目录

API概览

KS3请求相关

API	接口功能
请求签名	向KS3发起携带签名的请求
请求签名V4	向KS3发起携带V4签名的请求
公共请求头	KS3常用的请求头部
公共响应头	KS3常用的响应头部
错误码	可能出现的错误码列表

Service相关

API	接口功能
Get Service	返回请求者拥有的所有的存储空间

Bucket相关

API	接口功能
HEAD Bucket	判断某一存储空间是否存在，以及用户对其的访问权限
DELETE Bucket	删除给定的存储空间
GET BucketACL	返回存储空间的访问权限控制列表(Access control list，简称ACL)
PUT Bucket	创建存储空间
PUT BucketACL	设置存储空间的ACL
GET Bucket	列出指定空间中的部分或全部(上限1000)的对象
GET Bucket logging	返回用户存储空间的日志记录状态
PUT Bucket logging	配置用户存储空间的日志参数、指定允许查看/修改日志参数的用户
GET Location	返回用户存储空间的区域
List Multipart Uploads	返回存储空间下所有正在进行分块上传的任务
Put Bucket Policy	为指定存储空间添加空间策略(Bucket Policy)
Get Bucket Policy	获取指定存储空间的空间策略(Bucket Policy)
Delete Bucket Policy	删除指定存储空间的空间策略(Bucket Policy)
Put Bucket Lifecycle	为指定存储空间添加生命周期规则
Get Bucket Lifecycle	获取指定存储空间的生命周期规则配置
Delete Bucket Lifecycle	删除指定存储空间的生命周期规则
Get Bucket Replication	获取源存储空间的跨区域复制规则
Put Bucket Replication	为源存储空间设置跨区域复制规则
Delete Bucket Replication	关闭源存储空间的跨区域复制规则

Object相关

API	接口功能
DELETE Object	删除对象
GET Object	获取对象
HEAD Object	返回对象的元数据信息
PUT Object	添加一个对象到某个存储空间
POST Object	使用HTML POST表单添加一个对象到某个存储空间
GET Object ACL	返回对象的ACL
PUT Object ACL	设置对象的ACL
PUT Object Copy	将KS3中某个对象拷贝到指定存储空间中
PUT Fetch	从第三方URL拉取文件，并且上传到KS3指定存储空间中
Restore Object	解冻归档存储类型的对象
Delete Object Tagging	删除指定对象的所有标签信息
Get Object Tagging	获取指定对象的所有标签信息
Put Object Tagging	设置或者更新对象的标签信息

分块上传相关

API	接口功能
Initiate Multipart Upload	启动分块上传任务
Upload Part	在分块上传任务中，上传一个块
Complete Multipart Upload	完成分块上传任务，即将所有的块组装成一个对象
Abort Multipart Upload	放弃一个分块上传任务
List Parts	列出指定分块上传任务中已上传的块
Upload Part Copy	通过拷贝已存在的对象的方式实现上传一个块
List Multipart Uploads	返回存储空间下所有正在进行分块上传的任务

对象标签相关

API	接口功能
Delete Object Tagging	删除指定对象的所有标签信息
Get Object Tagging	用于获取对象的标签信息
Put Object Tagging	用于设置或更新对象的标签信息。

跨域资源共享(CORS)相关

API	接口功能
GET Bucket CORS	获取存储空间的CORS信息
PUT Bucket CORS	配置存储空间的CORS信息
DELETE Bucket CORS	删除指定存储空间的CORS配置
OPTIONS Object	获取存储空间的CORS信息

异步数据处理相关

API	接口功能
上传回调处理(Upload Callback Processing)	在调用PUT Object、Complete Multipart Upload API时，携带相关的Callback参数，实现上传回调处理（Upload CallBack Processing，简称UCP）

加密相关

API	接口功能
加密相关	KS3提供的两种服务器端加密的方式的使用方式

低频存储相关

API	接口功能
Infrequent Access	KS3低频存储的相关接口

归档存储相关

API	接口功能
ARCHIVE	KS3归档存储的相关接口
Restore Object	解冻归档存储类型的对象

AWS S3协议兼容介绍

API	接口功能
AWS S3协议兼容	KS3 API与AWS S3协议的兼容性对比

请求签名V2

当用户请求KS3时，可以使用AccessKey和SecretKey对请求做签名，当KS3收到带签名信息的请求之后，将使用相同的算法验证签名，如果发现签名不一致，KS3将会返回403给用户。如果KS3验证签名一致，且AccessKey对应的用户有权限操作请求的资源，则请求成功，否则KS3返回403。 如果用户请求KS3时，在请求中没有携带签名信息，那么KS3认为该请求是匿名的。当KS3接收到匿名请求时，如果发现用户请求的资源不允许匿名请求，将会返回403。

KS3提供了可视化签名工具，方便客户调试签名错误，快速定位问题。

1. 通过 HTTP 请求 Header 发送签名

方法: 在请求中加入名为 Authorization 的 Header，值为签名值。如下：

```
Authorization: KSS P3UPCMORAFON76Q6RTNQ:vU9XqPLcXd3nWd1fLWIhruZrLAM=
```

签名（Authorization）计算方法：

```
Authorization = “KSS YourAccessKey:Signature”

Signature = Base64(HMAC-SHA1(YourSecretKey, UTF-8-Encoding-Of( StringToSign ) ) );

StringToSign = HTTP-Verb + "\n" +
Content-MD5 + "\n" +
Content-Type + "\n" +
Date + "\n" +
CanonicalizedKssHeaders+
CanonicalizedResource;
```

Content-MD5, Content-Type, CanonicalizedKssHeaders可为空， 若为空，则使用空字符串("")替代, HTTP-Verb、Date和CanonicalizedResource不能为空

- HTTP-Verb 表示请求的方法，如：GET\PUT\POST\DELETE等
- Content-MD5 表示请求内容数据的MD5值，使用Base64编码。当请求的header中包含Content-MD5时，需要在StringToSign中包含，否则用("")替代。 注意：Content-MD5的算法为先对数据做MD5摘要，再将MD5摘要做Base64编码。中间不需要做HEX编码，由于部分语言或工具包的MD5是默认做HEX编码的，所以当MD5算出来的结果为HEX编码的，首先需要对算出来的结果做HEX解码，然后再做Base64编码。详解RFC2616
- Content-Type 表示请求内容的类型，取HTTP header中的Content-Type
- Date 表示此次操作的时间,且必须为 HTTP1.1 中支持的 GMT 格式。取HTTP Header中的Date,如果该时间与KS3服务端时间相差15分钟以外，将导致KS3返回403。比如：Wed, 17 Feb 2012 15:31:56 GMT

  注意： 有的客户端不支持发送Date请求头。这种情况下，计算签名时需要保持Date字段的同时，在CanonicalizedKssHeaders中加入x-kss-date，格式与Date一致。发送请求时，需要添加x-kss-date请求头。详见示例。

- CanonicalizedKssHeaders 表示HTTP请求中的以x-kss开头的Header组合，详见CanonicalizedKssHeaders计算方法。
- CanonicalizedResource 表示用户访问的资源，详见CanonicalizedResource的计算方法。

CanonicalizedKssHeaders计算方法

计算方法如下 ：

1. 先选出所有以x-kss-开头的 HTTP 请求Header，并把Header Name全部转成小写。如： X-KSS-Meta-Myname: Jack把Header Name转成小写后为x-kss-meta-myname: Jack
2. 将这些Header按Header Name的名字的字典序进行升序排列
3. 删除请求头和内容之间分隔符两端出现的任何空格。如x-kss-meta-myname: Jack转换为：x-kss-meta-myname:Jack
4. 将这些Header Name， Header Value对使用"\n"分隔符连接在一起。

   注意：

   - CanonicalizedKssHeaders为空，无需添加最后的\n。
   - 如果只有一个，则需要在最后添加\n，例如：x-kss-meta-yourname:Lee\n
   - 如果有多个，则使用使用"\n"分隔符连接在一起，并且在最后添加\n，例如：x-kss-meta-myname:Jack\nx-kss-meta-yourname:Lee\n
   - 如果客户端不支持发送 Date 请求头时，在计算CanonicalizedKssHeaders时必须增加 x-kss-date 请求头。详见示例。

CanonicalizedResource的计算方法

CanonicalizedResource 代表了请求的目标资源,结构如下:

```
/[BucketName/[ObjectKey[?SubResource]]]
```

