater than one thousand, HCP returns a 400 (Invalid Argument) status code and does not return a part listing.

The max-parts parameter name is case sensitive.

part-number-marker query parameter

By default, a part listing starts with the lowest-numbered part of the multipart upload. You can use the part-number-marker query parameter to request a listing that starts with a part with a higher number. When you include this query parameter in the request for a part listing, the returned listing starts with the part with the next higher number than the value you specified for the parameter.

part-number-marker parameter rules

Valid values for the part-number-marker query parameter are integers in the range 0 (zero) through 10,000.

The part-number-marker parameter name is case sensitive.

encoding-type parameter

You use the encoding-type query parameter to request the S3 compatible API to encode the response. You can also use this query parameter to specify the encoding method to use.

encoding-type parameter rules

The key for an object can contain any Unicode character. Some characters, such as those with an ASCII value from 0 to 10, cannot be parsed by XML 1.0 parsers. For these characters, you can add the encoding-type query parameter to request the S3 compatible API to encode the keys in the response.

The encoding-type parameter name is case sensitive.
IsTruncated and NextPartNumberMarker response body elements

The response body for a part listing request includes the IsTruncated element. If the multipart upload has any parts with higher numbers than the last part included in the returned listing, the value of the IsTruncated element is true. If the returned listing includes the highest-numbered part, the value of the IsTruncated element is false.

If the value of the IsTruncated element is true, the response body includes the NextPartNumberMarker element. The value of this element is the part number of the last part included in the returned listing. This is the number to use as the value of the part-number-marker query parameter in a request to retrieve the next group of parts of the multipart upload.

Paging through a part listing

By using the max-parts and part-number-marker query parameters in conjunction with each other, you can page through a multipart upload listing. For example, suppose a multipart upload with object name acctg/AcctgAtExampleCorp-Introduction.mov and upload ID 9 has parts numbered 1, 2, 3, 4, 5, 6, 7, and 8. To page through a listing of these parts three parts at a time, you would use a sequence of requests like this:

1. Request line:
   
   GET /acctg/AcctgAtExampleCorp-Introduction.mov?uploadId=9&max-parts=3 HTTP/1.1

   The response body lists parts 1, 2, and 3 and includes these elements:

   <IsTruncated>true</IsTruncated>
   <NextPartNumberMarker>3</NextPartNumberMarker>

2. Request line:
   
   GET /acctg/AcctgAtExampleCorp-Introduction.mov?uploadId=9&max-parts=3
   &part-number-marker=3 HTTP/1.1

   The response body lists parts 4, 5, and 6 and includes these elements:

   <IsTruncated>true</IsTruncated>
   <NextPartNumberMarker>6</NextPartNumberMarker>

3. Request line:
   
   GET /acctg/AcctgAtExampleCorp-Introduction.mov?uploadId=9&max-parts=3
   &part-number-marker=6 HTTP/1.1

   The response body lists parts 7 and 8 and includes this element:

   <IsTruncated>false</IsTruncated>

Trademarks and Legal Disclaimer

https://knowledge.hitachivantara.com/Documents/Storage/Content_Platform/8.2/Developer_documentation/Using_the_Hitachi...
## Request headers (GET object list parts)

You are here:

The table below describes the headers you can use in a request to list the parts of a multipart upload.

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>See Authentication.</td>
<td>Yes</td>
</tr>
<tr>
<td>Date</td>
<td>See Common request headers.</td>
<td>Date or x-amz-date</td>
</tr>
<tr>
<td>Host</td>
<td>See Common request headers.</td>
<td>Yes</td>
</tr>
<tr>
<td>x-amz-date</td>
<td>See Common request headers.</td>
<td>x-amz-date or Date</td>
</tr>
</tbody>
</table>
| x-hcp-pretty-print | Optionally, requests that the XML response body be formatted for readability. Valid values are:  
  • true — Format the XML response body for readability.  
  • false — Do not apply any special formatting to the XML response body.  
  The default is false.  
  The values true and false are not case sensitive. | No               |

---

## Response headers (GET object list parts)

You are here:

