文档类别

杭 州 海 康 威 视 科 技 有 限 公 司

文档编号

海康云存储 V2.0 对象存储
REST 接口文教卫使用方式说明

密级级别：[内部公开]
生效时间： 2015 年 12 月 11 日
保密期：无

海康云存储 V2.0 对象存储
REST 接口文教卫使用方式说明

密级级别：[内部公开]
生效时间： 2015 年 12 月 11 日
保密期：无

目录

1. 前言

海康云存储 2.0 对象存储服务(简称 HikCStor2.0)，是海康云对外提供的海量、安全、低成本、高可靠的云存储服务。用户可以通过简单的 REST 接口，在任何时间、任何地点上进行上传和下载数据。同时，HikCStor 同时还提供 JAVA、C/C++语言的 SDK，简化用户的编程。基于 HikCStor，用户可以搭建出各种大规模数据服务。

后面概念中海康云存储与 HikCStor 含义相同。

2. 简介

2.1 编写目的

本文档描述了如何使用海康对象存储服务进行 bucket 的管理、object 的管理、权限管理、容量管理等。并对一些常规操作进行特例化说明，如文件上传、文件数据提取、存储周期设置、用户存储空间管理等。

2.2 编写背景

该文档主要面向使用海康对象存储服务进行数据存储的项目开发人员以及测试人员。

3. 相关概念

3.1 Access Key

用户在使用云存储服务前，需先注册用户名，该用户名必须在海康云存储中唯一，在注册成功之后，云存储会给该用户自动生成一个 Access Key 和一个 Secret Key，用户在使用协议对接之前，需先获取用户对应的 Access Key 与 Secret Key，该 Access Key 会在所有的操作中被使用，并且与 Secret Key 一起，经过特别不可逆运算之后，得到一个签名字符串，该字符串与 Access key 一起以明文形式传输至服务器。

云存储注册时用户名格式目前只支持小写字母、大写字母、数字、”_”、”-”、”@”、”.”符号。并且只能以字母或者数字开头。

3.2 Secret Key

用户在注册用户名时与 Access Key 一起生成，该值与 Access Key 一起使用，用户需确保该 Access Key 与 Secret Key 的安全性。若用户发现 Access Key 与 Secret Key 泄露，可以在云存储中重新申请新的 Access

Key 与 Secret Key。

3.3 Bucket

Bucket 是存放 Object 的容器，用户可以根据自身需要创建不同的 Bucket，Bucket 名称在海康云存储系统中必须整个唯一，每个用户最多只能拥有 10 个 Bucket，且 Bucket 与 Bucket 之间不能嵌套。

由于 Bucket 的操作是中心化的，所有用户应尽量避免进行频繁的创建、删除等相关操作。

Bucket 的命名必须必须经过 utf-8 编码，且遵循以下规则：

1) 由小写字母、数字、下划线组合而成
2) 开头必须是数字或小写字母
3) 长度必须大于等 3 字节，小于等于 255 字节
4) 不能与 hikcstor 作为 Bucket 的开头(hikcstor 可能为海康云存储内部标示)

3.4 Object

Object 是承载数据的实体，由键(key)、数据(data)、元数据(metadata)三部分组成，关于数据，海康云存储并不关心其具体内容是什么，而元数据则是一组键值(key-value)组合，分为系统预定义元数据(system-defined metadata)和用户自定义元数据(user-defined metadata)两部分。元数据大小不能超过 2048 字节，若超过海康云存储系统定义长度，则会返回错误。

系统预定义元数据：为了系统内部管理的需要，海康云存储对其存储的每个 Object 都维护一组系统定义元数据，包括 Object 的长度、创建时间、最后修改时间、MD5 等。

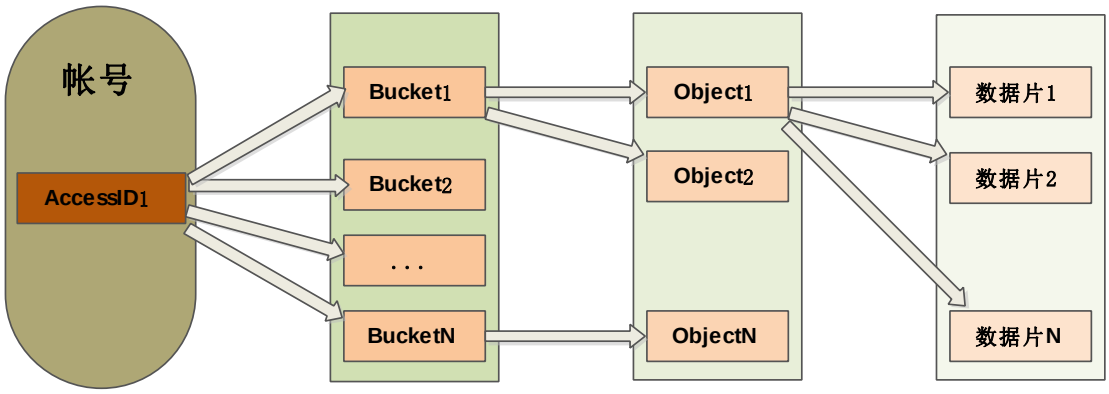
用户自定义元数据：由用户自己定义，海康云存储并不关心用户自定义元数据的具体内容。

Object key 用来标示 Bucket 内部的唯一 Object，用户可以根据 Bucket + Key 的方式对 Bucket 下的 Object 进行读写操作。例如对于 URL：http://object.hikcstor.com/mybucket/public/test.html，域名之后到第一个斜线(/)是 Bucket，Bucket 之后的第一个斜线(/)之后的内容全部为 Object 的 key，在这里为 public/test.html。

Object 的 Key 需经过 UTF-8 编码，且符合以下规则：

1) 长度必须大于等 1 字节，小于等于 260 字节

3.5 Object 存储模型



Object 对象的实际数据根据大小可能分布在不同的存储设备中，以数据片形式进行存储，数据片可能是一个完整的 Object，可能是 Object 的一部分。

4. 安全认证

海康云存储系统会对访问的请求进行安全认证，只有通过认证的请求，才能访问云存储中的资源。

认证过程如下：

1、 客户端构造访问云存储的请求

2、 通过 Secret Key 计算请求的签名(signature)

3、 将附带 Access Key 与签名的信息发送给海康云存储系统

4、 海康云存储系统利用 Access Key 对应的 Secret Key 重新计算请求的签名

5、 若海康云存储系统计算的签名与客户端提供的签名一致，则认证通过，否则请求被拒绝。

海康云存储系统通过基于 HMAC(hash message authentication code)的算法自定义 http 头来进行认证操作。要认证一个请求，客户端首先从请求中选择一些关键的元素组成一个字符串，然后利用自己的 Secret Key 与该字符串进行 hmac 运算，得出签名之后，用户将该签名以请求参数形式发送给海康云存储。

带有请求认证信息的请求样例如下：

```
DELETE /mybucket/myobject HTTP/1.1

Date: Tue, 21 May 2015 12:00:00 GMT

Authorization: hikcstor laskdfsirios567sdfsd:VREEI/ASfskjfagzdgG=

Host: object.hikcstor.com
```

4.1 构造认证请求

海康云存储 REST API 通过标准的 HTTP 头 Authorization 来传递认证信息，下面是 HTTP 认证头部模板:

```
Authorization: hikcstor    Access Key:Signature
```

下面是生成 Authorization 头的伪代码：

```
Authorization = “hikcstor” + “ ” + Access Key + “:” + Signature;

Signature = Base64(HMAC_SHA1(Secret Key, UTF-8(StringToSign)));

StringToSign = HTTP-Verb + “\n” +

                Content-MD5 + “\n” +

                Content-Type + “\n” +

                Date + “\n” +

                CanonicalizedHCSHeaders + “\n” +

                URI;
```

注意：CanonicalizedHCSHeaders 代表 HTTP 头中以 x-hcs-为前缀的组合，URI 表示用户想要访问的资源。若伪代码中的某些内容不存在，则使用空字符串代替。

4.1.1HMAC_SHA1

HMAC-SHA1 的算法详细信息可以参考 RFC 2104 标准，该算法输入 2 个参数：key 和 Message，并计算出一个结果 Digest，对于海康云存储来说，Key 就是 Secret Key，Message 就是经过 UTF-8 编码之后的 StringToSign，计算出的结果 Digest 经过 Base64 编码之后就是 Signature。

4.1.2例子

AccessKey	SecretKey
9c379f079214447fad2959c4621cd6feVb797oH1	5e998dbbafb44ca783099afcdead40fa7A3Vf7Fh

Put Bucket 请求

Request	StringToSign
PUT /ab52b360-5370-4c03-906f-8b80e7e0c130 HTTP/1.1 Date: Wed, 22 May 2013 02:05:58 GMT Authorization: hikcstor 9c379f079214447fad2959c4621cd6feVb797oH1:5IGUVXmvjWCJfkRDH7G+/gyIsf8=	PUT \n \n Wed, 22 May 2013 02:05:58 GMT \n /ab52b360-5370-4c03-906f-8b80e7e0c130

Put Object 请求

Request	StringToSign
PUT /7d84df14-6e90-4101-bd92-0201966eacc5/24b1c9ba-c889-4a76-8edc-bd8fa7e417dc HTTP/1.1 Content-MD5: 670f34c390bd3deb23c99999771064ad	PUT 670f34c390bd3deb23c99999771064ad

Content-Type: application/octet-stream Date: Wed, 22 May 2013 02:37:02 GMT Authorization:hikcstor 9c379f079214447fad2959c4621cd6feVb797oH1:J6yRNUPxjixPsJusHuHk0JN K1Lo=	application/oc tet-stream Wed, 22 May 2013 02:37:02 GMT /7d84df14-6e90-4101-bd92- 0201966eacc5/24b1c9ba-c889- 4a76-8edc-bd8fa7e417dc

5. 通过 REST API 对云存储进行管理

海康云存储服务提供开放的 REST 风格的 HTTP API，用户可以根据该 API 来进行 Object 的上传或者下载操作。

5.1 错误响应

5.1.1错误代码列表

错误码	HTTP 状态码	描述
AccessDenied	403 Forbidden	访问被拒绝
AccessIDProblem	403 Forbidden	账号异常
BadDigest	400 Bad Request	提供的 Content-MD5 值与服务器计算的不匹配
BucketAccessDenied	403 Forbidden	没有对应的 Bucket 权限
BucketAlreadyExists	409 Conflict	Bucket 以存在
BucketNotEmpty	409 Conflict	Bucket 不为空，无法删除
IncompleteBoby	400 Bad Request	数据大小与 Content-Length 不对应
InvalidBucketName	400 Bad Request	Bucket 名称不合法
KeyTooLong	400 Bad Request	Object Key 名字过长
NoSuchBucket	404 Not Found	请求的 Bucket 不存在
NoSuchKey	404 Not Found	请求的 Object Key 不存在
RequestTimeTooSkewed	403 Forbidden	请求的时间戳与服务器时间戳相隔太大(15 分钟)
InvalidPartID	400 Bad Request	分片上传时 PartID 不存在