- BucketName:用户请求的Bucket名称。
- ObjectKey:用户请求的Object名称,需要对Object名称做URL编码。
- SubResource:用户请求的子资源。把URL参数中的"acl","lifecycle","location","logging","notification","partNumber","policy","requestPayment","torrent","uploadId","uploads","versionId","versioning","versions","website","content-type","response-content-language","response-expires","response-cache-control","response-content-disposition","response-content-encoding"筛选出来，将这些查询字符串及其请求值(不做URL编码的请求值)按照字典序，从小到大排列，以&为分隔符排列，即可得到SubResource。

计算方法如下：

1. CanonicalizedResource="/"
2. 如果BucketName不为空,则 CanonicalizedResource = CanonicalizedResource + BucketName + "/"
3. 如果ObjectKey不为空,则 CanonicalizedResource = CanonicalizedResource + ObjectKey
4. 替换CanonicalizedResource中的双斜杠("//")为"/%2F"
5. 如果SubResource不为空，则CanonicalizedResource = CanonicalizedResource + "?" + SubResource

1. 计算签名的例子：

示例中的ObjectKey为经过URL编码的ObjectKey

```
PUT /{BucketName}/{ObjectKey} HTTP/1.0
Content-Md5: 1B2M2Y8AsgTpgAmY7PhCfg==
Content-Type: text/html
Content-Length: 1024
Date: Wed, 17 Feb 2012 15:31:56 GMT
Host: ks3-cn-beijing.ksyun.com
```

假设 SecretKey 为：Ik9OeHJ6eElzZnBGakE3U3dQeklMd3k,其签名算法为：

```
import base64
import hmac
from hashlib import sha1
h = hmac.new("Ik9OeHJ6eElzZnBGakE3U3dQeklMd3k", "PUT\n1B2M2Y8AsgTpgAmY7PhCfg==\ntext/html\nWed, 17 Feb 2012 15:31:56 GMT\n/{BucketName}/{ObjectKey}", sha1)
Signature = base64.encodestring(h.digest()).strip()
```

2. 客户端不支持发送Date请求头的例子

计算签名时需要保持Date字段的同时，在CanonicalizedKssHeaders中加入x-kss-date，格式与Date一致。发送请求时，需要添加x-kss-date请求头。示例中的ObjectKey为经过URL编码的ObjectKey。

```
PUT /{BucketName}/{ObjectKey} HTTP/1.0
Content-Md5: 1B2M2Y8AsgTpgAmY7PhCfg==
Content-Type: text/html
Content-Length: 1024
Date: Wed, 17 Feb 2012 15:31:56 GMT
Host: ks3-cn-beijing.ksyun.com
x-kss-date: Wed, 17 Feb 2012 15:31:56 GMT
```

假设 SecretKey 为：Ik9OeHJ6eElzZnBGakE3U3dQeklMd3k,其签名算法为：

```
import base64
import hmac
from hashlib import sha1
h = hmac.new("Ik9OeHJ6eElzZnBGakE3U3dQeklMd3k", "PUT\n1B2M2Y8AsgTpgAmY7PhCfg==\ntext/html\nWed, 17 Feb 2012 15:31:56 GMT\nx-kss-date:Wed, 17 Feb 2012 15:31:56 GMT\n/{BucketName}/{ObjectKey}", sha1)
Signature = base64.encodestring(h.digest()).strip()
```

3. 通过 URL QueryString 发送签名

携带签名的URL示例：

```
https://ks3-cn-beijing.ksyun.com/{BucketName}/{ObjectKey}?KSSAccessKeyId=VSDNT6SHFNDWBXYZRS3A&Expires=1435550417&Signature=a2JnaLMuN%2FWmcKL%2FW4aibMCa4BY%3D
```

KSSAccessKeyId为用户的AccessKey

Expires为该链接的过期时间，使用Unix_Time表示

Signature的计算法同上，只是把Date换成Expires值

请求签名V4

使用对象存储服务（KS3） 时，可通过 RESTful API 对 KS3 发起 HTTP 匿名请求或 HTTP 签名请求，对于签名请求，KS3服务器端将会进行对请求发起者的身份验证。

- 匿名请求：HTTP 请求不携带任何身份标识和鉴权信息，通过 RESTful API 进行 HTTP 请求操作。
- 签名请求：HTTP 请求时携带签名，KS3服务器端收到消息后，进行身份验证，验证成功则可接受并执行请求，否则将会返回错误信息并丢弃此请求。

KS3 V4签名兼容 AWS Signature Version 4，签名方法见Authenticating Requests - AWS Signature Version 4。

注意：

本文所描述的 API 请求签名，如果您使用 SDK 进行开发，SDK中会包含计算方法。仅在您希望通过原始 API 进行二次开发时，需要根据本文所描述步骤进行操作。

V4 签名流程

1. StringToSign计算

1.1 CanonicalRequest计算方法

计算公式

```
CanonicalRequest = HTTPRequestMethod + '\n'
        + CanonicalURI + '\n'
        + CanonicalQueryString + '\n'
        + CanonicalHeaders + '\n'
        + SignedHeaders + '\n'
        + HexEncode(Sha256Hash(RequestPayload))
```

说明：

HTTPRequestMethod

首先是 HTTP 请求方法:GET、PUT、POST、DELETE等

CanonicalURI

/[BucketName]/[ObjectKey]

- 对请求资源进行urlencode，'/'不做encode
- 不包含?及后面的queryString
- urlencode参考URLencode方法;
- bucketName作为host一部分，文件名字为空时，请求资源为'/'
- 查询bucket 列表接口，请求资源为'/'

CanonicalQueryString

添加规范查询字符串，后跟换行符。

如果请求不包括查询字符串，请使用空字符串填充（实际上是空白行）

要构建规范查询字符串，请完成以下步骤：

- 按字符代码升序顺序对参数名称进行排序。具有重复名称的参数应按值进行排序。例如，以大写字母 F 开头的参数名称排在以小写字母 b 开头的参数名称之前。
- 根据以下规则对每个参数名称和值进行 URI 编码。
- 以排序后的列表中第一个参数名称开头，构造规范查询字符串。
- 对于每个参数，追加 URI 编码的参数名称，后跟等号字符 (=)，再接 URI 编码的参数值。对没有值的参数使用空字符串。
- 在每个参数值后追加与字符 (&)，列表中最后一个值除外。

计算公式如下:

```
UriEncode("query1")+"="+UriEncode("value1")+"&"+
UriEncode("query2")+"="+UriEncode("value2") + "&" +
...
UriEncode("queryn")+"="+UriEncode("valuen")
```

CanonicalHeaders

添加规范请求header，每个请求header健值对之间用"\n"进行分割，健值用‘:’连接

构造请求前需要对header进行排序，按照字符升序顺序对header名称进行排序

header名称需要转换为小写字母，value需要做trim

构造公式如下：

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

The CanonicalHeaders 列表如下:

- 必须包含host头
- 如果请求头有Content-Type，必须添加到CanonicalHeaders
- 任何以x-kss-开头的请求头，需要添加到CanonicalHeaders中。
- 临时授权，需要添加x-kss-security-token
- x-kss-content-sha256是必须的。
- 除了上述列出的必须增加的头，只计算SignedHeaders中包含的头
- 注意 Authorization不参与签名内容计算

SignedHeaders

防止用户请求header被篡改，列出参与鉴权运算的鉴权头，列表和CanonicalHeaders 一致，中间用‘;’号进行分割。运算格式为

```
Lowercase(<HeaderName1>)+";"+Lowercase(<HeaderName2>)+";"+...+Lowercase(<HeaderNameN>)
```

时间戳（x-kss-date或Date请求头）15分钟内有效，没有权限的用户可通过截获已签名的请求，并篡改SignedHeaders中没有包含的部分，所以建议您签名所有请求头和请求体

x-kss-content-sha256

x-kss-content-sha256 请求头是必须的，取值范围如下：

值	描述
HexEncode(Sha256Hash(RequestPayload))	请求正文经过哈希处理的以小写十六进制字符串表示形式
UNSIGNED-PAYLOAD	忽略请求正文的has校验
STREAMING-AWS4-HMAC-SHA256-PAYLOAD	chunk传输校验，基于v4的新接口协议，暂不支持