The table below describes the headers returned in response to a successful request to list the parts of a multipart upload.
<table>
<thead>
<tr>
<th>Response header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the response body. For a request to list the parts of a multipart upload, the value of this header is always application/xml;charset=UTF-8.</td>
</tr>
<tr>
<td>Date</td>
<td>See Common response headers.</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>Indicates that HCP could not determine the size of the response body before formulating the response. For a request to list the parts of a multipart upload, the value of this header is always chunked.</td>
</tr>
</tbody>
</table>
| x-amz-abort-date  | Specifies the date and time when the multipart upload will be automatically aborted if it has not already been completed. The date and time are expressed in this format: DDD, ddMMdd yyyy hh:mm:ss GMT  
For example: Sat, 25 Mar 2017 17:19:26 GMT  
For information about automatic abort of multipart uploads, see Considerations for working with multipart uploads. |

**Response body (GET object list parts)**

You are here:

HCP returns the list of parts of a multipart upload in an XML response body, in this format:

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Bucket>bucket-name</Bucket>
  <Key>object-name</Key>
  <UploadId>upload-id</UploadId>
  <Initiator>
    <ID>initiator-user-id</ID>
    <DisplayName>initiator-username</DisplayName>
  </Initiator>
</ListPartsResult>
```
<Owner>
  <ID>owner-user-id</ID>
  <DisplayName>owner-username</DisplayName>
</Owner>
<StorageClass>STANDARD</StorageClass>
<PartNumberMarker>part-number-marker</PartNumberMarker>
<NextPartNumberMarker>number-of-last-part-in-listing</NextPartNumberMarker>
<MaxParts>maximum-parts-to-list</MaxParts>
<IsTruncated>true|false</IsTruncated>

Format for a part:
<Part>
  <PartNumber>part-number</PartNumber>
  <LastModified>date-and-time-last-modified</LastModified>
  <ETag>"etag"</ETag>
  <Size>part-size</Size>
</Part>

The table below describes the XML elements in the response body returned in response to a request to list the parts of a multipart upload. The elements are listed in alphabetical order.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bucket</td>
<td>Child of the ListPartsResult element. The Bucket element specifies the name of the bucket where the multipart upload is being performed.</td>
</tr>
<tr>
<td>DisplayName</td>
<td>Child of the Initiator or Owner element. If the multipart upload initiator or object owner is identified by an HCP user account, the value of the DisplayName element is the username for that account. If the multipart upload initiator or object owner is identified by an AD user account, the value of the DisplayName element is the username of that account followed by an at sign (@) and the AD domain name.</td>
</tr>
<tr>
<td>ETag</td>
<td>Child of the Part element.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>The ETag element specifies the ETag of the applicable part. For information about ETags, see <a href="https://knowledge.hitachivantara.com/Documents/Storage/Content_Platform/8.2/Developer_documentation/Using_the_Hitachi%E2%80%A6">Uploading a part of a multipart upload</a>.</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>Child of the Initiator or Owner element. If the multipart upload initiator or object owner is identified by an HCP user account, the value of the ID element is the user ID for that account. If the multipart upload initiator or object owner is identified by an AD user account, the value of the ID element is the SID for that account.</td>
</tr>
<tr>
<td>Initiator</td>
<td>Child of the ListPartsResult element and container for the DisplayName and ID elements. The Initiator element identifies the user who initiated the multipart upload.</td>
</tr>
<tr>
<td>IsTruncated</td>
<td>Child of the ListPartsResult element. The IsTruncated element indicates whether the returned part listing includes the last part that satisfies the request criteria. Possible values are: • <code>true</code> — The returned listing includes the last part that satisfies the request criteria. • <code>false</code> — The returned listing does not include the last part that satisfies the request criteria.</td>
</tr>
<tr>
<td>Key</td>
<td>Child of the ListPartsResult element. The Key element specifies the name of the object being created by the multipart upload.</td>
</tr>
<tr>
<td>LastModified</td>
<td>Child of the Part element.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>The LastModified element specifies the date and time when the applicable part was last modified, in Greenwich Mean Time (GMT).</td>
<td></td>
</tr>
<tr>
<td>The date and time are expressed in this format:</td>
<td></td>
</tr>
<tr>
<td>yyyy-MM-ddTHH:mm:ss.SSSZ</td>
<td></td>
</tr>
<tr>
<td>For example:</td>
<td></td>
</tr>
<tr>
<td>2019-02-18T19:46:03.856Z</td>
<td></td>
</tr>
<tr>
<td>ListPartsResult</td>
<td>Root element.</td>