5.1.2错误相应结构

当有错误发生时，hikcstor 的响应包括：

1、 相应的 HTTP 3xx、 4xx、5xx 状态码

2、 XML 格式的消息体。

错误消息响应体如下：

```
<?xml version=”1.0” encoding=”UTF-8”?>
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestID>85235468E64C39638743ad9049D56EAE83CB91</ RequestID >
</ Error >
```

其中 Code：错误代码。如 AccessDenied

Message：错误消息的描述信息，可以通过此信息来初步定为问题所在。

RequestID：表示请求的唯一 ID。

5.2 REST API 接口

5.2.1通用请求头

下面的表格是访问海康云存储系统 API 的通用请求头(HTTP Headers):

Header Name	Description
Authorization	本次请求生成的签名值
Content-Length	消息体的长度，不包括请求头部
Content-Type	消息体的类型，如:text/plain
Content-MD5	使用 base64 进行了编码的请求体的 MD5 校验和

Date	请求端的当前本地时间，例如:12 Dec 2014 12:00:00 GMT
Host	服务器的域名或 IP 地址,如果服务器是动态端口，则需指明端口号，如 192.168.1.1:5120

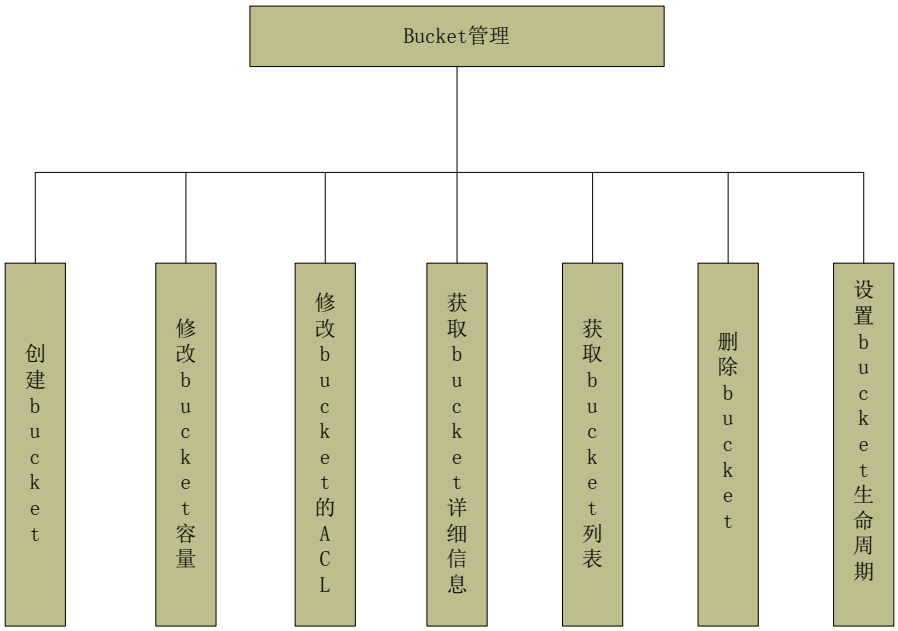
5.2.2通用响应头

下面是海康云存储系统返回的通用头部:

Header Name	Description
Content-Length	消息体的长度，不包括请求头部
Content-Type	消息体的类型，如:text/plain
Connection	表明服务器上的本次连接状态是 open 还是 close
Date	请求端的当前本地时间，例如:12 Dec 2014 12:00:00 GMT
ETag	标记 object 的 hash 值，它通常用于校验一个实体有没有被修改过，在 HikCStor 中，当 object 被创建或修改时，返回的是 object 的 MD5 值。
Server	响应该请求的服务器的域名或 IP 地址

5.2.3Bucket 相关接口

用户可以使用 Bucket 相关接口对云存储中的 Bucket 进行管理，目前海康云存储支持的管理方法如下表：



5.2.3.1 创建 Bucket

5.2.3.1.1 描述

hikcstor 中存储的每个 Object 都包含在 Bucket 中，Bucket 是 Object 对应的容器，在创建 Bucket 时，需要为 Bucket 命名，该命名需在 hikcstor 中唯一。且 Bucket 下不能嵌套创建 Bucket。

在 hikcstor 中，每个用户最多能创建 100 个 Bucket。其他用户在访问该 Bucket 时，需拥有该 Bucket 的权限，在删除 Bucket 时，需确保该 Bucket 下无任何 Object。

Bucket 下能拥有的 Object 数量无限制，Bucket 由于是中心化管理，所以不建议用户对 Bucket 进行频繁的创建、删除等操作。

Bucket 与地理区域相关，每个 Bucket 里面的 Object 只能存在在一个区域中，当用户在创建 Bucket 时，需指定该 Bucket 存储在哪个区域中，之后处于该 Bucket 中的所有 Object 都处于该区域中。目前 hikcstor 该字段只做保留字段。

在创建 Bucket 的同时需指定该 Bucket 的大小，Bucket 的大小单位为 GB。

Hikcstor 中的 Bucket 具有周期性(单位为天)，若用户不设置 Bucket 的存储周期，则 Bucket 下面的所有 Object 除非用户主动删除，否则一直存在于 Bucket 中。若用户设置了 Bucket 的存储周期，则该 Bucket 下

面的 Object 在上传成功那刻起，在 Bucket 中存储的时长达到了 Bucket 的存储周期，则 hikcstor 会自动删除该 Object，以达到 Bucket 下 Object 的过期自动删除的目的。

目前 hikcstor 支持的 Bucket 的权限包括：

1) public-read-write

2) public-read

3) private

5.2.3.1.2 请求语法

```
PUT /YourBucketName HTTP/1.1
Content-Length: YourLength
Date: YourDate
Authorization: YourAuthorization string
```

其中 Request 中还可以根据需要携带其他元素信息：

Header Name	Description	是否必须
Acl	Bucket 的 ACL 权限值(默认 private)	否
LocationConstraint	Bucket 所在的地理区域(默认本地机房)	否
Cycle	Bucket 的存储周期(默认永久存储,单位为天)	否
Size	Bucket 的存储空间大小(单位为 MB)	是

5.2.3.1.3 例子

创建一个名为 test 的 bucket 的请求,且 Bucket 存储区域为 huazhong-1，Bucket 的 ACL 值为 public。且设置其存储周期为 3 天，容量为 102400GB。

请求：

```
PUT /test    HTTP/1.1

Host: object.hikcstor.com

Content-Length: 512

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: hikcstor test: KLKVOIASIOHV/JOISUDKJ===+SSDGG


<?xml version=”1.0” encoding=”UTF-8”?>

<CreateBucketConfiguration>

<Acl>public </Acl>

<LocationConstraint>huazhong-1</LocationConstraint>

<Cycle>3</Cycle>

<Size>102400</Size>

</CreateBucketConfiguration>
```

响应：

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.3.2 修改 Bucket 的容量

5.2.3.2.1 描述

海康云存储允许用户修改一个已经存在的 Bucket 的容量，Bucket 的容量只能被原始用户操作。

5.2.4.2.1 请求语法

```
PUT /BucketName?size=new_size    HTTP/1.1

Host:object.hikcstor.com
```

```
Date: date

Content-Length:0

Authorization: authorization string
```

5.2.3.2.2 例子

将 Bucket 的名字为 test 的容量修改为 65536GB。

请求：

```
PUT /test? size=65536 HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.3.3 修改 Bucket 的 ACL

5.2.3.3.1 描述

海康云存储允许用户修改一个已经存在的 Bucket 的 ACL，Bucket 的 ACL 只能被原始用户操作。

当 Bucket 重新设置 ACL 之后，原来 ACL 会被覆盖。

目前 hikcstor 支持的 Bucket 的 ACL 包括：

1) public-read-write

2) public-read

3) private

其中 public-read-write 除了自己能访问自己 bucket 下面的 object 外，其他人也能访问并且能对 bucket 下面的 object 进行删除，所以若无特殊场景，推荐将自己 bucket 的 ACL 值设置为 private。若用户需要做 bucket 共享时，可以将 bucket 的 ACL 设置为 public-read，这样其他用户就能读取该 bucket 下面的 object 数据，但是不能对 object 进行删除操作。若用户不想其他人知晓自己 bucket 的任何信息，可以将 bucket 的 ACL 设置为 private。

5.2.3.3.2 请求语法

```
PUT /BucketName?acl    HTTP/1.1

Host:object.hikcstor.com

Date: date

Content-Length: length of request body

Authorization: authorization string


<?xml version=”1.0” encoding=”UTF-8”?>

< AclConfiguration >

<Acl>new acl</Acl>

</ AclConfiguration >
```

5.2.3.3.3 例子

请求修改 test 的 ACL 值为 public。

请求：

```
PUT /test? acl    HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT
```

```
Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG


<?xml version=”1.0” encoding=”UTF-8”?>
< AclConfiguration >
<Acl>public</Acl>
</ AclConfiguration >
```

回复

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
Connection: close
```

5.2.3.4 获取单个 Bucket 的详细信息

5.2.3.4.1 描述

获取一个已经存在的 Bucket 的详细信息。

5.2.3.4.2 请求语法

```
GET /BucketName?property  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
```

5.2.3.4.3 例子

请求获取 test 的详细信息。

请求：

```
GET /test?property    HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Last-Modified: Sun, 1 Jan 2014 12:00:00 GMT

Content-Length: 124

Content-Type: text/xml

Connection: close


<?xml version=”1.0” encoding=”UTF-8”?>

<Bucket>

<Owner>zhangsan </Owner>

<Name>bucket_4</Name>

<CreateDate>Mon, 9 Feb 2015 13:48:12 GMT</CreateDate>

<Location>LOCAL-1</Location>

<Acl>public</Acl>

<LifeCycle>none</LifeCycle>

</Bucket>
```

注意：若用户没有设置 Bucket 的存储周期，则 hikcstor 默认其为 none。

5.2.3.5 获取用户下 Bucket 列表

5.2.3.5.1 描述

获取一个用户下所拥有的所有 Bucket 列表信息。

5.2.3.5.2 请求语法

```
GET /    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string
```

5.2.3.5.3 例子

请求获取所有的 bucket

```
GET /    HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 124

Content-Type: application/xml

Connection: closed


<?xml version=”1.0” encoding=”UTF-8”?>
```

```
<ListAllMyBucketsResult>
<TotalNum>2</TotalNum>
<Owner>video</Owner>
<Buckets>
<Bucket>
<Name>test1</Name>
<CreateDate>Fri, 16 Oct 2015 05:51:36 GMT</CreateDate>
<Location>LOCAL-1</Location>
<Acl>public-read-write</Acl>
<VersionPolicy>0</VersionPolicy>
<LifeCycle>-1</LifeCycle>
<Size>100</Size>
<FreeSize>95.416016</FreeSize>
</Bucket>
<Bucket>
<Name>test2</Name>
<CreateDate>Mon, 19 Oct 2015 12:13:46 GMT</CreateDate>
<Location>LOCAL-1</Location>
<Acl>private</Acl>
<VersionPolicy>0</VersionPolicy>
<LifeCycle>-1</LifeCycle>
<Size>1</Size>
<FreeSize>0.000000</FreeSize>
</Bucket>
</Buckets>
</ListAllMyBucketsResult>
```