计算方法为

```
HashedPayload = Lowercase(HexEncode(SHA256Hash(requestPayload)))
```

使用 SHA256 等哈希（摘要）函数以基于 HTTP 或 HTTPS 请求正文中的负载创建哈希值。签名版本 4 不需要您使用特定字符编码来对负载中的文本进行编码。

在创建待签字符串后，请指定用于对负载进行哈希处理的签名算法。例如，如果您使用的是 SHA256，则将指定 KSS4-HMAC-SHA256 作为签名算法。经过哈希处理的负载必须以小写十六进制字符串形式表示。

如果负载为空，则使用空字符串作为哈希函数的输入。

1.2 StringToSign

要创建待签字符串（StringToSign），请如以下伪代码所示，连接算法、日期和时间、凭证范围和规范请求的摘要：

StringToSign结构

```
StringToSign =
    Algorithm + \n +
    RequestDateTime + \n +
    CredentialScope + \n +
    Hex (SHA256HASH(CanonicalRequest))
```

创建StringToSign

- 以算法名称开头，后跟换行符。该值是您用于计算规范请求摘要的哈希算法。对于 SHA256，算法是 KSS4-HMAC-SHA256。
- 追加请求日期值，后跟换行符。该日期是使用 ISO8601 基本格式以 YYYYMMDD'T'HHMMSS'Z' 格式在 x-kss-date（x-maz-date） 标头中指定的。此值必须与您在前面所有步骤中使用的值匹配。
- 追加凭证范围值，后跟换行符。此值是一个字符串，包含日期、目标区域、所请求的服务和小写字符形式的终止字符串（“kss4_request”）。区域和服务名称字符串必须采用 UTF-8 编码。

```
date.Format(<YYYYMMDD>) + "/" + <region> + "/" + <service> + "/kss4_request"
```

日期必须为 YYYYMMDD 格式。请注意，日期不包括时间值。

请确保您指定的区域是您将请求发送到的目标区域。 KS3的region 示例：如 中国北京 为 BEIJING ，其余对应关系请参阅Region英文名称。

2. SigningKey计算方法

SigningKey: 要对您的消息进行签名，需使用您的AccessKey计算您的签名密钥。为此，请使用您的AccessKey创建一系列基于哈希的消息身份验证代码 (HMAC)。此代码显示在以下伪代码中，其中 HMAC(key, data) 表示以二进制格式返回输出的 HMAC-SHA256 函数。每个哈希函数的结果将成为下一个函数的输入。

用于计算SigningKey的伪代码

```
kSecret = your Access Key
kDate = HMAC("KSS4" + kSecret, Date)
kRegion = HMAC(kDate, Region)
kService = HMAC(kRegion, Service)
kSigning = HMAC(kService, "kss4_request")
```

请注意，哈希过程中所使用的日期的格式为 YYYYMMDD（例如，20150830），不包括时间。

确保以正确的顺序为您要使用的编程语言指定 HMAC 参数。在此示例中，密钥是第一个参数，数据（消息）是第二个参数，但您使用的函数可能以不同顺序指定密钥和数据。

使用摘要（二进制格式）来派生密钥。大多数语言都有用来计算二进制格式哈希（通常称为摘要）或十六进制编码哈希（称为十六进制摘要）的函数。派生密钥需要使用二进制格式摘要。

以下示例显示了用于计算SigningKey的输入以及所生成的输出，其中 kSecret = wJalrXUtnFEMI/K7MDENG+bPxRfiCYEXAMPLEKEY。

示例输入：

```
HMAC(HMAC(HMAC(HMAC("KSS4" + kSecret,"20150830"),"BEIJING"),"ks3"),"kss4_request")
```

3. signature计算

计算公式为

```
HMAC-SHA256(SigningKey, StringToSign)
```

签名使用

通过 RESTful API 对 KS3 发起的 HTTP 签名请求，可以通过以下几种方式传递签名：

- 通过标准的 HTTP Authorization 头;
- 作为 HTTP QueryString，请注意 UrlEncode;
- 签名在表单中;

通过通过标准的 HTTP Authorization 头发送签名请求

签名头格式为

```
Authorization: algorithm Credential=access key ID/credential scope, SignedHeaders=SignedHeaders, Signature=signature
```

示例：

```
PUT /test/uploadfile HTTP/1.1
Host: test-bucket.s3-us-west-2.amazonaws.com
X-Amz-Date: 20190927T100359Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIPXSWFJRPW3RWPJ8PK/20190927/us-west-2/s3/aws4_request, SignedHeaders=amz-sdk-invocation-id;amz-sdk-retry;content-length;content-type;host;user-agent;x-amz-content-sha256;x-amz-date, Signature=6d9df5717ebb7cdeae7d08d817b80d6a90ca32320b07e46f30c64136eade9d48
amz-sdk-retry: 0/0/500
User-Agent: aws-sdk-java/1.11.311 Windows_8.1/6.3 Java_HotSpot(TM)_64-Bit_Server_VM/24.60-b09 java/1.7.0_60
x-amz-content-sha256: f68d27da17d9fd49440b0b8dac130f4f98a2d4c6aca3c9221cdbc8b264b2c8be
amz-sdk-invocation-id: e5d51221-298a-2fc0-3adf-09390a1f5beb
Content-Type: text/plain
Content-Length: 6
Connection: Keep-Alive
```

通过 URL QueryString 发送签名

示例：

```
http://ks3-cn-beijing.ksyun.com/54554/hosts.txt?X-Kss-Algorithm=KSS4-HMAC-SHA256&X-Kss-Credential=Vc7A7P4Z6wBy%2BwrOE6ff%2F20191024%2FBEIJING%2Fks3%2Fkss4_request&X-Kss-Date=20191024T065239Z&X-Kss-Expires=900&X-Kss-Signature=865fe37a5562e79aec9da8a7f252987c4c62b29d80adadbed8a699ae877e73f1&X-Kss-SignedHeaders=host
```

签名在URL QueryString中和在header中计算方法一样，只是签名信息在queryString中。

queryString列表如下

queryString	描述
X-Kss-Algorithm	签名算法:HMAC-SHA256
X-Kss-Credential	签名范围：格式为 \/\/\/\/kss4_request
X-Kss-Date	请求时间:时间格式为 yyyyMMddTHHmmssZ （X-Kss-Date<=now+15m） 请求的时间戳不能大于（服务器时间戳+15m）
X-Kss-Expires	过期时间，单位秒。时间范围为1-604800（7天）（X-Kss-Date+expire>=now）请求时间戳+过期时间 大于服务器时间戳
X-Kss-SignedHeaders	签名header
X-Kss-Signature	签名结果（计算签名时queryString不包含X-Kss-Signature）

签名在表单中

对于验证的 Post 请求，HTML 表单中必须包含 Policy 和 Signature 信息。 Policy 控制请求中哪些值是允许的。

计算 Signature 的具体流程为：

1. 创建一个 UTF-8 编码的 Policy文档，如何构建policy ，详见Policy构建方法。
2. 将 Policy 进行 base64 编码，其值即为 Policy 表单域中填入的值，将该值作为签名的字符串(StringToSign)。
3. 签名Signature =HMAC-SHA256(SigningKey, StringToSign)

表单	说明
policy	表单上传安全策略
X-Kss-Algorithm	签名算法:HMAC-SHA256
X-Kss-Credential	签名范围：格式为\/\/\/\/kss4_request
X-Kss-Date	请求时间:时间格式为 yyyyMMddTHHmmssZ （X-Kss-Date<=now+15m） 请求的时间戳不能大于（服务器时间戳+15m）
X-Kss-Signature	签名结果（计算签名时queryString不包含X-Kss-Signature）

其它说明

1. 日期

您在凭证范围中使用的日期必须与您的请求的日期匹配。您可以用多种方法将日期包括在请求中。您可以使用 date 标头或 x-kss-date（x-amz-date）标头，或者将 x-kss-date（x-amz-date）作为查询参数包含在内。

时间戳必须采用 UTC 表示，并具有以下 ISO 8601 格式：YYYYMMDD'T'HHMMSS'Z'。例如，20150830T123600Z 是有效时间戳。请勿在时间戳中包含毫秒。