</tr>
<tr>
<td>MaxParts</td>
<td>Child of the ListPartsResult element.</td>
</tr>
<tr>
<td>The MaxParts element specifies the value of the max-parts query parameter included in the request. If the request did not include the max-parts query parameter, the value of the MaxParts element is 1000.</td>
<td></td>
</tr>
<tr>
<td>NextPartNumberMarker</td>
<td>Child of the ListPartsResult element.</td>
</tr>
<tr>
<td>The NextPartNumberMarker element specifies the number of the last part included in the returned part listing. If the returned listing doesn't include any parts, the value of the NextPartNumberMarker element is 0 (zero).</td>
<td></td>
</tr>
<tr>
<td>If the returned listing is truncated, you can use the value of the NextPartNumberMarker element as the value of the part-number-marker query parameter in a new request to retrieve the next set of parts that satisfy the request criteria.</td>
<td></td>
</tr>
<tr>
<td>Owner</td>
<td>Child of the ListPartsResult element and container for the DisplayName and ID elements.</td>
</tr>
<tr>
<td>The Owner element identifies the user who will own the object created by the multipart upload.</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Part</td>
<td>Child of the ListPartsResult element and container for the elements that describe a part. The response body contains one Part element for each part that has been uploaded for the multipart upload.</td>
</tr>
<tr>
<td>PartNumber</td>
<td>Child of the Part element. The PartNumber element specifies the part number of the applicable part.</td>
</tr>
<tr>
<td>PartNumberMarker</td>
<td>Child of the ListPartsResult element. The PartNumberMarker element specifies the value of the part-number-marker query parameter included in the request. If the request did not include the part-number-marker query parameter, the value of the PartNumberMarker element is 0 (zero).</td>
</tr>
<tr>
<td>Size</td>
<td>Child of the Part element. The Size element specifies the size, in bytes, of the applicable part.</td>
</tr>
<tr>
<td>StorageClass</td>
<td>Child of the ListPartsResult element. The value of the StorageClass element is always STANDARD.</td>
</tr>
<tr>
<td>UploadId</td>
<td>Child of the ListPartsResult element. The UploadId element specifies the upload ID of the multipart upload.</td>
</tr>
</tbody>
</table>

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.
The table below describes HTTP status codes that can be returned in response to a request to list the parts of a multipart upload. For more information about HTTP status codes and the error codes that can accompany them, see [Error codes](#).

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>HCP successfully listed the parts of the multipart upload.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>Possible reasons include that a query parameter has an invalid value.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request specifies anonymous access. Only an authenticated user can list the parts of a multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request are invalid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request do not match the credentials used to initiate the multipart upload or the credentials for the owner of the bucket where the multipart upload is being performed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You do not have permission to list the parts of a multipart upload in the specified bucket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The S3 compatible API is currently disabled for the specified bucket.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>One of these:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified multipart upload does not exist. Either the upload ID is invalid, or the multipart upload was aborted or completed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified bucket does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
<td>An internal error occurred.</td>
</tr>
<tr>
<td>Code</td>
<td>Meaning</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this error persists, contact your tenant administrator.</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
<td>HCP is temporarily unable to handle the request, probably due to system overload, maintenance, or upgrade. Try the request again, gradually increasing the delay between each successive attempt. If this error persists, contact your tenant administrator.</td>
</tr>
</tbody>
</table>

**Example: Listing the parts of a multipart upload**

You are here:

Here’s a sample GET request for a listing of the parts of an in-progress multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket. The request uses these query parameters in addition to uploadId:

- `part-number-marker=1` — Starts the list with the first part with a part number greater than one
- `max-parts=3` — Lists a maximum of three parts

The example assumes that parts one, two, three, five, and six have already been uploaded.

Request with s3curl command line