5.2.3.6 删除 Bucket

5.2.3.6.1 描述

用户可以删除一个 Bucket，但是在删除 Bucket 前，需用户确保删除的 Bucket 下的 Object 列表为空，否则海康云存储系统将会返回对应的失败错误码。

5.2.3.6.2 请求语法

```
DELETE / BucketName    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string
```

5.2.3.6.3 例子

请求删除名为 test 的 Bucket

```
DELETE /test HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.3.7 设置 Bucket 的生命周期值

5.2.3.7.1 描述

用户可以用此接口来设置 bucket 下面的 object 的存储周期，单位为天。当设置了 bucket 的生命周期成功后，该 bucket 下面的 object 存储到云存储成功之后，若 object 在 hikcstor 中存在的时长超过了该 object 所在的 bucket 的生命周期值范围后，hikcstor 会将其自动删除。故除非有明确需求，否则不应开启此功能。

5.2.3.7.2 请求语法

```
PUT    /BucketName?lifecycle=new_lifecycle    HTTP/1.1

Host: Host Server

Date: date

Authorization: authorization
```

5.2.3.7.3 例子

请求设置 test 的生命周期管理值为 365 天：

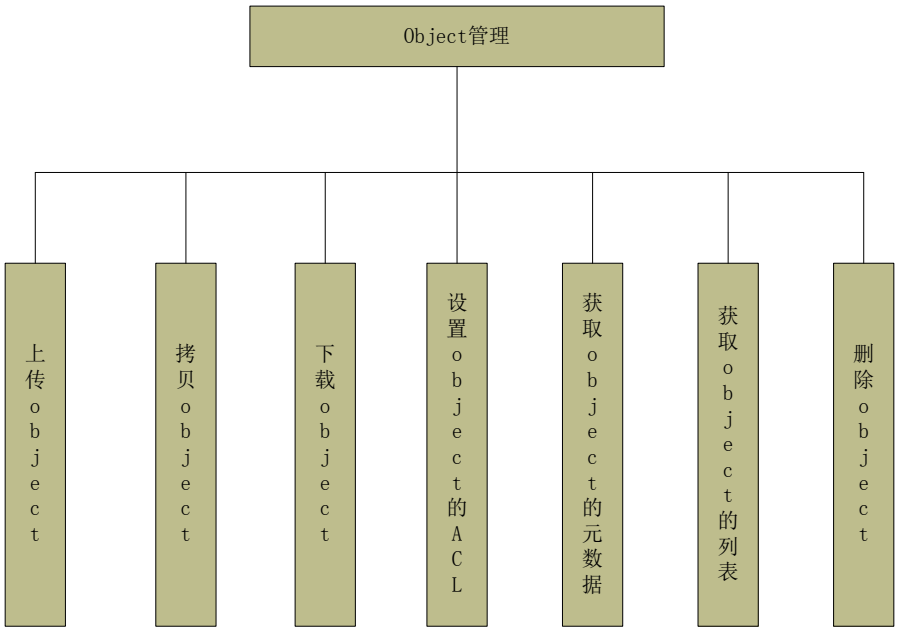
```
PUT    /test?lifecycle=365    HTTP/1.1

Host: Host Server

Date: date

Authorization: authorization
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Last-Modified: Sun, 1 Jan 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.4 Object 相关接口

用户可以通过简单的 REST 接口对自己已经上传的 object 进行管理，目前海康云存储支持的 object 操作有：



5.2.4.1 上传 Object

5.2.4.1.1 描述

用户在上传 object 前，需先获取云存储中的最优节点(参考 5.2.6)，获取最优节点成功之后，用户指需向该节点上传数据即可。

该 API 允许用户上传一个 Object 对象，一个 Object 对象唯一只能属于一个 Bucket。Object 是实际存储数据的载体，由 Key,data,metadata 组成。

若用户需要上传的 Object 很大(如超过 1G 大小)，推荐使用 Multipart Upload 上传接口进行操作。详细请参考 Multipart Upload 上传接口说明。

单次上传的 Object 的大小不能超过 5G，否则云存储会返回对应错误码。

用户并发上传同一个 Object 本身不影响 hikcstor 正常运行，但是 hikcstor 只保存时间戳最新的 Object，故用户需避免对同一个 Object 的并发写操作。

为了保证上传到 hikcstor 中的数据的准确性，用户可以在上传前事先计算 object 的 MD5 值，并在上传

时将该 MD5 值发送至 hikcstor，hikcstor 会重新计算收到的 object 的 MD5，并与用户发送的 MD5 进行比较，若不一致，云存储将返回相应错误码。

5.2.4.1.2 请求语法

```
PUT / BucketName/Object_Key  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: length of data

[object data]
```

该请求支持如下扩充头部：

Header Name	Description	是否必须
Content-Length	需要上传的 Object 的大小	是
Content-MD5	期待上传的 Object 的 MD5 值	否
Content-Type	Object 的数据类型，默认 binary/octet-stream	否

5.2.4.1.3 例子

请求上传 test 的 Object 至 testbukcet 中

```
PUT /testbukcet/test HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG
Content-Length: 10240
x-hcs-meta-who：xiaoming
x-hcs-meta-old: 23
```

```
Test For Object Data
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.4.2 拷贝 Object

5.2.4.2.1 描述

该操作用来将海康云存储中的某个 Object 拷贝至新 Bucket 中，当用户在拷贝 Object 时，可以为新 Object 重新制定 metadata,否则将使用源 Object 的 metadata。

对于大对象的拷贝，海康云存储可能花费很长时间，当海康云存储收到拷贝请求后，在拷贝还未开始前，如出现错误，海康云存储返回对应错误码，当海康云存储在后台开启了异步拷贝任务之后，海康云存储会返回用户成功开启拷贝任务的消息。

用户在进行对象拷贝时，需拥有源 Bucket 的读权限、源 Object 的读权限、目的 Bucket 的写权限，否则云存储会返回权限不足错误。

5.2.4.2.2 请求语法

```
PUT / DestBucketName/Dest_Object_Key    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

x-hcs-copy-source：/source_bucket/source_object_key
```

在进行 Object 的拷贝时，请求头部支持如下扩展头：

Header Name	Description	是否必须
x-hcs-copy-source	源 Object 所在的 Bucket 名称与 Object key	是
x-hcs-meta-	新 Object 对应的用户自定义信息	否

5.2.4.2.3 例子

请求拷贝对象/ testbukcet /testobject 到/newbukcet/new_obj 中，则请求如下：

```
PUT /newbukcet/new_obj?copy HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG

x-hcs-copy-source：/ testbukcet /testobject

x-hcs-meta-location: wuhan
```

回复

```
HTTP/1.1 202 Accepted

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.4.3 下载 Object

5.2.4.3.1 描述

用户可以下载需要的 Object，在下载前，用户需拥有该 object 所在的 Bucket 与该 Object 对应的权限。

5.2.4.3.2 请求语法

```
GET / BucketName/Object_Key    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string
```

在该请求中，还支持如下 HTTP 头部：

Header Name	Description	是否必须
Range	获取指定范围的数据内容	否

5.2.4.3.3 例子

请求获取/bucketkey/objectkey 对象的数据：

```
GET /bucketkey/objectkey    HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG

Range: bytes=0-1024
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 1025

Content-Type: text/plain

Connection: close

Content-Range: bytes 0-1024/2048


[1025 bytes of object data]
```

5.2.4.4 设置 Object 的 ACL

5.2.4.4.1 描述

hikcstor 允许用户修改一个已经存在的 object 的 ACL，object 的 ACL 只能被原始用户(即创建该 object 的用户)或 bucket 的拥有者操作(即创建该 bucket 的用户)操作。

当 object 重新设置 ACL 之后，原来 ACL 会被覆盖。

目前 hikcstor 支持的 object 的 ACL 包括：

1) public-read-write

2) public-read

3) private

其中 public-read-write 除了自己能访问该 object 外，其他人也能访问并且能对该 object 进行删除，所以若无特殊场景，推荐将自己 object 的 ACL 值设置为 private。若用户需要做 object 共享时，可以将 object 的 ACL 设置为 public-read，这样其他用户就能读取该 object 数据，但是不能对 object 进行删除操作。若用户不想其他人知晓自己 object 的任何信息，可以将 object 的 ACL 设置为 private。

5.2.4.4.2 请求语法

```
PUT /BucketName/ObjectKey?acl    HTTP/1.1
Host:object.hikcstor.com
Date: date
Authorization: authorization string


<?xml version="1.0" encoding="UTF-8"?>
< AclConfiguration >
<Acl>new acl</Acl>
</ AclConfiguration >
```

5.2.4.4.3 例子

修改 test_bucket 下的 test_obj 的 ACL 值为 public。

请求：

```
PUT /test/test? acl HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG


<?xml version="1.0" encoding="UTF-8"?>
< AclConfiguration >
<Acl>public-read-write</Acl>
</ AclConfiguration >
```

回复

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
Connection: close
```

5.2.4.5 获取 Object 的 metadata 信息

5.2.4.5.1 描述

用户在上传完 object 之后，可以重新获取 object 的用户自定义 metadata 信息。该请求只返回 object 的 metadata 信息，不返回 object 数据，若用户在上传 object 时指定了 x-hcs-meta-信息，此时会在 HTTP 响应头部中一起返回。

5.2.4.5.2 请求语法

```
GET /BucketName/ObjectKey?metadata  HTTP/1.1

Host:object.hikcstor.com

Date: date

Authorization: authorization string



<Version>

<VersionID>24</VersionID>

</Version>
```

其中消息体可以根据需要添加，不指定版本号时候，就不需要传递消息体，对象存储系统会返回版本号最新对象的 metadata 信息。

5.2.4.5.3 例子

请求获取/test/test 的元数据。

请求：

```
GET  /test/test?metadata  HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: KLKVOIASIOHV/JOISUDKJ===+SSDGG



<Version>

<VersionID>24</VersionID>

</Version>
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Connection: close

ETag: 957112sdfased213d2fas4er65a4sdf
```

```
Last-Modified: Mon, 27 Feb 2015 08:03:45 GMT

Content-Type: text/plain

Content-Length: 10240

x-hcs-meta-who: xiaoming
```

5.2.4.6 获取 Object 列表

5.2.4.6.1 描述

用户可以根据需要获取某个 Bucket 下的所有 Object 列表，用户可以添加入前缀等参数增加获取结果速度。

由于一个 Bucket 下的 Object 数量较多，不建议频繁的进行该操作。而是利用获取单个 Object 详细信息接口。

5.2.4.6.2 请求语法