先检查时间戳的x-kss-date（x-amz-date)标头或参数。如果 无法找到 x-kss-date（x-amz-date）的值，则将寻找 date 标头。 检查八位数字字符串形式的凭证范围，表示请求的年 (YYYY)、月 (MM) 和日 (DD)。例如，如果 x-amz-date 标头值为 20111015T080000Z，并且凭证范围的日期部分为 20111015，则 允许身份验证过程继续执行。

如果日期不匹配，则 拒绝请求，即使时间戳距离凭证范围中的日期仅有数秒之差也是如此。例如，将拒绝其x-kss-date（x-amz-date）标头值为 20151014T235959Z 且凭证范围包括日期 20151015 的请求。

2. URLencode方法

- 请勿对 RFC 3986 定义的任何非预留字符进行 URI 编码，这些字符包括：A-Z、a-z、0-9、连字符 (-)、下划线 (_)、句点 (.) 和波形符 (~)。
- 空格字符必须编码为 %20（不像某些编码方案那样使用“+”）
- 使用 %XY 对所有其他字符进行百分比编码，其中“X”和“Y”为十六进制字符（0-9 和大写字母 A-F），扩展 UTF-8 字符必须采用格式 %XY%ZA%BC
- 除了对象名称之外，对前斜杠字符‘/’进行编码

3. 协议头

AWS 以x-amz-开头，KS3以x-kss-开头，本协议兼容AWS，同时又支持ks3 v4签名。

-	s3	ks3
请求头前缀	x-amz-	x-kss-

queryString前缀	X-Amz-	X-Kss-
协议	AWS4	KSS4
产品	s3	ks3
定值	aws4_request	kss4_request

公共请求头

下表列出了KS3中常用的请求头部。

名称	描述
Authorization	必要的请求验证信息。匿名用户不需要。
Content-Length	请求体的长度信息（不包含头部），以字节为单位。
Content-Type	请求体中资源的类型。例如： text/plain 。
Content-MD5	请求对象的MD5摘要信息。
Date	当前请求的时间和日期。格式如: Wed, 01 Mar 2006 12:00:00 GMT。
Expect	当用户应用使用100-continue, 认证通过才会发送请求体。如果头部信息认证无效，则不会发送请求内容。 有效值: 100-continue
Host	路径样式的请求，其值为：ks3-cn-beijing.ksyun.com。 虚拟样式的请求中，样式为：BucketName.ks3-cn-beijing.ksyun.com。

公共响应头

下表列出了KS3中常用的响应头部。

名称	描述
Content-Length	响应体的字节数。 类型: String 默认值: 无
Content-Type	内容的 MIME 类型。例如：Content-Type: text/html; charset=utf-8 。 类型: String 默认值: 无
Connection	指定服务器的连接的开放和关闭。 类型: Enum 有效值: open &##124; close 默认值: 无
Date	响应时的时间和日期，格式如：Wed, 01 Mar 2014 12:00:00 GMT 。 类型: String 默认值: 无
ETag	响应的对象的实体标签，与对象内容有关，与名称无关。 类型: String
Server	响应服务器名称。 类型: String 默认值: Tengine
x-kss-request-id	由KS3指定的唯一值，可用于解决KS3出现的问题。 类型: String 默认值: 无

错误码

message	HTTP Status	描述
AccessDenied	403	拒绝访问
BadDigest	400	错误的摘要
BucketAlreadyExists	409	Bucket已经存在
BucketAlreadyOwnedByYou	409	用户已经是Bucket的拥有者
BucketNotEmpty	409	Bucket不为空
InternalError	500	内部错误
InvalidAccessKey	403	无效的AccessKey
InvalidACLString	400	ACL配置无效
InvalidAuthorizationString	400	无效的验证字符串
InvalidBucketName	400	无效的Bucket名称
InvalidDateFormat	400	无效的日期格式
InvalidDigest	400	无效的摘要
InvalidEncryptionAlgorithm	400	无效的指定加密算法
InvalidHostHeader	400	无效的头信息
InvalidParameter	400	无效的参数
InvalidPath	400	无效的路径
InvalidQueryString	400	无效的请求字符串
InvalidRange	416	无效的range
KeyTooLong	400	Key太长
MetadataTooLarge	400	metadata过大
MethodNotAllowed	405	不支持的方法
MissingDateHeader	400	头信息中缺少data
MissingHostHeader	400	头信息中缺少host
NoSuchBucket	404	该Bucket不存在
NoSuchKey	404	该Key不存在
NotImplemented	501	无法处理的方法
RequestTimeTooSkewed	403	发起请求的时间和服务器时间超出15分钟
SignatureDoesNotMatch	403	签名不匹配
TooManyBuckets	400	用户的Bucket数目超过限制
URLExpired	403	url过期
BadParams	400	参数错误
ImageTypeNotSupport	400	图片类型不支持
MissingFormArgs	400	没有上传Policy
ContentRangeError	400	Range错误
ContentLengthOutOfRange	400	上传文件内容大于range
PolicyError	400	Policy错误
ExpirationError	400	Policy中没有expiration
FormUnmatchPolicy	400	表单中的内容和policy不匹配
RequestEntityTooLarge	413	上传文件的大小不符合限制
Request Timeout	408	服务器等候请求时发生超时
Precondition Failed	412	调用GetObject接口时， object 的 ETag(entity tag)与指定值不一致

Service相关

GET Service

描述

此GET操作将返回给发出请求的验证用户一个包含其所有bucket的列表。
你需要使用KS3颁发的AccessKey来验证请求。匿名请求不会返回bucket的列表，并且你也无法得到不属于你的bucket的列表。

请求

语法

```
GET / HTTP/1.1
Host: ks3-cn-beijing.ksyun.com
Date: date
Authorization: authorization string
```

支持按照项目返回bucket列表，语法为：

```
GET  /?projectIds={projectId} HTTP/1.1
Host: ks3-cn-beijing.ksyun.com
Date: date
Authorization: authorization string
```

说明：

- 如果需要返回项目ID为2345的项目下的bucket列表，请求为/?projectIds=2345
- 如需返回多个项目下的bucket列表，例如需返回项目ID为1234、3456、3344下所有的bucket列表，多个项目ID需以英文半角逗号分隔，请求为/?projectIds=1234,3456,3344
- 如果没有projectIds参数，或者projectIds参数为空，将返回所有项目下的bucket列表

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

响应

响应内容

名称	描述
Bucket	包含bucket信息的容器 类型：Container 子节点：Name, CreationDate 父节点：ListAllMyBucketsResult.Buckets
Buckets	包含一个或多个bucket的容器 类型：Container 子节点：Bucket 父节点：ListAllMyBucketsResult
CreationDate	bucket的创建日期 类型：date(yyyy-mm-ddThh:mm:ss.timezone, e.g., 2009-02-03T16:45:09.000Z) 父节点：ListAllMyBucketsResult.Buckets.Bucket
DisplayName	Bucket拥有者的名称 类型：String 父节点：ListAllMyBucketsResult.Owner
ID	Bucket拥有者的用户ID 类型：String 父节点：ListAllMyBucketsResult.Owner
ListAllMyBucketsResult	响应信息容器 类型：Container 子节点：Owner, Buckets 父节点：无
Name	Bucket的名字 类型：String 父节点：ListAllMyBucketsResult.Buckets.Bucket
Type	Bucket的类型，NORMAL为非归档，可以上传标准存储和低频存储；ARCHIVE为归档，只能上传归档文件 类型：String 父节点：ListAllMyBucketsResult.Buckets.Bucket
Region	Bucket所在区域 类型：String 父节点：ListAllMyBucketsResult.Buckets.Bucket
Owner	包含bucket拥有者信息的容器 类型：Container 父节点：ListAllMyBucketsResult

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET / HTTP/1.1
Host: ks3-cn-beijing.ksyun.com
Date: Wed, 01 Jan  2014 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
<?xml version="1.0" encoding="UTF-8"?>
<ListAllMyBucketsResult>
  <Owner>
    <ID>bcaf1ffd86f461ca5fb16fd081034f</ID>
    <DisplayName>webfile</DisplayName>
  </Owner>
  <Buckets>
    <Bucket>
      <Name>quotes</Name>
      <CreationDate>2014-01-01T16:45:09.000Z</CreationDate>
      <Type>NORMAL</Type>
      <Region>SHANGHAI</Region>
    </Bucket>
    <Bucket>
      <Name>samples</Name>
```

```
      <CreationDate>2014-01-01T16:41:58.000Z</CreationDate>
      <Type>ARCHIVE</Type>
      <Region>BEIJING</Region>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