```
./s3curl.pl --id=lgreen -- -k
    "https://finance.europe.hcp.example.com/acctg/RulesAndRegulations.pdf?uploadId=94837746087105&part-number-marker=1&max-parts=3"
    -H "x-hcp-pretty-print: true"
```

Request headers

```
GET /acctg/RulesAndRegulations.pdf?uploadId=94837746087105
    &part-number-marker=1&max-parts=3 HTTP/1.1
Host: finance.europe.hcp.example.com
Date: Thu, 10 Jan 2019 17:19:26 +0000
Authorization: AWS bGdyZWVu:Ey256+39Nz19jRw7chPIOJwD8m4=
x-hcp-pretty-print: true
```
Response headers

HTTP/1.1 200 OK
Date: Thu, 10 Jan 2019 17:19:26 GMT
x-amz-abort-date: Sat, 25 Feb 2017 17:19:26 GMT
Content-Type: application/xml;charset=UTF-8
Transfer-Encoding: chunked

Response body

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Bucket>finance</Bucket>
  <Key>acctg/RulesAndRegulations.pdf</Key>
  <UploadId>94837746087105</UploadId>
  <Initiator>
    <ID>835be4b1-8f84-407b-8084-b9329beadf9b</ID>
    <DisplayName>lgreen</DisplayName>
  </Initiator>
  <Owner>
    <ID>835be4b1-8f84-407b-8084-b9329beadf9b</ID>
    <DisplayName>lgreen</DisplayName>
  </Owner>
  <StorageClass>STANDARD</StorageClass>
  <PartNumberMarker>1</PartNumberMarker>
  <NextPartNumberMarker>5</NextPartNumberMarker>
  <MaxParts>3</MaxParts>
  <IsTruncated>true</IsTruncated>
  <Part>
    <PartNumber>2</PartNumber>
    <LastModified>2017-02-23T17:19:26.000Z</LastModified>
    <ETag>97ede335e69d08bb3cfbbd974bae66e4</ETag>
    <Size>6800000</Size>
  </Part>
  <Part>
    <PartNumber>3</PartNumber>
    <LastModified>2017-02-23T17:19:26.000Z</LastModified>
    <ETag>a39ebe49948a821cab2c4d7d04f9b1c3</ETag>
    <Size>6800000</Size>
  </Part>
  <Part>
    <PartNumber>5</PartNumber>
    <LastModified>2017-02-23T17:19:26.000Z</LastModified>
  </Part>
</ListPartsResult>
Completing a multipart upload

You are here:

You use the HTTP POST method with the uploadId query parameter to complete a multipart upload.

To complete a multipart upload, you must be an authenticated user. You need to use the same user account as the account used in the request to initiate the multipart upload. Additionally, you need write permission for the bucket where the multipart upload was initiated.

You use the uploadId query parameter to specify the upload ID of the multipart upload you're completing. If the upload ID you specify doesn't match the upload ID of an in-progress multipart upload or isn't associated with the object specified in the request, HCP returns a 404 (Not Found) status code and does not complete the multipart upload.

To specify the list of parts to use when completing a multipart upload, you use an XML request body. You identify each part by its part number and the ETag returned when the part was uploaded. The list of parts must be in ascending order by part number. For the format of the request body, see Request body (POST object complete multipart upload).

Only the parts listed in the request to complete a multipart upload are used as data for the resulting object. Parts that were uploaded for the multipart upload but that are not listed in the request are automatically deleted after the request is processed.

To be completed, a multipart upload must have at least one part.

When you complete a multipart upload, you become the owner of the resulting object. For information about object ownership, see Object owners.

If a request to complete a multipart upload includes any custom metadata (x-amz-meta-) headers, HCP returns a 400 (Bad Request) status code and does not complete the multipart upload.

HCP ignores ACL headers in requests to complete a multipart upload.