```
GET /BucketName    HTTP/1.1

Host:object.hikcstor.com

Date: date

Authorization: authorization string
```

该请求支持一些特定的限定语，如下所示：

名称	描述	是否必须
Marker	返回以字典序排列的 Object 信息的起始标志，若结果中不包含 marker 信息，可以利用 max-keys 实现分页。	否
MaxKeys	返回 object 信息的数量，最大支持 1000，若用户不指	否

	定该值，云存储会默认该值为 1000,。若有更多的信息未返回，则响应头重会包含 <IsTruncated>true</IsTruncated>元素，	
Prefix	限制返回 object 的 key 的前缀	否
Delimiter	分组符，用于分组返回的Object的Key。当prefix未指定时，从Object的Key中提取第一个字符到第一个delimiter之间的字符串放在CommonPrefixes中返回。若指定了prefix，则提取prefix到第一个delimiter之间的字符串放在CommonPrefixes中返回，具体使用请参看后面的例子。	否

hikcstor 会在响应体重添加一些返回元素，供用户使用，如：

名称	描述
Name	Bucket 的名称
Delimiter	用户指定的分组符
Prefix	指定的Object Key的前缀
Max-Keys	指定返回 Object 的数量
Marker	指定的 Object 的 Key 的起始标志
NextMarker	列出下一个集合的Marker，只有当用户设置了Delimiter，且HasNext为true时才会出现，其值可能是当前集合中最后一个key，或者是CommonPrefixes中的最后一个prefix。 若用户没有设置Delimiter，且IsTruncated为true时，可以直接将返回集合中的最后一个key作为下一个集合的Marker
IsTruncated	标示返回的结果是否完整(true/false)，若符合条件的Object信息数量超过了指定的MaxKeys，则该值为true，且多余的结果不会被返回。
TotalNum	本次总共返回了多少条Object记录

Contents	单个Object的信息容器
Key	Object的Key值
LastModified	Object的最后修改时间
Etag	Object的ETag是一个MD5 hash值，其只反映Object的内容的变化，而不是元数据。
Size	Object的大小
CommonPrefixes	用户指定了Delimiter时，该元素才会出现，用户存放分组后的Object的前缀

5.2.4.6.3 例子

请求获取 Bucket 为/testbucket 下的所有以 home/为前缀的 Object 列表

```
GET /mybucket ? Prefix =home/    HTTP/1.1

Date: Wed, 22 Feb 2012 07:15:56 GMT

Host: object.hikcstor.com

Authorization:hikcstor::im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 512

Connection: close


<?xml version="1.0" encoding="UTF-8"?>

< ListObjectResult >

    < Name > mybucket</ Name >

    <Prefix>home/ </Prefix>

    <Marker>null</Marker>
```

```
    <Delimiter>null</ Delimiter >
    <MaxKeys>1000</ MaxKeys >
    < IsTruncated >false</ IsTruncated >
    <ContentsNum>1< ContentsNum >
    <ContentsList>
        <Contents>
                <Key>Object1</Name>
                <LastModified>2014-12-12 12:00:00</ LastModified >
                <Etag>123</ Etag >
                <Size>65536</Size>
                <DisplayName></ DisplayName >
        </Contents>
    </ContentsList>
    < CommonPrefixesNum>1< /CommonPrefixesNum>
    <CommonPrefixesList>
        <CommonPrefixes>
            <Prefix>books/</Prefix>
        </CommonPrefixes>
    <CommonPrefixesList>
</ ListObjectResult >
```

5.2.4.7 删除 Object

5.2.4.7.1 描述

用户可以删除某个 Object，在删除 Object 前，用户需确保拥有该 Bucket 与该 Object 的权限。

5.2.4.7.2 请求语法

```
DELETE / BucketName/Object_Key   HTTP/1.1
Host: object.hikcstor.com
```

```
Date: date

Authorization: authorization string
```

5.2.4.7.3 例子

请求删除/bucket/object 的数据：

```
DELETE / bucket /object    HTTP/1.1

Host: object.hikcstor.com

Date: Wed, 12 Dec 2014 12:00:00 GMT

Authorization: hikcstor im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
```

回复：

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.4.8 Append Object

5.2.4.8.1 描述

Append object 功能主要应对用户利用追加方式将数据上传至云存储中，用户在进行该操作时，云存储会返回下次用户需要继续 append 的位置。

用户进行追加的 object 对应的类型会被自动至为 APPENDABLE，在用户进行 object 列表查询时，云存储会将该类型在协议中进行表明，否则 object 的类型则为 STANDARD, 用户不能对一个 object 类型是 STANDARD 的 object 进行 append 操作，反之，用户若对一个 APPENDABLE 类型的 object 进行普通的上传操作，则之前 APPENDABLE 类型对应的数据则会被后者覆盖。

类型为 APPENDABLE 的 object 不能被执行拷贝操作，因为云存储无法确认用户是否会继续对该 object

进行 append 操作。

用户在进行 append 操作时，若云存储中记录的下次 append 位置与用户本次传递的不一致，则云存储会返回一个错误，并将云存储记录的 position 位置也一并返回。这种情况下云存储返回的协议头如下：

名称	描述
x-hcs-next-append-positon	表明下次应该提供的 position 位置，实际上该值就表明用户在这之前已经 append 了多少数据。当 append 返回成功，或者云存储中记录的 append position 与用户传递不一致时，该头部都会被云存储返回。

用户在调用该接口时，应预先向云存储申请一个最优上传节点(参考协议 5.2.6.1)，云存储会返回一个 IP 地址与端口号，用户将接下来的上传协议发送至云存储返回的 IP 与端口即可。

Append 方式实现了数据追加到云存储中已存在 key 后面的方法，用户需要自己保证 append 的数据对于 key 是有效的。

5.2.4.8.2 请求语法

```
POST / BucketName/Object_Key?append&position=your_position  HTTP/1.1
Host: object.hikcstor.com
Content-Length：ContentLength
Date: date
Authorization:  authorization string

[object data]
```

5.2.4.8.3 例子

请求追加/bucket/object 的数据：

```
POST / bucket/object?append&position=0  HTTP/1.1
Host: object.hikcstor.com
Content-Length：1024
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization:  hikcstor im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
```

```
[object data]
```

回复：

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close

x-hcs-next-append-positon: 1024
```

5.2.4.9 表单上传 Object

5.2.4.9.1 描述

该功能可以让开发者使用 HTML 表单上传文件到指定的 bucket，使得基于浏览器上传文件到 bucket 成为可能。该功能适用于那些较小对象的提交，且无法支持断点上传，若用户需要支持断点上传，可以使用分片上传(5.2.5 节)功能。

5.2.4.9.2 请求语法

```
POST /bucket HTTP/1.1

Host: object.hikcstor.com

Content-Length：ContentLength

Content-Type: multipart/form-data; boundary=<Boundary>


--<Boundary>

Content-Disposition: form-data; name="key"


key

--<Boundary>

Content-Disposition: form-data; name=" x-hcs-meta-** "
```

value

--<Boundary>

Content-Disposition: form-data; name="AccessKey "

access_key

--<Boundary>

Content-Disposition: form-data; name="Signature"

Signature_value

--<Boundary>

Content-Disposition: form-data; name="Date"

date_value

--<Boundary>

Content-Disposition: form-data; name="Act "

act_value

--<Boundary>

Content-Disposition: form-data; name="CallBack "

callback_value

--<Boundary>

Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"

Content-Type: Mime Type

file_content

--<Boundary>--

其中参数说明如下：

名称	描述	是否必须

key	指定上传的对象的 key 值，该 key 值必须经过 UTF-8 编码。	是
x-hcs-meta-	用户自定义元数据信息。如 x-hcs-meta-who 等。	否
AccessKey	请求用户的 AccessKey，该值在云存储创建用户时可以获取到。	是
Signature	请求计算签名信息，该值生成方法请参考4.1节。	是
Date	请求发送的时间戳	是
CallBack	回复给客户端的消息体中的回调地址	否
Act	用户要求的云存储动作。 0：表示无动作 1：表示需要帧分析 默认为0。	否

5.2.4.9.3 摘要计算字符串

表单由于参数大部分在体中定义，数据体的 MD5 在网页中无法计算等原因，摘要计算部分需要特殊处理，摘要计算流程请参考 4.1 章节内容，计算字符串定义如下：

StringToSign = HTTP-Verb + “\n” +

Content-MD5 + “\n” +

Content-Type + “\n” +

Date + “\n” +

CanonicalizedHCSHeaders + “\n” +

URI + “\n”

Key + “\n”

CallBack + “\n”

Act;

其中 Content-Type 采用表单 http 头中的 Content-Type，并且去掉了值中的 boundary 部分数据，只取分号”;”之前的值（不包括分号），Content-MD5 传入空字符串，CanonicalizedHCSHeaders 加入了体中的自定

义元数据，Key，Date，CallBack，Act 采用体中的参数，没有传递的参数传入空字符串。

5.2.4.9.4 例子

例如用户想要在名字为 bucket1 的 bucket 中通过表单方式上传一个 key 为 test 的对象数据。那么协议交互如下：

```
POST /bucket1 HTTP/1.1
Host: object.hikcstor.com
Content-Length：95632
Content-Type: multipart/form-data; boundary=88211491561685

--88211491561685
Content-Disposition: form-data; name=”key”

test
--88211491561685
Content-Disposition: form-data; name=”x-hcs-meta-who “

xiaoming
--88211491561685
Content-Disposition: form-data; name=”AccessKey “

im09ttopcc2a5qybj
--88211491561685
Content-Disposition: form-data; name=”Signature”

dFrR+YSSAWodMz3p85i/6Xw3GKI=
--88211491561685
Content-Disposition: form-data; name=”Date”

Wed, 2 Mar 2016 8:50:33 GMT
--88211491561685
Content-Disposition: form-data; name=”Act”

0
--88211491561685
Content-Disposition: form-data; name=”CallBack”

http://10.20.34.106:8080/modules/classmanage/uploadCallBack.action
--88211491561685
Content-Disposition: form-data; name=”file “; filename=”cat.jpg “
Content-Type: Mime Type

[object data]
```

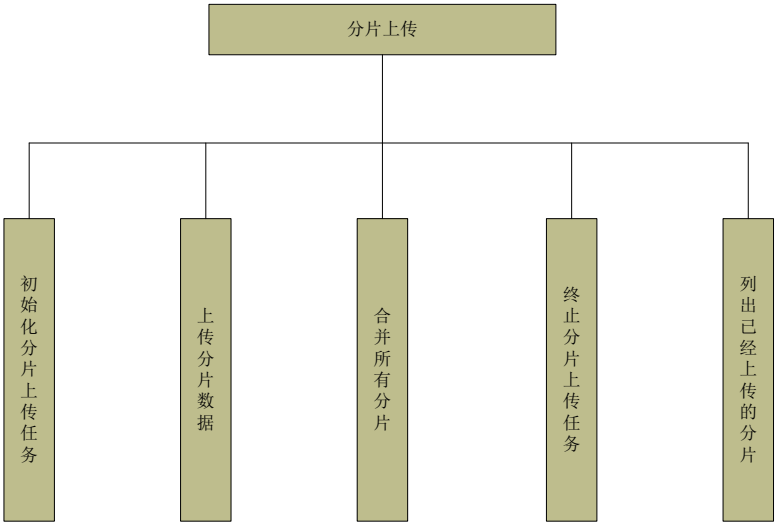
```
--88211491561685--
```

回复：

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 336

Connection: close

<!DOCTYPE html>

<html lang="zh_CN">

<head>

<script type="text/javascript">

try{

var callback = "http://10.20.34.106:8080/modules/classmanage/uploadCallBack.action";

var params = "?res=" + encodeURIComponent('{"ret":0,"msg":"操作成功"}'); //{"ret":1,"msg":"失败原因"}

location.href = callback == null || callback == "" ? location.href + "#" : callback + params;