HEAD Bucket

描述

此 HEAD 操作用于判断某一用户空间是否存在，以及用户所拥有的操作它的权限。如果指定空间存在且用户拥有访问它的权限，KS3将返回 200 OK。否则，将会返回 404 Not Found 或 403 Forbidden 错误。

请求

语法

```
HEAD / HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

除常用响应头部外，还返回x-kss-bucket-type响应头，表示Bucket的类型，NORMAL为非归档类型，可以上传标准存储和低频存储；ARCHIVE为归档类型，只能上传归档文件。

名称	描述
x-kss-bucket-type	用于说明Bucket类型。 类型：String 默认值：NORMAL 有效值：NORMAL ARCHIVE

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
HEAD / HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: authorization string
Connection: Keep-Alive
```

响应示例

```
HTTP/1.1 200 OK
Date: Fri, 10 2012 21:34:56 GMT
Server: KS3
x-kss-bucket-type: NORMAL
```

DELETE Bucket

描述

此DELETE操作将删除URI中给定的空间。在删除空间前必须保证其中的所有对象均已删除。

请求

语法

```
DELETE / HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
DELETE / HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 204 No Content
Date: Wed, 01 Mar 2006 12:00:00 GMT
Connection: close
Server: Tengine
```

接口细节说明

- 如果Bucket非空，KS3将返回409，对应的错误码是BucketNotEmpty

GET Bucket ACL

描述

此GET操作使用 acl 子资源来返回 bucket 的 ACL(access control list)。 即用户空间的访问权限控制列表。 使用此接口，只有bucket的所有者有权限操作。

请求

语法

```
GET /?acl HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignitureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

名称	描述
AccessControlList	包含 Grant,Grantee,Permission 的容器。 类型：Container 父节点：AccessControlPolicy
AccessControlPolicy	包含了每一个 Grantee 对于某个对象的 ACL 权限设置信息。 类型：Container 父节点：无
Grant	包含被授权者和其权限信息。 类型：String 父节点：AccessControlPolicy.AccessControlList
Grantee	被授权者。 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant
DisplayName	Bucket拥有者的名称。 类型：String 父节点：AccessControlPolicy.Owner
ID	Bucket拥有者的用户ID。 类型：String 父节点：AccessControlPolicy.Owner
Owner	包含bucket拥有者信息（DisplayName,ID）的容器。 类型：Container 父节点：AccessControlPolicy
Permission	指明授予被授权者的权限信息（FULL_CONTROL,READ,WRITE）。 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET /?acl HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2006 12:00:00 GMT
Content-Length: 124
Content-Type: text/plain
Connection: close
Server: Tengine

<AccessControlPolicy>
<Owner>
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Owner>
<AccessControlList>
<Grant>
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="CanonicalUser">
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Grantee>
<Permission>FULL_CONTROL</Permission>
</Grant>
</AccessControlList>
</AccessControlPolicy>
```

接口细节说明

- 只有bucket的所有者拥有GET Bucket acl的权限

PUT Bucket

描述

此 PUT 操作将为用户创建一个新的空间。用户需要是已注册用户，并且使用有效的 Access Key ID 验证来发送请求。任何匿名请求都不会被允许创建用户空间。用户将成为其创建空间的拥有者，拥有者有最高的权限。

存储空间命名规则

我们建议所有存储空间名称都遵循 DNS 命名惯例。

> 注意 ： 如果您使用 KS3 管理控制台，则在所有区域中，存储空间名称都必须符合 DNS 标准。

符合 DNS 标准的存储空间名称使客户能够受益于新功能和操作改进，并支持对存储空间进行虚拟托管类型访问。存储空间只有一种统一的命名方法。符合 DNS 标准的存储空间名称规则如下：

- 存储空间名称的长度必须为至少 3 个字符，且不能超过 63 个字符。
- 存储空间名称必须是一系列的一个或多个标签，标签间可以用连字符（-）连接。存储空间名称可以包含小写字母、数字和连字符（-），且不能包含句点（.）。存储空间名称必须以小写字母或者数字开头和结尾。
- 存储空间名称不得采用 IP 地址格式（例如，192.168.5.4）。

以下示例是有效存储空间名称：

- my-ksbucket
- myksbucket123
- 123myksbucket

以下示例是无效存储空间名称：

- -myksbucket *存储段名称不能以连字符（-）开始。*
- myksbucket- *存储段名称不能以连字符（-）结束。*
- my.ksbucket *存储段名称不能包含句点(.)。*

当用户使用此接口创建空间时，用户可以授予其他用户或者群组关于此空间的操作权限。以下列出了通过请求头部实现的两种授权方式。

设置存储空间访问权限

- 使用 x-kss-acl 请求头部，指定一个预定义的 ACL。
- 使用 x-kss-grant-read,x-kss-grant-write,x-kss-grant-full-control 请求头部，来明确指定具体的访问权限。

设置存储空间类型

用户可以通过x-kss-bucket-type请求头指定Bucket类型。

- NORMAL表示非归档存储Bucket，若上传时，不指定Object存储类型则默认为标准存储类型
- ARCHIVE表示归档存储Bucket，若上传时，不指定Object存储类型则默认为归档存储类型

请求

语法

```
PUT / HTTP/1.1
Host: {BucketName}.{endpoint}
Content-Length: {length}
Date: {date}
Authorization: {SignatureValue}

<CreateBucketConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <LocationConstraint>{BucketRegion}</LocationConstraint>
</CreateBucketConfiguration>
```

> 注意：
>
> - endpoint与Region对应关系
> - SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口可以使用常用请求头部。获取更多信息，请点击常用请求头部

用户可以通过x-kss-bucket-type请求头指定Bucket类型，NORMAL表示非归档存储Bucket，ARCHIVE表示归档存储Bucket

名称	描述	必需
x-kss-bucket-type	用于指定bucket类型。 类型：String 默认值：NORMAL 有效值：NORMAL ARCHIVE 约束条件：无	否

用户可以使用以下header为bucket设置一个预设的ACL

名称	描述	必需
x-kss-acl	用于对象的预定义权限。 类型：String 默认值：private 有效值：private public-read public-read-write 约束条件：无	否

用户可以使用以下header为bucket设置详细的ACL

名称	描述	必须
x-kss-grant-read	为若干用户授予READ权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-write	为若干用户授予WRITE权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限。 类型：String 默认值：无 约束条件：无	否

以上header值的值为以一个逗号","分割的授权列表。每个授权信息的格式为type=value,当前type支持id：

- id:被授权者的用户id

例如，要给id为1234578和3344211的两个用户授予WRITE权限: x-kss-grant-write:id="1234578",id="3344211"

请求内容

名称	描述	必需
CreateBucketConfiguration	用户空间配置信息的容器。 类型: Container 父节点: None	否
LocationConstraint	指定用户空间将要被创建的区域。 类型: String 有效值: BEIJING SHANGHAI HONGKONG GUANGZHOU RUSSIA SINGAPORE Default: BEIJING 父节点: CreateBucketConfiguration	否

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
PUT / HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Content-Length: 0
Date: Fri, 26 Dec 2014 06:30:04 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Fri, 26 Dec 2014 06:30:04 GMT
Content-Length: 0
Connection: close
Server: Tengine
```

接口细节说明

- bucket名称是全局唯一的，如果要新建的bucket已经存在，KS3将会返回409.
- 对于大部分用户，使用x-kss-acl设置ACL即可。对于bucket，一般需要设置为私有。

PUT Bucket ACL

描述

此 PUT 操作使用 acl 资源通过访问权限列表为已经存在的用户空间设定访问权限。

当前只有Bucket的所有者拥有该权限

你可以使用下面两种方式来设置对象的权限。

- 在请求体中指定 ACL。
- 使用请求头部来设置访问权限。

  注意 ： 不能同时使用以上两种方式。

请求

语法

下面展示的是通过在请求体中指定 ACL 的方式进行设定。

```
PUT /?acl HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}

<AccessControlPolicy>
  <Owner>
    <ID>{ID}</ID>
    <DisplayName>{EmailAddress}</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
        <ID>{ID}</ID>
        <DisplayName>{EmailAddress}</DisplayName>
      </Grantee>
      <Permission>{Permission}</Permission>
    </Grant>
    ...
  </AccessControlList>
</AccessControlPolicy>
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口可以使用所有常用请求头部。获取更多信息，请点击常用请求头部。 用户可以通过以下的header为Bucket设置预设的ACL