Conditionally completing a multipart upload

If versioning is enabled for a bucket, you can use a multipart upload to create a new version of an object in the bucket. However, between the time you initiated the multipart upload and the time processing of the request to complete the multipart upload finishes, another user may have stored a different new version of the object (by using either a single
PUT object request or a multipart upload).

To ensure that the version of the object that was most recent when you initiated the multipart upload is still the most recent version, you can include the If-Match header in the request to complete the multipart upload. If the value you specify for the If-Match header matches the ETag of the most recent version of the object, HCP completes the multipart upload. If the value doesn't match the ETag or if the object doesn't exist, HCP returns an error response body and does not complete the multipart upload.

Note: The If-Match header is not part of the Amazon S3 API.

Complete multipart upload request result

Processing a request to complete a multipart upload can take several minutes. When processing starts, HCP returns a 200 (OK) status code. While processing continues, HCP periodically sends white-space characters to prevent the connection from timing out.

If a request to complete a multipart upload succeeds, the response body contains information about the resulting object. If the request fails, the response body contains error information. Therefore, to determine the outcome of a request to complete a multipart upload, you need to check the content of the response body in addition to the status code.

These errors can cause a request to complete a multipart upload to fail after HCP returns the 200 (OK) status code:

• The list of parts in the request body includes a part, other than the last part, that is smaller than one megabyte.

• The list of parts in the request body includes one or more part numbers that are not the number of an uploaded part.

• The ETag associated with one or more part numbers listed in the request body is not the ETag returned for the part with the specified part number.

• The parts in the request body are not listed in ascending order.

• The request includes the If-Match request header, and either the value specified by the header does not match the ETag of the most recent version of the specified object, or the specified object does not already exist.

For information about error response bodies, see Error response body.

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.

Request line (POST object complete multipart upload)

You are here:

Depending on whether the bucket name is included in the hostname in the S3 compatible request, a request to complete
A multipart upload has either of these formats:

- With the bucket name included in the hostname:
  
  POST /object-name?uploadId=upload-id HTTP/1.1

- With the bucket name following the hostname:
  
  POST /bucket-name/object-name?uploadId=upload-id HTTP/1.1

The uploadId query parameter is case sensitive.

Request body (POST object complete multipart upload)

You are here:

For the content of a complete multipart upload request body, you use XML in this format:

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUpload
    xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  Include one Part element for each part to be used for the completed object.
  <Part>
    <PartNumber>part-number</PartNumber>
    <ETag>etag</ETag>
  </Part>
  .
  .
  .
</CompleteMultipartUpload>
```