}catch(e){}

</script>

</head>

</html>
```

5.2.5Multipart Upload 相关接口

若用户需要上传的 object 大小超过 100MB，海康云存储推荐使用分片上传接口。而分片上传接口集提供如下功能：



5.2.5.1 初始化 Multipart Upload 任务

5.2.5.1.1 描述

用户在开始上传一个很大对象前，可以先将对象进行分片，海康云存储建议每片数据大小为 2M，最后一片数据以实际大小为准。

当初始化一个任务成功之后，海康云存储系统会返回一个对应的 UploadID，该 ID 在云存储中唯一。用户在进行后续的操作时，必须携带该 ID 进行操作。如上传 part 数据、终止一个 Multipart Upload 任务、列出已经上传成功的所有 part 信息、合并所有 part。

5.2.5.1.2 请求语法

```
POST /BucketName/ObjectKey ? uploads    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string
```

用户可以根据需要填充具体参数值，海康云存储目前支持的参数值如下：

Header Name	Description	是否必须

x-hcs-meta-	Object 对应的用户自定义信息	否

5.2.5.1.3 例子

请求分片上传一个名为 bigdata 的大对象：

```
POST /mybucket/bigdata?uploads HTTP/1.1

Date: Wed, 22 Feb 2014 07:15:56 GMT

Host:object.hikcstor.com

Authorization: hikcstor: im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=

Content-Length: 0

x-hcs-meta-who：xiaoming
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 512

Connection: close


<?xml version=”1.0” encoding=”UTF-8”?>

<InitiateMultipartUploadResult>

<Bucket>mybucket</Bucket>

<Key>bigdata</Key>

<UploadId>CCBKWW315BNIS2G7O6BOK9X2Y</UploadId>

</InitiateMultipartUploadResult>
```

5.2.5.2 上传 part 数据

5.2.6.1.1 描述

在该 API 中，需要用户填入 part_number，表明该片数据对应的序号值，序号值必须单调递增，且从 1 开始。若用户使用相同的 part_number 上传不同的数据，则海康云存储会进行覆盖操作。故 part_number 的使用合法性需用户自己进行保证。

海康云存储系统推荐用户使用的分片大小为 2M 大小，最后一片数据为实际大小。

5.2.5.2.1 请求语法

```
POST /BucketName/ObjectKey ? upload_id=yourid&part_number=your_partnum HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length:512
Content-MD5:2315a4sedfasdfsdg

[Object data]
```

5.2.5.2.2 例子

请求分片上传一个名为 bigdata 的大对象的其中一片数据，片序号为 123：

```
POST /mybucket/bigdata?upload_id=54asd54f6as5d76df&part_number=123  HTTP/1.1
Date: Wed, 22 Feb 2014 07:15:56 GMT
Host: object.hikcstor.com
Authorization: hiksctor  im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
Content-Length: 2048

[2048 bytes of data]
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Connection: close
```

5.2.5.3 终止一个 Multipart Upload 任务

5.2.5.3.1 描述

用户可以随时终止一个已经初始化的 Multipart Upload 上传任务，若用户终止任务之后，海康云存储系统会删除已经上传成功的所有分片信息。

5.2.5.3.2 请求语法

```
DELETE /BucketName/ObjectKey ? upload_id=yourid    HTTP/1.1

Host:object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:0
```

5.2.5.3.3 例子

终止一个 uploadid 为 54asd54f6as5d76df 的操作：

```
DELETE /mybucket/bigdata?uploadId=54asd54f6as5d76df    HTTP/1.1

Date: Wed, 22 Feb 2014 07:15:56 GMT

Host: object.hikcstor.com

Authorization: hikcstor    im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=

Content-Length: 0
```

回复

```
HTTP/1.1 204  No Content

Date: Wed, 12 Dec 2014 12:00:00 GMT

Connection: close
```

5.2.5.4 合并所有 part

5.2.5.4.1 描述

将所有已经上传成功的 part 信合并为一个大对象。在使用该接口时，用户需提供所有已经上传完成的分片的具体信息，如分片 1 对应的大小等。Hikcstor 在接收到对比分片列表后会与自身所记录的分片信息进行对比，若发现分片信息不一致，则会返回对应错误。

5.2.5.4.2 请求语法

```
POST /BucketName/ObjectKey ? upload_id=yourid HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:512


<?xml version=”1.0” encoding=”UTF-8”?>

<CompleteMultipartUpload>

<TotalPartNum></TotalPartNum>

<Part>

    <PartNumber> </PartNumber>

    <PartSize></PartSize>

    <ETag></ETag>
```

```
</Part>

…

</CompleteMultipartUpload>
```

5.2.5.4.3 例子

合并一个 uploadid 为 54asd54f6as5d76df 的操作：

```
POST /mybucket/bigdata?uploadId=54asd54f6as5d76df    HTTP/1.1

Date: Wed, 22 Feb 2014 07:15:56 GMT

Host: object.hikcstor.com

Authorization: hikcstor im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=

Content-Length: 2048


<?xml version=”1.0” encoding=”UTF-8”?>

<CompleteMultipartUpload>

<TotalPartNum>100</TotalPartNum>

<Part>

<PartNumber>1</PartNumber>

<PartSize>65536</PartSize>

<ETag>asdfasdgsdfgsdfg</ETag>

</Part>

<Part>

<PartNumber>2</PartNumber>

<PartSize>65536</PartSize>

<ETag>asdfasdfasdsgsdfgsdfg</ETag>

</Part>

…

</CompleteMultipartUpload>
```

回复

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 0

Connection: close
```

5.2.5.5 列出已经上传成功的所有 part 信息

5.2.5.5.1 描述

用户通过分片任务 ID 可以列出该 ID 对应的已经成功存储至海康云存储中的分片信息。

5.2.5.5.2 请求语法

```
GET /BucketName/ObjectKey ? upload_id=yourid HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:0
```

在该 API 的响应中，海康云存储系统会返回如下元素供用户使用：

Header Name	Description
Bucket	Bucket 的名称
Key	Object 对应的 Key
uploadId	任务对应的 ID 号
PartsNum	本次返回的 part 的数量
Part	包含了一个 Part 的信息的容器
PartNumber	Part 的序号，唯一标识一个 Part

LastModified	Part 的上传时间
ETag	Part 对应的 ETag，一般为 MD5 值
Size	Part 的大小

5.2.5.5.3 例子

请求获取 UploadID 为 weoiaosdjflkasdg 的所有分片信息：

```
GET /BucketName/ObjectKey ? uploadId=weoiaosdjflkasdg HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:0
```

回复：

```
HTTP/1.1 200 OK

Date: Wed, 12 Dec 2014 12:00:00 GMT

Content-Length: 512

Connection: close


<?xml version=”1.0” encoding=”UTF-8”?>

<ListPartsResult>

<Bucket> BucketName </Bucket>

<Key> ObjectKey </Key>

<UploadId>weoiaosdjflkasdg </UploadId>

<PartsNum>1000</PartsNum>

<Part>

    <PartNumber>1</PartNumber>
```

```
    <LastModified>2012-10-12T19:00:35.000Z</LastModified>
    <ETag>”8e68dbaafbf3f0ddc40ce3369ed12c1f”</ETag>
    <Size>10485760</Size>
    </Part>
<Part>
    …
</Part>
</ListPartsResult>
```

5.2.6其他接口(视频文件与图片)

5.2.6.1 获取最优上传节点

5.2.6.1.1 描述

该接口允许用户先获取一个 hikcstor 的最优上传节点，以便用户在随后时间进行上传，上传协议与 5.2.4.1 一致。该协议使得云存储的对接多样化。

5.2.6.1.2 请求语法

```
GET /BucketName/ObjectKey ? bestnode    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length:0
```

5.2.6.1.3 例子

请求获取一个最优上传节点：

```
GET /BucketName/ObjectKey ? bestnode    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length:0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 512
Connection: close

<?xml version=”1.0” encoding=”UTF-8”?>
<BestNodeInfo>
<Location>192.168.10.100:8080</Location>
</BestNodeInfo>
```

5.2.6.2 图片按高宽参数返回数据

5.2.6.2.1 描述

在一些特殊场景下，用户需要在云存储中提取图片数据，用户将数据显示在 web 浏览器中，若原始图片太大，则可能对用户观看造成一定影响。而云存储目前支持通过指定图片的宽与高参数返回对应数据。如用户在云存储中存储了一张 1024*1024 的图片，用户在获取该图片时只想获取 100*100 的图片，这时候该接口就能很好的满足用户的需求。

用户除了指定返回的图片数据的宽与高之外，还可以同时指定返回的图片数据的格式，用户在云存储中存储了一张宽与高位 1024*1024，格式为 jpg 的图片，用户在获取该图片时想将图片缩小成 100*100、且格式的图片格式为 png。这种复杂情况下，该接口也能满足用户的需求。

5.2.6.2.2 请求语法

```
GET /BucketName/ObjectKey ? image&w=width_value&h= high_value& format=pic_format  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length:0
```

上述格式中需要注意的是,若用户指定了 w 参数，则必须指定对应的 h 参数，否则云存储会返回对应错误码。Format 参数与 w、h 参数可以共存，也可以单独存在。当然，用户也可以一个参数都不输入，那样云存储会将原始图片返回。这三个参数的说明如下：

Name	Description	是否必须
w	表明需要返回的图片的宽度	否(与 h 参数共存)
h	表明需要返回的图片的高度	否(与 w 参数共存)
format	表明需要返回的图片的格式,目前该参数支持如下格式转换：jpg, png, bmp，其中参数赋值 0 表示按照源有格式输出，赋值 1 表示将图片转换为 jpg 格式输出,2 表示将图片转换为 png 输出,3 表示将图片转换为 bmp 格式输出。若用户输入其他值，则不会进行格式转换。	否

5.2.6.2.3 例子

用户请求提取 object 的 key 为 cat.jpg 的图片，并按 200*200 的规格进行返回，且返回的图片的格式为 png：

```
GET /test/cat.jpg ? image&w=100&h=100& format=png    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length:0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 512
Connection: close

[对应的图片数据]
```