名称	描述	必须
x-kss-acl	用于对象的预定义权限。 类型：String 默认值：private 有效值：private &##124; public-read &##124; public-read-write 约束条件：无	否

如果用户期望为Bucket设置详细的ACL，可以通过以下header设置

名称	描述	必须
x-kss-grant-read	为若干用户授予READ权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-write	为若干用户授予WRITE权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限。 类型：String 默认值：无 约束条件：无	否

以上header值的值为以一个逗号",“分割的授权列表。每个授权信息的格式为type=value,当前type支持id：

- id:被授权者的用户id

例如，要给id为1234578和3344211的两个用户授予WRITE权限: x-kss-grant-write:id="1234578",id="3344211"

请求内容

如果用户决定使用请求体来指定访问权限列表，需要使用下表元素。

注意 *如果你使用请求体设置ACL，你不能再通过请求头部设置ACL*

名称	描述
AccessControlList	包含 Grant,Grantee,Permission 的容器 类型：Container 父节点：AccessControlPolicy
AccessControlPolicy	包含了每一个 Grantee 对于某个对象的 ACL 权限设置信息 类型：Container 父节点：无
Grant	包含被授权者和其权限信息。 类型：String 父节点：AccessControlPolicy.AccessControlList
Grantee	被授权者，参考授予权限方式 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant
DisplayName	Bucket拥有者的名称 类型：String 父节点：AccessControlPolicy.Owner
ID	Bucket拥有者的用户ID，或者是被授权者的ID 类型：String 父节点：AccessControlPolicy.Owner 或者 AccessControlPolicy.AccessControlList.Grant
Owner	包含bucket拥有者信息（DisplayName,ID）的容器 类型：Container 父节点：AccessControlPolicy
Permission	指明授予被授权者的权限信息（FULL_CONTROL,READ,WRITE） 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant

授予权限方式

用户可以通过以下方式来授予某个用户对用户空间的权限。

通过用户ID,即根据用户ID授予特定用户权限

IDGranteesEmail

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
PUT /?acl HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Content-Length: 1660
Date: Fri, 26 Dec 2014 06:34:32 GMT
Authorization: authorization string

<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Owner>
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Owner>
<AccessControlList>
<Grant>
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Grantee>
<Permission>FULL_CONTROL</Permission>
</Grant>
<Grant>
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Grantee>
<Permission>READ</Permission>
</Grant>
</AccessControlList>
</AccessControlPolicy>
```

响应示例

```
HTTP/1.1 200 OK
Date: Fri, 26 Dec 2014 06:34:32 GMT
x-kss-request-id: dbea4ce4ec23415b9e454ecfa25ec4d9
Content-Length: 0
Server: Tengine
```

接口细节分析

- Bucket的权限含义详见 ACL
- 当同时在header中和Body中设置了ACL，最后只有header中的会生效。当同时在header中设置了x-kss-acl和x-kss-grant-*时，后者生效。
- 对于大部分用户，使用x-kss-acl在header中设置预设的ACL就可以满足大部分需求。对于bucket，一般需要设置x-kss-acl为private。

GET Bucket(ListObjects)

描述

此 GET 操作将指定罗列空间中的部分或全部(上限1000)的对象。
用户可以设定请求参数来选择指定空间中所要列出的对象。

用户使用此接口，需要具有对空间的 READ 权限。

> 注意：如果想要列举用户空间，可以使用 GET Service 接口。

请求

语法

```
GET / HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

> 注意：
>
> - endpoint与Region对应关系
> - SignatureValue签名算法

请求参数

参数	描述	必需
delimiter	分隔符，用于对一组参数进行分割的字符。 类型：String 默认值：无	否
encoding-type	指明请求KS3与KS3响应使用的编码方式。object key 可以包含任意Unicode字符。然而，XML 1.0解析器无法解析某些字符，如ASCII码中的0到10。对于这些不能被解析的字符可以添加到请求中，KS3会在响应中对他们进行编码。 类型：String 默认值：无 有效值：url	否
marker	指定列举指定空间中对象的起始位置。KS3按照字母排序方式返回结果，将从给定的 marker 开始返回列表。 类型：String 默认值：无	否
max-keys	设置响应体中返回的最大记录数（最后实际返回可能小于该值）。默认为1000。如果你想要的结果在1000条以后，你可以设定 marker 的值来调整起始位置。 类型：String 默认值：1000	否
prefix	限定响应结果列表使用的前缀，正如你在电脑中使用的文件夹一样。 类型：String 默认值：无	否

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，常用响应头部。

响应内容

名称	描述
Contents	每一个对象返回的元数据。 类型：XML metadata 父节点：ListBucketResult
CommonPrefixes	当用户指定分隔符后，KS3会返回他们的公共前缀。实际上，公共前缀包括的值类似于文件目录中的同一个目录下的子目录。值的数量不能超过最大数量。例如：指定分隔符为 /,对于notes/summer/a.txt 和 notes/summer/b.xml，其公共前缀为 notes/summer/。 类型：String 父节点：ListBucketResult
Delimiter	分隔符，用于分割参数。分割后便于确定公共前缀。 类型：String 父节点：ListBucketResult
DisplayName	对象的名称。 类型：String 父节点：ListBucketResult.Contents.Owner
Encoding-Type	KS3响应中对对象名称的编码方式。 类型：String 父节点：ListBucketResult
ETag	使用对象MD5摘要的实体标签。仅取决于对象的内容。 类型：String 父节点：ListBucketResult.Contents
ID	对象拥有者的用户ID。 类型：String 父节点：ListBucketResult.Contents.Owner
IsTruncated	是否被截断。如果对象列表记录数超过了设定的最大值，那么将会被截断。 类型：BooleanAncestor：ListBucketResult
Key	对象的 key。 类型：String 父节点：ListBucketResult.Contents
LastModified	最后一次被改动的时间和日期。 类型：DateAncestor：ListBucketResult.Contents
Marker	指定列举指定空间中对象的起始位置。KS3按照字母排序方式返回结果，将从给定的 marker 开始返回列表。 类型：String 父节点：ListBucketResult
MaxKeys	响应体中返回的最大记录数。默认为1000。 类型：String 父节点：ListBucketResult
Name	用户空间的名称。 类型：String 父节点：ListBucketResult
NextMarker	当用户空间中对象列表记录数超过了最大值，会标记列表被截断(IsTruncated=true)，同时返回下个记录的位置信息。用户在下次list objects的时候，可以使用该值作为marker参数。注意：当不提供delimiter参数的时候，KS3将不会返回NextMarker,如果IsTruncated为true，则可以使用返回的Contents中的最后一个key作为下次list的marker参数 类型：String 父节点：ListBucketResult
Owner	用户空间拥有者信息。 类型：String 子节点：DisplayName, ID 父：ListBucketResult.Contents | CommonPrefixes
Prefix	该list请求时指定的key前缀 类型：String 父节点：ListBucketResult
Size	对象的大小，按字节统计。 类型：String 父节点：ListBucketResult.Contents
StorageClass	存储类型，包括： STANDARD/STANDARD_IA/ARCHIVE 类型：String 父节点：ListBucketResult.Contents

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET / HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 12 Oct 2009 17:50:00 GMT
Authorization: authorization string
Content-Type: text/plain
```

响应示例