The table below describes the XML elements in a complete multipart upload request body. The elements are listed in alphabetical order.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompleteMultipartUpload</td>
<td>Root element. This must be the first element in the complete multipart upload request body. The CompleteMultipartUpload element can optionally include this XML namespace specification:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>xmlns</td>
<td><code>http://s3.amazonaws.com/doc/2006-03-01/</code></td>
</tr>
<tr>
<td>ETag</td>
<td>Child of the Part element. The ETag element specifies the ETag of the applicable part.</td>
</tr>
<tr>
<td>Part</td>
<td>Child of the CompleteMultipartUpload element and container for the elements that describe a part. The Part element identifies a part that has been uploaded for the multipart upload.</td>
</tr>
<tr>
<td>PartNumber</td>
<td>Child of the Part element. The PartNumber element specifies the part number of the applicable part.</td>
</tr>
</tbody>
</table>

**Request headers (POST object complete multipart upload)**

You are here:

The table below describes the headers you can use in a request to complete a multipart upload.

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>See <a href="#">Authentication</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Length</td>
<td>Specifies the size, in bytes, of the request body.</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the request body. The only valid values are application/xml and text/xml.</td>
<td>Yes</td>
</tr>
<tr>
<td>Date</td>
<td>See <a href="#">Common request headers</a></td>
<td>Date or x-amz-date</td>
</tr>
<tr>
<td>Request header</td>
<td>Description</td>
<td>Required</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>Host</td>
<td>See Common request headers.</td>
<td>Yes</td>
</tr>
<tr>
<td>If-Match</td>
<td>Specifies a value for comparison with the ETag of the most recent version of the object specified in the request. If the specified value matches the ETag, HCP continues processing the request. If the specified value doesn't match the ETag or the specified object doesn't exist, HCP returns an error response body and does not complete the multipart upload. The value of this header can be any string of one or more characters.</td>
<td>No</td>
</tr>
<tr>
<td>x-amz-date</td>
<td>See Common request headers.</td>
<td>x-amz-date or Date</td>
</tr>
</tbody>
</table>
| x-hcp-pretty-print | Optionally, requests that the XML response body be formatted for readability. Valid values are:  
• true — Format the XML response body for readability.  
• false — Do not apply any special formatting to the XML response body.  
The default is false.  
The values true and false are not case sensitive. | No      |
| x-hcp-retention | Specifies the retention value for the object being stored. This value can be a fixed date, an offset, a retention class, or a special value. | No      |
| x-hcp-retentionhold | Specifies whether the object is on hold. This value can be either true or false. | No      |
The table below describes the headers returned in response to a successful request to complete a multipart upload.

<table>
<thead>
<tr>
<th>Response header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the Internet media type of the response body. For a request to complete a multipart upload, the value of this header is always application/xml;charset=UTF-8.</td>
</tr>
<tr>
<td>Date</td>
<td>See Common response headers.</td>
</tr>
<tr>
<td>x-amz-server-side-encryption</td>
<td>Specifies whether objects stored in HCP are encrypted. Possible values are: • If objects are encrypted, AES256 • If objects are not encrypted, None</td>
</tr>
<tr>
<td>x-amz-version-id</td>
<td>Specifies the version ID of the object created by the multipart upload. This header is returned only while versioning is enabled for the bucket.</td>
</tr>
</tbody>
</table>

Response body (POST object complete multipart upload)

HCP returns information about the object that results from a successful complete multipart upload request in an XML response body, in this format:

```xml
<?xml version='1.0' encoding='UTF-8'?>
  <Location>object-url</Location>
  <Bucket>bucket-name</Bucket>
  <Key>object-name</Key>
  <ETag>"etag"</ETag>
</CompleteMultipartUploadResult>
```

The table below describes the XML elements in the response body returned in response to a request to complete a multipart upload. The elements are listed in alphabetical order.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bucket</td>
<td>Child of the CopyPartResult element. The Bucket element specifies the bucket where the object was created.</td>
</tr>
<tr>
<td>CompleteMultipart UploadResult</td>
<td>Root element.</td>
</tr>
<tr>
<td>ETag</td>
<td>Child of the CopyPartResult element. The ETag element specifies the ETag for the object.</td>
</tr>
<tr>
<td>Key</td>
<td>Child of the CopyPartResult element. The Key element specifies the name of the object.</td>
</tr>
<tr>
<td>Location</td>
<td>Child of the CopyPartResult element. The Location element specifies the complete URL for the object.</td>
</tr>
</tbody>
</table>

**HTTP status codes (POST object complete multipart upload)**

You are here:

The table below describes HTTP status codes that can be returned in response to a request to complete a multipart upload. For more information about HTTP status codes and the error codes that can accompany them, see Error codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>HCP successfully started processing the request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To determine whether the multipart upload was successfully completed, check whether HCP returned a response body that</td>
</tr>
<tr>
<td>Code</td>
<td>Meaning</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>identifies the resulting multipart object or an error response body.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request does not include a Content-Type header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The XML in the request body is malformed or contains an invalid value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request includes a custom metadata (x-amz-meta-) header.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request specifies anonymous access. Only an authenticated user can</td>
</tr>
<tr>
<td></td>
<td></td>
<td>complete a multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request are invalid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request do not match the credentials</td>
</tr>
<tr>
<td></td>
<td></td>
<td>used to initiate the multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You do not have permission to complete multipart uploads in the specified</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bucket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The S3 compatible API is currently disabled for the specified bucket.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>One of these:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified multipart upload does not exist. Either the upload ID is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalid, or the multipart upload was aborted or already completed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified bucket does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
<td>An internal error occurred.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this error persists, contact your tenant administrator.</td>
</tr>
<tr>
<td>Code</td>
<td>Meaning</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
<td>HCP is temporarily unable to handle the request, probably due to system overload, maintenance, or upgrade. Try the request again, gradually increasing the delay between each successive attempt. If this error persists, contact your tenant administrator.</td>
</tr>
</tbody>
</table>