5.2.6.3 手动录像

5.2.6.3.1 描述

用户可以给云存储下发开始录像指令，同时用户需填充协议中的取流 URL 地址、用户名、密码等信息，云存储执行具体的取流任务与用户下发录像指令是 2 个步骤，若用户填充的 URL 地址有误，则会导致云存

储取流失败，故用户在下发录像指令时需确保取流 URL 地址的正确性。

用户需确保协议中的 object key 的唯一性，否则云存储可能会出现录像数据交叉或者覆盖的问题。

用户下发了开启录像任务后，若一直不停止，该任务默认的持续时长为三天，即云存储收到手动录像任务后，即从前端开始取流，三天后，若用户不手动停止录像，云存储会自动终止该任务的执行，此时需要用户再次下发录像任务，云存储才能继续录像。

5.2.6.3.2 请求语法

```
POST /BucketName/Object_Key?realtime    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: length of data

<?xml version=”1.0” encoding=”UTF-8”?>
<RealTimePlan>
    <StreamDevInfo>
        <UserName></UserName>// 用户名
        <PassWord></PassWord>// 密码
        <StreamURL></StreamURL>// 取流 URL 地址信息
    </StreamDevInfo>
</RealTimePlan>
```

其中参数说明如下：

Name	Description	是否必须
UserName	用户名	是
PassWord	密码	是
StreamURL	取流 URL 地址(用户确保 URL 正确性)	是

5.2.6.3.3 例子

若用户需要现在开始录像，且将取到的流存入名称为 bucket_1 的 bucket 中，数据追加至 key 为 plan_key 的对象中，那么协议例子如下：

```
POST /bucket_1/plan_key?realtime    HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: hikcstor::im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
Content-Length: 172

<?xml version=”1.0” encoding=”UTF-8”?>
<RealTimePlan>
```

```
    <StreamDevInfo>
        <UserName>root</UserName>// 用户名
        <PassWord>123</PassWord>// 密码
        <StreamURL>rtsp://10.64.49.198:4554/hikvision://10.64.51.111:8000:0:0</StreamURL>
    </StreamDevInfo>
</RealTimePlan>
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
Connection: close
```

5.2.6.4 停止录像

5.2.6.4.1 描述

用户给云存储下发开启录像之后，若想停止录像，则需下发停止命令，云存储收到该命令后，会立即终止手动录像任务。该命令只对手动录像有效，手动补录暂不支持停止。

5.2.6.4.2 请求语法

```
DELETE /BucketName/Object_Key?record    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: 0
```

5.2.6.4.3 例子

若用户需要现在停止录像 key 为 plan_key 的录像对应的录像任务，那么协议例子如下：

```
DELETE /bucket_1/plan_key?record    HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: hikcstor::im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
Content-Length: 0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
Connection: close
```

5.2.6.5 查询任务对应状态

5.2.6.5.1 描述

用户给云存储下发开启录像之后，可以向云存储查询任务对应的状态，云存储会返回任务是否是正在执行或是未执行或是执行完成。用户可以根据任务具体的状态做其他的事情。

在云存储返回的协议体中，Status 的值为 0 表示任务未执行，1 表示任务正在执行，2 表示任务已经执行完成，3 表示任务已经停止，4 表示任务执行异常，5 表示任务无法执行(可能是用户输入的取流 URL 地址不正确)。

云存储会将手动录像任务保存 3 天（可配置），3 天后，无论任务成功与否，用户都不能再次查询该录像任务结果。故要求用户在下发录像任务后，及时的查看任务进度。手动补录任务没有 3 天后无法查询任务结果的限制。

5.2.6.5.2 请求语法

```
GET /BucketName/Object_Key?recordstatus    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: 0
```

5.2.6.5.3 例子

若用户需要现在查询录像 key 为 plan_key 的录像对应的录像任务的状态，那么协议例子如下：

```
GET /bucket_1/plan_key?recordstatus HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: hikcstor::im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
Content-Length: 0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
```

```
Content-Length: 512
Connection: close


<?xml version=”1.0” encoding=”UTF-8”?>
<QueryTaskStatus>
    <Status >1</ Status>
</ QueryTaskStatus>
```

5.2.6.6 查询手动录像任务列表

5.2.6.6.1 描述

用户给云存储下发开启录像之后，可以查询所有已经下发的手动录像的任务列表。云存储会返回对应的 key 的所有手动录像任务，以及任务对应的状态。

录像任务列表的查询以 bucket 为单位，不同的 bucket 需要多次获取。

若用户已经停止了手动录像任务，则云存储不会返回该任务。

5.2.6.6.2 请求语法

```
GET /BucketName?realtimelist    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: 0
```

5.2.6.6.3 例子

若用户需要查询 bucket1 下的所有已经下发的手动录像任务列表，那么协议例子如下：

```
GET /bucket1?realtimelist    HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: hikcstor::im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
Content-Length: 0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
```

```
Content-Length: 65535
Connection: close

<?xml version=”1.0” encoding=”UTF-8”?>
<RealtimeTaskList >
<TotalNum>2</TotalNum>
    <Task >
        <Key>Key1</Key>
        <StartExeTime >13645256984</ StartExeTime >
        <Status >0</Status>
    </ Task >
    < Task >
        <Key>Key2</Key>
        <StartExeTime >13645256984</ StartExeTime >
        <Status >0</Status>
    </ Task >
</ RealtimeTaskList >
```

其中云存储返回的协议体字段说明如下：

名称	描述
RealtimeTaskList	任务列表的容器。
TotalNum	本次检索出的任务的总条数
Task	任务容器
Key	任务对应的 Object 的 Key 值
StartExeTime	任务开始执行的时间点,该时间点为 Unix 时间戳类型,如 13645256984。
Status	任务对应的指定状态：0 表示正在执行，1 表示取流异常。 用户可以在下发手动录像任务后，查询任务列表，通过任务中的该字段判断云存储取流是否正常。

5.2.6.7 手动补录

5.2.6.7.1 描述

云存储支持从前端或者其他设备中获取过去的时间段数据进行存储，用户在下发计划时，需提供获取的历史数据的开始时间与结束时间，且开始时间与结束时间需包含在取流 URL 地址中，如 dvr://127.0.0.1:8000/0/channels/1?range=20140303000000-20140304000000，该 URL 中的 range 参数表示获取开始时间为 2014 年 3 月 3 日 0 点 0 分 0 秒，结束时间为 2014 年 3 月 4 日 0 点 0 分 0 秒的数据，并将其存储到云存储中。

用户需确保取流 URL 地址正确，否则云存储无法正常取流。

用户需确保协议中的 object key 的唯一性，否则云存储可能会出现录像数据交叉或者覆盖的问题。

5.2.6.7.2 请求语法

```
POST /BucketName/Object_Key? history HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: length of data

<?xml version=”1.0” encoding=”UTF-8”?>
<HistoryPlan>
    <StreamDevInfo>
        <UserName></UserName>// 用户名
        <PassWord></PassWord>// 密码
        <StreamURL></StreamURL>// 取流 URL 地址信息 开始时间与结束事件在 URL 中
    </StreamDevInfo>
</HistoryPlan>
```

其中参数说明如下：

Name	Description	是否必须
UserName	用户名	是
PassWord	密码	是
StreamURL	取流 URL 地址(用户确保 URL 正确性)	是

5.2.6.7.3 例子

若用户需要云存储提取，取到的流存入名称为 bucket_1 的 bucket 中，数据追加至 key 为 plan_key 的对象中，那么协议例子如下：

```
POST /bucket_1/plan_key? history HTTP/1.1
Host: object.hikcstor.com
Date: Wed, 12 Dec 2014 12:00:00 GMT
Authorization: hikcstor::im09ttopcc2a5qybj:dFrR+YSSAWodMz3p85i/6Xw3GKI=
Content-Length: 172

<?xml version=”1.0” encoding=”UTF-8”?>
<HistoryPlan>
    <StreamDevInfo>
        <UserName>root</UserName>// 用户名
        <PassWord>123</PassWord>// 密码
        <StreamURL> dvr://127.0.0.1:8000/0?range=20140303000000-20140304000000</StreamURL>
    </StreamDevInfo>
</HistoryPlan>
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 512
Connection: close
```

5.2.6.8 视频文件上传

5.2.6.8.1 描述

用户在上传视频文件前需要先获取一个最优节点,然后用户需向该最优节点打开一个视频文件上传通道，该最优节点会返回一个通道 ID，然后用户可以持续往该通道写入数据，若用户所有数据上传完成，则需向该最优节点关闭通道。

若用户在上传过程中出现错误，则需向云存储重新申请一个最优节点，且向该节点打开上传通道，然后将剩下的数据持续写入该通道即可。

用户在打开通道时还可以指定云存储的帧分析与转码动作。

用户上传一个完整视频文件的流程大致如下：

1、 申请最优节点(参考 5.2.6.1)
2、 向最优节点打开上传通道
3、 往通道持续写入数据
4、 所有数据写入完成关闭通道

用户可以用同一个 key 上传不同的流数据，云存储会将这些不同的数据自动视为同一个 object 的数据。若视频文件指定的 key 与普通文件 key 重复则会将普通文件 key 覆盖。

5.2.6.8.2 请求语法

- 打开上传通道：

```
POST / BucketName/Object_Key?stream&act=your_act  HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:0
```

其中参数说明如下：

Name	Description	是否必须
act	表示上传时的后续流需要云存储的处理动作。0 表示无动作，1 表示帧分析，若不赋值该参数，默认值为 0	否

- 往通道中持续写入数据流

```
POST / BucketName/Object_Key?stream&streamid=your_streamid  HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length: length of data


[length of data]
```

其中：streamid 参数的值由打开上传通道时云存储返回

- 关闭流通道

```
DELETE / BucketName/Object_Key?stream&streamid=your_streamid  HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:0
```

其中：streamid 参数的值由打开上传通道时云存储返回

5.2.6.8.3 例子

如果用户想要在 bucket 名字为 test 中上传一个 key 为 test.mp4 的一个视频文件，且需要云存储进行帧分析且转码。则交互协议如下：

- 打开上传通道：

```
POST / test/test.mp4?stream&act=1    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length:0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 342

<?xml version=”1.0” encoding=”UTF-8”?>

<OpenStreamResult >

<StreamID >124 </ StreamID >

</ OpenStreamResult>
```

- 往通道中持续写入数据流

```
POST / test / test.mp4?stream&streamid=124    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length: length of data


[length of data]
```

其中：streamid 参数的值由打开上传通道时云存储返回

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
```

- 关闭流通道

```
DELETE  / test / test.mp4?stream&streamid=124  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length:0
```

其中：streamid 参数的值由打开上传通道时云存储返回

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
```