```
<?xml version="1.0" encoding="UTF-8"?>
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Name>ks3-example</Name>
    <Prefix/>
    <Marker/>
    <MaxKeys>1000</MaxKeys>
    <IsTruncated>false</IsTruncated>
    <Contents>
        <Key>my-image.jpg</Key>
        <LastModified>2009-10-12T17:50:30.000Z</LastModified>
        <ETag>fba9dede5f27731c9771645a39863328</ETag>
        <Size>434234</Size>
        <StorageClass>STANDARD</StorageClass>
        <Owner>
            <ID>73410125</ID>
            <DisplayName>ks3@kingsoft.com</DisplayName>
        </Owner>
    </Contents>
    <Contents>
       <Key>my-third-image.jpg</Key>
         <LastModified>2009-10-12T17:50:30.000Z</LastModified>
        <ETag>1b2cf535f27731c974343645a3985328</ETag>
        <Size>64994</Size>
        <StorageClass>STANDARD</StorageClass>
        <Owner>
            <ID>73410125</ID>
            <DisplayName>ks3@kingsoft.com</DisplayName>
        </Owner>
    </Contents>
</ListBucketResult>
```

接口细节分析

- 由于max-keys最大只能是1000，所以一次list不一定可以把所有文件都罗列出来。当返回的IsTruncated为true时，表示返回的结果被截断，也就是说这次list还没有把所有的object都list出来，

若请求参数中包含delimiter，则可以使用返回的NextMarker做为下次list的marker参数，若请求参数中不包含delimiter,则可以使用返回的Contents中的最后一个做为下次list的marker参数。

- 通过使用prefix、delimiter可以模拟目录结构。如果设定了max-keys和delimiter，返回的文件列表数量不一定等于max-keys，有可能会小于max-keys。假设bucket下有以下几个object，分别是：
  movie/action/1.mp4
  movie/fun/2.mp4 movie/fun/3.mp4
  photo/1.jpg
  4.txt
  当只提供prefix=movie/fun/时，返回的结果为movie/fun/2.mp4、movie/fun/3.mp4
  当提供delimiter=/时，返回的结果为CommonPrefixes:movie/、photo/ Contents:4.txt
  当提供prefix=movie/,delimiter=/时，返回的结果为CommonPrefixes:movie/action/、movie/fun/

对不可见字符处理的说明

1. 由于部分不可见字符转换xml时有异常，ks3对接口的不可见字符进行了转码
2. 转码的规则

对不可见字符的十六进制数值进行转换，转换为#x{dd};如，对不可见字符0x00（空字符）转换为#x00;

具体转换列表如下：

原不可见字符(十六进制)	转换后字符
0x00	#x00;
0x01	#x01;
0x02	#x02;
0x03	#x03;
0x04	#x04;
0x05	#x05;
0x06	#x06;
0x07	#x07;
0x08	#x08;
0x0b	#x0b;
0x0c	#x0c;
0x0e	#x0e;
0x0f	#x0f;
0x10	#x10;
0x11	#x11;
0x12	#x12;
0x13	#x13;
0x14	#x14;
0x15	#x15;
0x16	#x16;
0x17	#x17;
0x18	#x18;
0x19	#x19;
0x1a	#x1a;
0x1b	#x1b;
0x1c	#x1c;
0x1d	#x1d;
0x1e	#x1e;
0x1f	#x1f;
0xfffe	#xfffe;
0xffff	#xffff;

3. 转码示例

```
import java.util.ArrayList;
import java.util.List;

public class DecodeInvalidStr {

public static void main(String[] args) {
    String str = decodeInvalidStr("test#x1f;char#x1e;hello#xfffe;transfer");
    System.out.println(str);

}

 public static String decodeInvalidStr(String str) {
     if(str == null) {
         return null;
     }
       List<Character> newChar = new ArrayList<Character>();
      char[] array = str.toCharArray();
      int skipIndex = -1;
      for (int i = 0, length = array.length; i < length; i++) {
        if(i <= skipIndex){
             continue;
        }
          if(array[i] == (char)'#'
                    && (i+1) < array.length
                    && array[i+1] == (char)'x'){
               StringBuffer strChar = new StringBuffer();
               for(int j = i+2;j<i+7;j++){
                     if(array[j] == (char)';'){
                           skipIndex = j;
                           char value =(char) Integer.parseInt(strChar.toString(), 16);
                           newChar.add(value);
                           break;
                     }else{
                           strChar.append(array[j]);
                     }
               }
          }
          else{
               newChar.add(array[i]);
          }

     }
     return toString(newChar);
 }

private static String toString(List newChar){ char[] charArray = new char[newChar.size()]; int i = 0; for(Character c : newChar){ charArray[i++] = c; } return new
String(charArray); }

}
```

GET Bucket logging

描述

此 GET 操作使用 logging 资源来返回用户空间的日志记录状态。

若使用此接口需要用户是此空间的拥有者。

请求

语法

```
GET /?logging HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

名称	描述
BucketLoggingStatus	响应信息的容器。 类型：Container 父节点：无
LoggingEnabled	日志信息的容器。启用日志时，该容器及其子节点会出现，否则，消失。 类型：Container 父节点：BucketLoggingStatus
TargetBucket	指定需要返回日志信息的用户空间。 类型：String 父节点：BucketLoggingStatus.LoggingEnabled
TargetGrants	授权信息的容器。(暂不支持该功能) 类型：Container 父节点：BucketLoggingStatus.LoggingEnabled
TargetPrefix	指定日志文件被存放的键值(逻辑分层+文件名)的前缀。 类型：String 父节点：BucketLoggingStatus.LoggingEnabled

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET /?logging HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 25 Nov 2009 12:00:00 GMT
Authorization: authorization string
```

启用日志响应示例

```
HTTP/1.1 200 OK
Date: Wed, 25 Nov 2009 12:00:00 GMT
Connection: close
Server: Tengine

<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>ks3-example</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>
    <TargetGrants>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

不启用日志响应示例

```
HTTP/1.1 200 OK
Date: Wed, 25 Nov 2009 12:00:00 GMT
Connection: close
Server: KS3

<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01" />
```

PUT Bucket logging

描述

此 PUT 操作使用 logging 资源来设定用户空间的日志参数和指定允许查看日志和更改日志参数的权限的用户。

若使用此接口需要用户是此空间的拥有者。

空间拥有者对于所有日志默认被授予 FULL_CONTROL 权限。用户可以通过 Grantee 请求参数为其他用户授权。请求参数 Permissions 可以设定权限的具体内容。(暂不支持该类型授权)

要启用日志，用户需要设定 loggingEnabled 及其子标签。

要关闭日志，用户需要使用空的 BucketLoggingStatus 请求参数。

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01" />
```

请求

语法

```
PUT /?logging HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}

Request elements vary depending on what you're setting.
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

名称	描述
BucketLoggingStatus	响应信息的容器。 类型：Container 父节点：无
EmailAddress	拥有查看日志信息的用户的电子邮件。 类型：String 父节点：BucketLoggingStatus.LoggingEnabled.TargetGrants.Grant.Grantee
Grant	被授权者及其权限的信息的容器。(暂不支持) 类型：Container 父节点：BucketLoggingStatus.LoggingEnabled.TargetGrants
Grantee	拥有查看权限的用户信息的容器。(暂不支持) 类型：Container 父节点：BucketLoggingStatus.LoggingEnabled.TargetGrants.Grant
LoggingEnabled	日志信息的容器。启用日志时，该容器及其子节点会出现，否则，消失。 类型：Container 父节点：BucketLoggingStatus
Permission	被授权者拥有的对用户空间的日志的权限。(暂不支持)； 父节点：BucketLoggingStatus.LoggingEnabled.TargetGrants.Grant
TargetBucket	指定需要返回日志信息的用户空间。 类型：String 父节点：BucketLoggingStatus.LoggingEnabled
TargetGrants	授权信息的容器。(暂不支持) 类型：Container 父节点：BucketLoggingStatus.LoggingEnabled
TargetPrefix	指定日志文件被存放的键值(逻辑分层+文件名)的前缀。 类型：String 父节点：BucketLoggingStatus.LoggingEnabled

授予权限方式

用户可以通过以下方式来授予某个用户对日志的权限。

通过用户ID

```
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser"><ID>{ID}</ID><DisplayName>{GranteesEmail}</DisplayName>
</Grantee>
```

通过URI

```
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group"><URI>http://acs.ksyun.com/groups/global/AllUsers</URI></Grantee>
```

注意：当前仅支持http://acs.ksyun.com/groups/global/AllUsers，表示所有用户，包括匿名用户。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
PUT /?logging HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Content-Length: 214
Date: Fri, 26 Dec 2014 06:38:43 GMT
Authorization: authorization string

<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>ks3-example</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>
    <TargetGrants>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

响应示例

```
HTTP/1.1 200 OK
Date: Fri, 26 Dec 2014 06:38:43 GMT
Server: Tengine
```

关闭日志请求示例

```
PUT /?logging HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Content-Length: 214
Date: Fri, 26 Dec 2014 06:38:43 GMT
```

```
Authorization: authorization string
```

```
<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01" />
```

关闭日志响应示例

```
HTTP/1.1 200 OK
Date: Fri, 26 Dec 2014 06:38:43 GMT
Server: Tengine
```