**Example: Completing a multipart upload**

You are here:

Here’s a sample POST request that completes a multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket. In this example, the retention value of the object is set to five days past the time when the object was stored. The request body is in a file named Acctg-RulesAndRegulations.xml.

The request is being made while versioning is enabled for the bucket.

**Request body**

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUpload>
  <Part>
    <PartNumber>1</PartNumber>
    <ETag>4a01f8acf3353b0a7e1c03a0e34f63e4</ETag>
  </Part>
  <Part>
    <PartNumber>2</PartNumber>
    <ETag>97ede335e69d08bb3cfbbd974bae68e4</ETag>
  </Part>
  <Part>
    <PartNumber>3</PartNumber>
    <ETag>a39ebe49948a821cab2c4d7d04f9b1c3</ETag>
  </Part>
  <Part>
    <PartNumber>4</PartNumber>
    <ETag>a7b8192c98da2c13a034980376b6c809</ETag>
  </Part>
</CompleteMultipartUpload>
```


Updated: Mon, 16 Dec 2019 08:34:08 GMT

Powered by
Request with s3curl command line

```
./s3curl.pl --id=lgreen --post=Acctg-RulesAndRegulations.xml -- -k
  "https://finance.europe.hcp.example.com...egulations.pdf
?uploadId=94837746087105" -H "x-hcp-retention: A+5d" -H "x-hcp-pretty-print: true"
```

Request headers

```
POST /acctg/RulesAndRegulations.pdf?uploadId=94837746087105 HTTP/1.1
Host: finance.europe.hcp.example.com
Date: Thu, 10 Jan 2019 17:19:26 +0000
Authorization: AWS bGdyZWVu:y4wahRZz+oLpASXkNeBy9LQAJsE=
x-hcp-retention: A+5d
x-hcp-pretty-print: true
Content-Length: 884
```

Response headers

```
HTTP/1.1 200 OK
Date: Thu, 10 Jan 2019 17:19:26 GMT
x-amz-server-side-encryption: None
x-amz-version-id: 94874564091777
Content-Type: application/xml;charset=ISO-8859-1
ETag: "e879d0c42d032f7b55d2be82825071d6-6"
Content-Length: 401
```

Response body