5.2.6.9 转码任务下发

5.2.6.9.1 描述

用户若需要对某个视频文件进行转码，则只需给云存储下发转码任务即可，用户在下发转码任务之后，可以定时查询转码的进度。

5.2.6.9.2 请求语法

```
POST / BucketName/Object_Key?transcode  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: length of data
```

```
<?xml version=”1.0” encoding=”UTF-8”?>
<TranscodeTask>
    <TranscodeParam>
        <OutputBucket></ OutputBucket >转码之后存储的 Bucket
        <OutputKey></OutputKey>//转码之后的 object key
        <PacketType></PacketType>//封装格式 0-保持原有，1-海康封装，2-MPEG2-PS，3-MPEG2-TS，4-RTP，5-MP4/3GPP/MOV，6-AVI
        <VideoEncodeFormat></VideoEncodeFormat>// 视频封装格式 0-保持原有，1-HIK264，2-mpeg2，3-mpeg4，4-H264，5-MJPEG，6-MS_MPEG4V1/2/3，7-WMV2
        <AudioEncodeFormat></AudioEncodeFormat>//音频封装格式 0-保持原有，1-MEPGA，2-AAC，3-G.711，4-G.722，5-G.7231，6-G.726，7-G.729，8-AMR_NB，9-WNAV2，10-PCM
        <FrameWidth></FrameWidth>
        <FrameHeight></FrameHeight>
    </TranscodeParam>
    <TranscodeParam>
        <OutputBucket></ OutputBucket >转码之后存储的 Bucket
        <OutputKey></OutputKey>//转码之后的 object key
        <PacketType></PacketType>//封装格式 0-保持原有，1-海康封装，2-MPEG2-PS，3-MPEG2-TS，4-RTP，5-MP4/3GPP/MOV，6-AVI
        <VideoEncodeFormat></VideoEncodeFormat>// 视频封装格式 0-保持原有，1-HIK264，2-mpeg2，3-mpeg4，4-H264，5-MJPEG，6-MS_MPEG4V1/2/3，7-WMV2
        <AudioEncodeFormat></AudioEncodeFormat>//音频封装格式 0-保持原有，1-MEPGA，2-AAC，3-G.711，4-G.722，5-G.7231，6-G.726，7-G.729，8-AMR_NB，9-WNAV2，10-PCM
        <FrameWidth></FrameWidth>
        <FrameHeight></FrameHeight>
    </TranscodeParam>
    </TranscodeParam >
        ….
    </ TranscodeParam >
</TranscodeTask>
```


其中参数说明如下：

Name	Description	是否必须
OutputBucket	转码之后存储的 Bucket。	是
OutputKey	转码之后的文件对应的新的 key 值，该值由用户进行定义，且需保证唯一，否则可能会覆盖之前的 key 的数据。	是
PacketType	输出封装格式，0-保持原有，1-海康封装，2-MPEG2-PS， 3-MPEG2-TS， 4-RTP， 5-MP4/3GPP/MOV，6-AVI	是
VideoEncodeFormat	输出视频封装格式，0-保持原有，1-HIK264，2-mpeg2，3-mpeg4，4-H264，5-MJPEG，6-MS_MPEG4V1/2/3，7-WMV2	是
AudioEncodeFormat	输出音频封装格式，0-保持原有，1-MPGA，2-AAC，3-G.711，4-G.722，5-G.7231，6-G.726，7-G.729，8-AMR_NB，9-WNAV2，10-PCM	是
FrameWidth	帧宽	是
FrameHeight	帧高	是

5.2.6.9.3 例子

如用户想对 object key 为 mp4 的对象进行转码，则类似请求协议如下：

```
POST / bucket_1/mp4?transcode  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
Content-Length: length of data
```

```
<?xml version=”1.0” encoding=”UTF-8”?>
<TranscodeTask>
    <TranscodeParam>
        <OutputKey>transcode_mp4OutputKey>
        <PacketType>1/PacketType>
        <VideoEncodeFormat>0/VideoEncodeFormat>
        <AudioEncodeFormat>0/AudioEncodeFormat>
        <FrameWidth>1024ameWidth>
        <FrameHeight>720meHeight>
    </TranscodeParam>
</TranscodeTask>
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 0
```

5.2.6.10 转码进度查询

5.2.6.10.1 描述

发转码任务成功之后，可以定时查询转码进度，若转码进度查询到 100，则用户可以下载该转码之后的数据文件。

云存储会将转码结果保存 3 天（可配置），3 天后，无论转码成功与否，用户都不能再次查询该转码任务结果。故要求用户在下发转码任务后,及时的查看转码进度。

5.2.6.10.2 请求语法

```
GET / BucketName/Object_Key?transcode    HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
```

```
Content-Length: 0
```

5.2.6.10.3 例子

若用户下发了 object key 为 test_transcode 的转码任务，则可以定时查询该 key 对应的转码进度。

```
GET /bucket_1/test_transcode?transcode    HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length: 0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 342

<?xml version=”1.0” encoding=”UTF-8”?>
<QueryTranscodeResult>
    <OutputResult>
        <OutputBucket></ OutputBucket >
          <OutputKey></OutputKey>
          < Progress >60 </ Progress >
          <Result>1</Result>
    </OutputResult>
    <OutputResult>
        <OutputBucket></ OutputBucket >
          <OutputKey></OutputKey>
          < Progress >60 </ Progress >
          <Result>1</Result>
    </OutputResult>
```

```
</ QueryTranscodeResult >
```

其中：Progress 字段表示具体转码进度

Result 字段表示转码结果。0:表示转码失败(此时 Progress 字段值为 0) 1:表示任务正在排队执行 2:表示转码成功 3 :表示任务正在执行 4:表示任务不存在(此时 Progress 字段值为 0)

5.2.6.11 删除转码任务

5.2.6.11.1 描述

下发转码任务成功之后，用户可以手动删除转码任务。

5.2.6.11.2 请求语法

```
DELETE / BucketName/Object_Key?transcode HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length: 0
```

5.2.6.11.3 例子

若用户下发了 object key 为 test_transcode 的转码任务，则可以删除该 key 对应的转码。

```
DELETE /bucket_1/test_transcode?transcode HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string

Content-Length: 0
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 342
```

5.2.6.12 获取视频文件的时间段信息

5.2.6.12.1 描述

用户在上传完某个视频文件之后，若要求云存储进行了帧分析，可以指定时间范围，查询指定时间范围内的时间段信息。

5.2.6.12.2 请求语法

```
GET /BucketName/Object_Key? timesegment&starttime=&endtime=  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
```

其中参数说明如下：

Name	Description	是否必须
starttime	请求查询时间段的开始时间,类型为 Linux 时间戳类型(time_t)。如 1447729200。	是
endtime	请求查询时间段的结束时间,类型为 Linux 时间戳类型(time_t)。如 1447729300。	是

5.2.6.12.3 例子

用在 bucket 名称为 test_bucket 中上传了一个名为 mp4 的对象，如果要求云存储进行帧分析，且要求查询开始时间为 1447729200，结束时间为 1447729300 时间段内的时间段信息。

```
GET / test_bucket/mp4? timesegment &starttime=1447729200&endtime=1447729300  HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 65534

<?xml version = "1.0" encoding = "UTF-8"?>
    <QueryTimeSegment>
        <Bucket>testa</Bucket>
        <Key>200streamcopy3</Key>
        <SegmentList>
            <Segment>
                <StartTime>1261011564</StartTime>
                <EndTime>1261011658</EndTime>
                <DataSize>524288</DataSize>
            </Segment>
        </SegmentList>
    </QueryTimeSegment>
```

5.2.6.13 通过时间段获取视频文件数据

5.2.6.13.1 描述

用户在上传完某个视频文件之后，若要求云存储进行了帧分析，可以要求按照视频文件的开始时间与结束时间提取数据。

5.2.6.13.2 请求语法

```
GET / BucketName/Object_Key?stream&starttime=&endtime=&outputformat=  HTTP/1.1

Host: object.hikcstor.com

Date: date

Authorization: authorization string
```

其中参数说明如下：

Name	Description	是否必须
starttime	时间段的开始时间,类型为 Linux 时间戳类型(time_t)。如 1447729200。	是
endtime	时间段的结束时间,类型为 Linux 时间戳类	是

	型(time_t)。如 1447729300。	
outputformat	表示云存储需要以什么格式返回数据。 0：默认 1：海康文件层 2：ps 文件层 3：tx 文件层 4：rtp 文件层 5. MPEG4 文件层 若不指定，则以原有格式返回数据。	否

5.2.6.13.3 例子

用在 bucket 名称为 test_bucket 中上传了一个名为 mp4 的对象，如果要求云存储进行帧分析，且要求查询开始时间为 1447729200，结束时间为 1447729300 时间段内的录像。

```
GET / test_bucket/mp4?streamfile&starttime=1447729200&endtime=1447729300&outputformat=5
HTTP/1.1
Host: object.hikcstor.com
Date: date
Authorization: authorization string
```

回复：

```
HTTP/1.1 200 OK
Date: Wed, 12 Dec 2014 12:00:00 GMT
Content-Length: 65534

[length of data]
```

5.2.6.14 海康视频文件录像头上传

5.2.6.14.1 描述

用户可以针对海康视频文件单独上传 40 字节录像头，上传协议与普通小对象上传并无区别，用户在上传时可以将录像头的 key 按照一定规则关联具体录像数据的 key。如上传的视频文件的 key 为 test.mp4。那

么录像头的 key 用户可以指定为 test.mp4.header。用户在获取视频文件对应的头时可以直接按照这种方式先获取视频文件对应的头数据。

5.2.6.14.2 请求语法

参考 5.2.4.1

5.2.6.14.3 例子

参考 5.2.4.1

5.2.6.15 海康视频文件录像头获取

5.2.6.15.1 描述

获取录像头数据的协议与普通下载文件并无区别。详细参考 5.2.4.3。

5.2.6.15.2 请求语法

详细参考 5.2.4.3。

5.2.6.15.3 例子

详细参考 5.2.4.3。

6. 常见业务流程说明

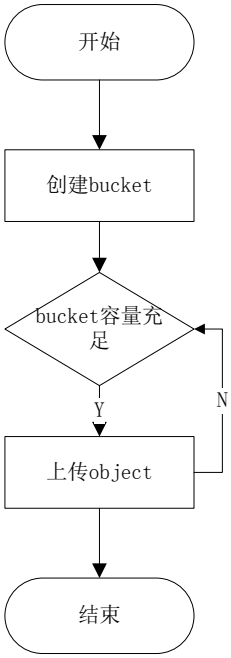
6.1 文件上传

6.1.1小文件上传

6.1.1.1 处理流程

用户在上传普通文件前，应先保证自己已经创建了对应的 bucket，且 bucket 中剩余容量充足，上传文件接口比较简单，详细接口参考 5.2.4.1。

普通文件上传流程如下：



关键流程点说明：

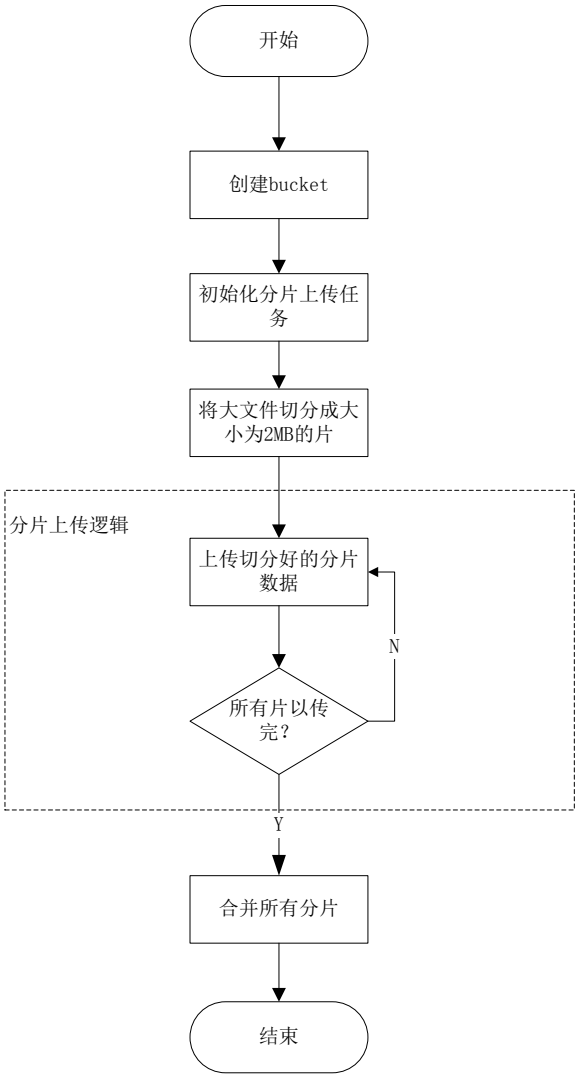
1、 上传 object 前需确保拥有 bucket 的写权限，且 bucket 容量充足