```
<?xml version='1.0' encoding='UTF-8'?><CompleteMultipartUploadResult
  xmlns="http://s3.amazonaws.com/doc/2006-03-01/*">
  <Location>https://finance.europe.hcp.example.com/hs3/acctg
   /RulesAndRegulations.pdf</Location>
  <Bucket>finance</Bucket>
  <Key>acctg/RulesAndRegulations.pdf</Key>
  <ETag>&quot;e879d0c42d032f7b55d2be82825071d6-6&quot;</ETag>
</CompleteMultipartUploadResult>
```
Aborting a multipart upload

You use the HTTP DELETE method with the uploadId query parameter to abort a multipart upload.

To abort a multipart upload, you must be an authenticated user. You need to use the same user account as the account used in the request to initiate the multipart upload or the user account for the owner of the bucket where the multipart upload was initiated. Additionally, you need delete permission for the bucket.

You use the uploadId query parameter to specify the upload ID of the multipart upload you're aborting. If the upload ID you specify doesn't match the upload ID of an in-progress multipart upload or isn't associated with the object specified in the request, HCP returns a 404 (Not Found) status code and does not abort the multipart upload.

You can abort any in-progress multipart upload, including multipart uploads that are in the process of being completed. You cannot abort a completed multipart upload.

When you abort a multipart upload, HCP stops uploading any parts that are in the process of being uploaded. You cannot upload additional parts for or list the parts of an aborted multipart upload.

Parts of an aborted upload that were already uploaded or that were in the process of being uploaded when you issued the abort request are automatically deleted.

Request line (DELETE object abort multipart upload)

Depending on whether the bucket name is included in the hostname in the S3 compatible request, a request to abort a multipart upload has either of these formats:

• With the bucket name included in the hostname:
  
  DELETE /object-name?uploadId=upload-id HTTP/1.1

• With the bucket name following the hostname:
  
  DELETE /bucket-name/object-name?uploadId=upload-id HTTP/1.1
The uploadId query parameter is case sensitive.

<table>
<thead>
<tr>
<th>Request header</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>See Authentication.</td>
<td>Yes</td>
</tr>
<tr>
<td>Date</td>
<td>See Common request headers.</td>
<td>Date or x-amz-date</td>
</tr>
<tr>
<td>Host</td>
<td>See Common request headers.</td>
<td>Yes</td>
</tr>
<tr>
<td>x-amz-date</td>
<td>See Common request headers.</td>
<td>x-amz-date or Date</td>
</tr>
</tbody>
</table>

**Note:** HCP ignores query parameters it doesn't recognize. Therefore, if you use the wrong case for the uploadId query parameter in a request to abort a multipart upload, the request becomes a DELETE request for the object named in the request.

If the named object exists, HCP deletes it and returns a 204 (No Content) status code, indicating that the request was successful. HCP also returns a 204 status code if the named object does not exist. Because HCP returns a 204 status code for a successful abort of a multipart upload, you cannot tell from the status code alone whether a request to abort a multipart upload had the intended result.
Response header (DELETE object abort multipart upload)

You are here:

The table below describes the headers returned in response to a successful request to abort a multipart upload.

<table>
<thead>
<tr>
<th>Response header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>See Common response headers.</td>
</tr>
</tbody>
</table>

HTTP status codes (DELETE object abort multipart upload)

You are here:

The table below describes HTTP status codes that can be returned in response to a request to complete a multipart upload. For more information about HTTP status codes and the error codes that can accompany them, see Error codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>No Content</td>
<td>HCP successfully aborted the multipart upload.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>Possible reasons include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The request specifies anonymous access. Only an authenticated user can abort a multipart upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request are invalid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The credentials provided with the request do not match the credentials used to initiate the multipart upload or the credentials for the owner of the bucket where the multipart upload is being performed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You do not have permission to abort multipart uploads in the specified bucket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The S3 compatible API is currently disabled for the specified bucket.</td>
</tr>
<tr>
<td>Code</td>
<td>Meaning</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>One of these:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified multipart upload does not exist. Either the upload ID is invalid, or the multipart upload was already aborted or completed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The specified bucket does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
<td>An internal error occurred.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this error persists, contact your tenant administrator.</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
<td>HCP is temporarily unable to handle the request, probably due to system overload, maintenance, or upgrade. Try the request again, gradually increasing the delay between each successive attempt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this error persists, contact your tenant administrator.</td>
</tr>
</tbody>
</table>

Example: Aborting a multipart upload

You are here:

Here’s a sample DELETE request that aborts a multipart upload for an object named acctg/RulesAndRegulations.pdf in the finance bucket.

Request with s3curl command line

```
./s3curl.pl --id=lgreen --delete -- -k
 *https://finance.europe.hcp.example.com...egulations.pdf
 ?uploadId=94874755807297"
```

Request headers

```
DELETE /acctg/RulesAndRegulations.pdf?uploadId=94874755807297 HTTP/1.1
Host: finance.europe.hcp.example.com
Date: Thu, 10 Jan 2019 17:19:26 +0000
Authorization: AWS bGdyZWWVv:KxEygWb4ay0jHtOrijBE5iGmls=
```
Response headers

HTTP/1.1 204 No Content
Date: Thu, 10 Jan 2019 17:19:26 GMT

Trademarks and Legal Disclaimer

© 2015, 2019 Hitachi Vantara Corporation. All rights reserved.