2、 上传完成后，该 object 对应的 URL 即"/" + bucket 名字 + "/" + object key，该 URL 可以直接用于提取已经上传的 object

6.1.2大文件上传

6.1.2.1 处理流程

大对象上传可以利用分片上传功能进行实现。分片上传功能支持将一个文件切割为一系列特定大小的数据片，分别将这些小数据上传到服务端，全部上传完后再在服务端将这些小数据合并为一个大的 object 资源。分片上传的基本流程如下：



其中关键要点如下：

1、 将上传的文件按 2MB 大小切分成若干片，若最后一个片按实际大小进行传输

2、 用户在上传大对象时需先初始化一个分片上传任务(参考 5.2.5.1)

3、 初始化分片上传任务成功后才能上传具体分片数据，分片数据的编号与其在文件中的偏移一一对应，且分片编号需单调递增(参考 5.2.5.2)

4、 当所有分片上传成功后，即可将所有已经上传的分片合并为一个完整的大对象(参考 5.2.5.4)

5、 当某个分片上传失败后，海康云存储支持重新上传该分片(参考 5.2.5.2)

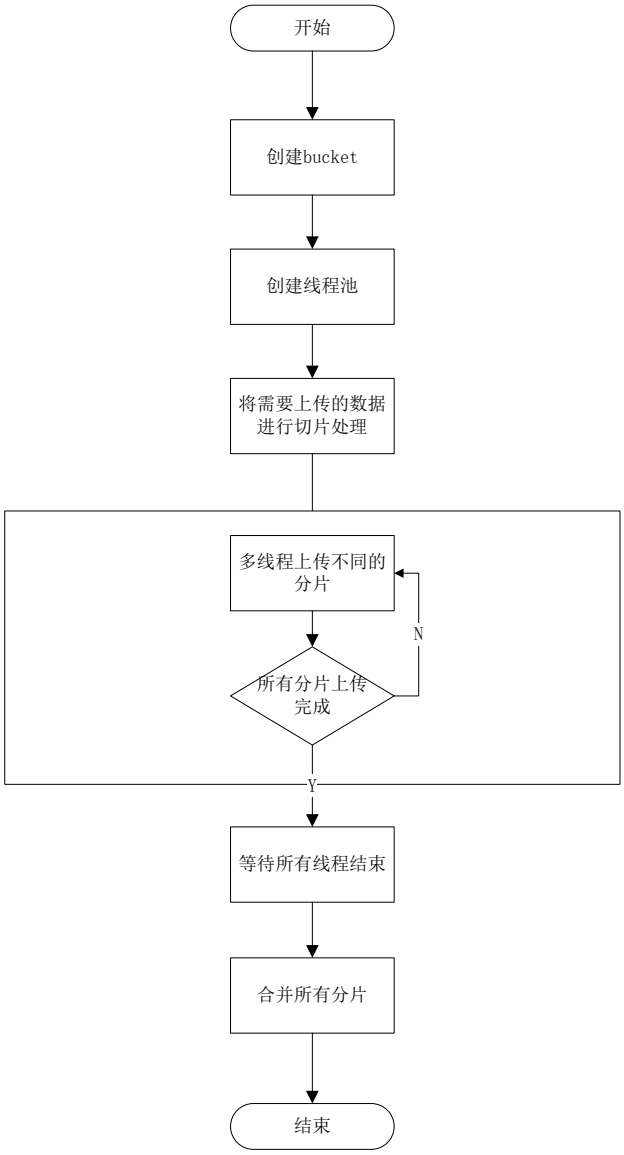
6、 若用户想终止某个分片任务，调用终止分片任务接口即可(参考 5.2.5.3)

7、 若用户需要查看自己已经上传成功了那些分片，则只需调用列出已经上传成功的所有 part 接口即可(参考 5.2.5.5)

6.1.2.2 并发上传

海康云存储天然支持用户并发上传某个大对象，用户将需要上传的数据切分好数据片之后，可以开启多个线程，每个线程上传不同的分片数据，最后当线程上传完所有的分片数据之后，调用合并所有分片将分片合并为一个完整的对象。达到并发上传的目的。

其中并发上传的一般处理流程如下：



其中关键点如下：

1、多线程上传用户需确保每个线程上传的是不同的分片数据

2、若用户想终止某个分片任务，调用终止分片任务接口即可(参考 5.2.5.3)

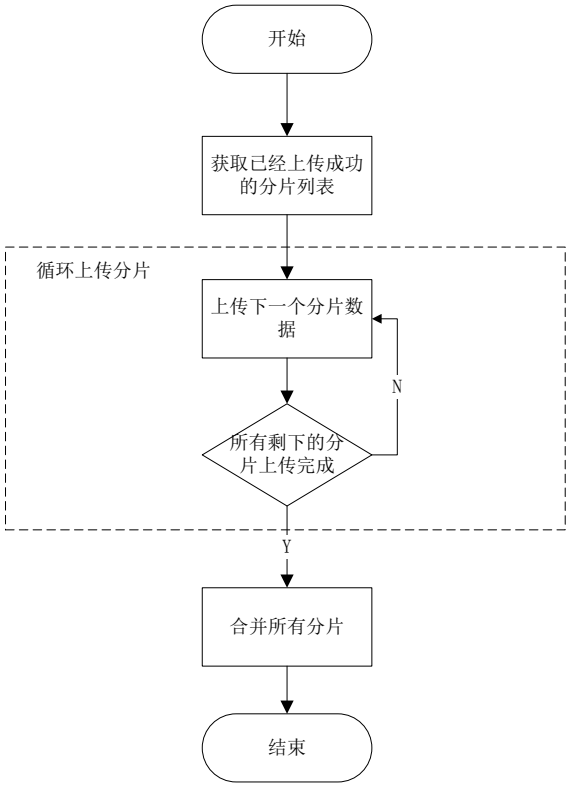
3、若用户需要查看自己已经上传成功了那些分片，则只需调用列出已经上传成功的所有 part 接口即可(参考 5.2.5.5)

6.1.2.3 断点续传

若用户需要上传的对象很大，但是本地网路环境非常恶劣，或者用户在上传过程中客户端程序发生了崩溃。这种情况在实际应用中非常常见，而海康云存储在用户使用分片上传时，天然支持断点续传。

用户进程崩溃或用户上传某个分片数据失败后，用户可以按照上次上传成功的分片后上传下一个分片。若用户未记录上传成功的分片信息，可以通过接口获取已经上传成功的分片列表(参考 5.2.5.5)。这样用户就能按照已经上传成功的信息，开始继续上传其他分片数据，直到所有分片上传成功为止，在进行合并所有分片，这样一个大对象就上传完成，也能实现断点续传的功能。

断点续传一般流程如下：



关键点如下：

1、 用户若未记录上次上传成功的分片信息，可以从云存储中获取已经上传成功的分片列表(参考 5.2.5.5)

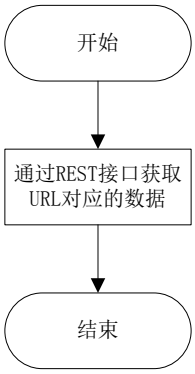
6.2 文件下载

6.2.1处理流程

若用户数据上传成功后，用户可以通过 HTTP 方式直接提取对应数据，而数据对应的 URL 就是用户上传时指定的 bucket 与 object 的 key 的拼接。拼接方式如下：”/” + bucket 名字 + “/” + object key。

如用户在名为 test 的 bucket 中上传了一个 key 为 upload_file.txt 的对象，那么这个对象对应的 URL 则为/test/upload_file.txt，用户在提取数据时，只需通过 URL 即可进行数据的提取。

数据提取流程图如下：



关键点如下：

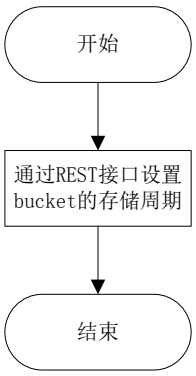
2015、 若用户无该 bucket 的读权限将获取数据失败

6.3 存储周期设置

6.3.1处理流程

海康云存储支持 bucket 级别的循环覆盖，用户可以设置 bucket 的存储周期(单位为天)，该 bucket 下的 object 从上传到海康云存储的那刻算起，若存储时长超过了 bucket 设置的存储周期，海康云存储会自动将该 object 删除。从而达到周期覆盖的目的。

Bucket 的存储周期设置只能由其拥有者发起，否则海康云存储会返回错误。设置 bucket 的存储周期流程较为简单，其流程图如下：



其中关键点如下：

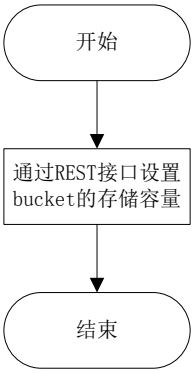
2015、 接口调用者需确保需修改的 bucket 是由自己创建

6.4 用户存储容量管理

6.4.1处理流程

海康云存储中的容量管理为 bucket 级别，用户可以按需要增加/缩减 bucket 的容量(单位为 GB)，若 bucket 下面存储的 object 总大小超过 bucket 的总容量，则用户后续针对该 bucket 的上传操作会失败。

若用户需要修改某个 Bucket 的容量，需确保该 bucket 由自己创建，否则海康云存储将会返回错误。设置 Bucket 的存储容量接口较为简单，流程如下：



其中关键点如下：

2015、 接口调用者需确保需修改的 bucket 是由自己创建