接口细节分析

- 该接口可以开启bucket的日志功能，KS3将会把bucket的每天的请求日志上传到用户指定的bucket下。
- 源bucket和目标bucket必须是属于同一个用户同一个region的两个或者一个bucket。建议使用两个不同的bucket，目标bucket专门用来存日志。
- 只有bucket的所有者才有权限使用该接口。

GET Location

描述

此 GET 操作将使用 location 资源来返回用户空间的区域。 用户在 PUT Bucket 请求中使用 LocationConstraint 来设置用户空间的区域。更多信息，请查看PUT Bucket。

用户使用此接口需要用户是此空间的拥有者。

请求

语法

```
GET /?location HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: authorization string
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

Name	Description
LocationConstraint	指定用户空间被安置的区域。. 类型: String 有效值: [BEIJING ; SHANGHAI]

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET /?location HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2010 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
<?xml version="1.0" encoding="UTF-8"?>
<LocationConstraint xmlns="http://s3.amazonaws.com/doc/2006-03-01/">BEIJING</LocationConstraint>
```

List Multipart Uploads

描述

此 List 操作将会列出所有正在进行的分块上传任务。正在进行的分块上传任务是指那些已经启动，却没有放弃或完成的分块上传任务。

在一次响应中，此操作最多返回1000（默认值）个分块上传任务。用户可以使用 max-uploads 参数来限定最大值。如果用户空间中正在进行的分块上传任务数大于此操作设定的最大值，响应中将会将 IsTruncated 元素设为 true。用户可以使用 key-marker 和 upload-id-marker 参数来列出未列出的任务。

请求

语法

```
GET /?uploads HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

参数	描述	必需

delimiter	分隔符，用于对一组参数进行分割的字符。 类型：String 默认值：无	否
encoding-type	指明请求KS3与KS3响应使用的编码方式。object key 可以包含任意Unicode字符。然而，XML 1.0解析器无法解析某些字符，如ASCII码中的0到10。对于这些不能被解析的字符可以添加到请求中，KS3会在响应中对他们进行编码。 类型：String 默认值：无 有效值：url	否
max-uploads	限定要列出正在进行任务的最大值([1,1000])。响应中将会在响应体中返回其值。 类型：Integer 默认值：1000	否
upload-id-marker	与key-marker一起使用，指定列举指定空间中正在进行分块上传任务的起始位置。如果 key-marker 没有指定，此参数将被忽略。否则将只会列出 upload ID 比设定的大的任务。 类型：String 默认值：无	否
key-marker	与 upload-id-marker 一起使用， 指定列举指定空间中正在进行分块上传任务的起始位置。如果 upload-id-marker 没有指定，则只会列出 key 按照词典顺序比给定的 key-marker 大的任务。如果 upload-id-marker 被指定了，那么将列出 key 等于给定的 key-marker 或被包含且上传ID大于指定的 upload-id-marker 的任务。 类型：String 默认值：1000	否
prefix	限定响应结果列表使用的前缀，正如你在电脑中使用的文件夹一样。 类型：String 默认值：无	否

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

除常用响应头部外，还返回storage-class响应头，表示文件的类型。

名称	描述
storage-class	用于说明Bucket文件类型。 标准存储不返回此响应头；低频存储此响应头内容为STANDARD_IA ；归档存储此响应头内容为ARCHIVE

响应内容

名称	描述
ListMultipartUploadsResult	响应内容的容器。 类型：Container 子节点:Bucket, KeyMarker, UploadIdMarker, NextKeyMarker, NextUploadIdMarker, MaxUploads, Delimiter, Prefix, CommonPrefixes, IsTruncated 父节点：无
Bucket	启动分块上传任务的用户空间名称。 类型：String 父节点：ListMultipartUploadsResult
KeyMarker	列表开始位置的 Key 。 类型：String 父节点：ListMultipartUploadsResult
UploadIdMarker	列表开始位置的 upload ID。 类型：String 父节点：ListMultipartUploadsResult
NextKeyMarker	在一个连续列表请求中，如果列表是被截断的，应该通过设定 key-marker 值来返回下次列表开始位置。 类型：String 父节点：ListMultipartUploadsResult
NextUploadIdMarker	在一个连续列表请求中，如果列表是被截断的，应该通过设定 upload-id-marker 值来返回下次列表开始位置 类型：String 父节点：ListMultipartUploadsResult
Encoding-Type	KS3响应中对对象名称的编码方式。 类型：String 父节点：ListBucketResult
MaxUploads	响应中列表应包含的最大条目数。 类型：Integer 父节点：ListMultipartUploadsResult
IsTruncated	是否被截断。如果对象列表记录数超过了设定的最大值，那么将会被截断。 类型：BooleanAncestor：ListMultipartUploadsResult
Upload	包含某个特定分块上传任务信息的容器。响应中应包含0个或多个 Upload 元素。 类型：Container 子节点：Key, UploadId, InitiatorOwner, StorageClass, Initiated 父节点：ListMultipartUploadsResult
Key	分块上传任务上传对象的 key。 类型：Integer 父节点：Upload
UploadID	分块上传任务的ID。 类型：Integer 父节点：Upload
Initiator	包含分块上传任务发起人信息的容器。 类型：Container 子节点：ID, DisplayName 父节点：Upload
ID	用户ID。 类型：String 父节点：Initiator,Owner
DisplayName	发起人或拥有者的名称。 类型：String 父节点：Initiator,Owner
Owner	用户空间拥有者信息。 类型：String 子节点：DisplayName, ID 父节点：Upload
StorageClass	上传对象的存储方式。 类型：String 说明：标准存储返回STANDARD；低频存储此响应内容值为STANDARD_IA ；归档存储此响应内容值为ARCHIVE 类型：String 父节点：Upload

Initiated	分块上传任务启动时的时间和日期。 类型：Date 父节点：Upload
ListMultipartUploadsResult.Prefix	对象 key 中指定的前缀。 类型：String 父节点：ListMultipartUploadsResult
Delimiter	分隔符，用于分割参数。分割后便于确定公共前缀。 类型：String 父节点：ListMultipartUploadsResult
CommonPrefixes	当用户指定分隔符后，KS3会返回他们的公共前缀。实际上，公共前缀包括的值类似于文件目录中的同一个目录下的子目录。值的数量不能超过最大数量。例如：指定分隔符为 /,对于notes/summer/a.txt 和 notes/summer/b.xml，其公共前缀为 notes/summer/。 类型：String 父节点：ListMultipartUploadsResult
CommonPrefixes.Prefix	如果设定了 Prefix 参数，则此参数的值将为Prefix后开始到第一个分隔符止，否则从头开始到第一个分隔符止。 类型：String 父节点:CommonPrefixes

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET /?uploads&max-uploads=3 HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Mon, 1 Nov 2010 20:34:56 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-kss-request-id: 656c76696e6727732072657175657374
Date: Mon, 1 Nov 2014 20:34:56 GMT
Content-Length: 1330
Connection: keep-alive
Server: KS3

<?xml version="1.0" encoding="UTF-8"?>
<ListMultipartUploadsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Bucket>ks3-example</Bucket>
  <KeyMarker></KeyMarker>
  <UploadIdMarker></UploadIdMarker>
  <NextKeyMarker>my-movie.m2ts</NextKeyMarker>
  <NextUploadIdMarker>YW55IGlkZWEgd2h5IGVsdmluZydzIHVwbG9hZCBmYW1sZWQ</NextUploadIdMarker>
  <MaxUploads>3</MaxUploads>
  <IsTruncated>true</IsTruncated>
  <Upload>
    <Key>my-divisor</Key>
    <UploadId>f9957b016aaf37c7569c91fd14501847</UploadId>
    <Initiator>
      <ID>KS3UserId</ID>
      <DisplayName>Ks3User</DisplayName>
    </Initiator>
    <Owner>
      <ID>KS3UserId</ID>
      <DisplayName>Ks3User</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <Initiated>2014-11-10T20:48:33.000Z</Initiated>
  </Upload>
  <Upload>
    <Key>my-movie.m2ts</Key>
    <UploadId>f9957b016aaf37c7569c91fd14501847</UploadId>
    <Initiator>
      <ID>73404060</ID>
      <DisplayName>ks3_dt@kingsoft.com</DisplayName>
    </Initiator>
    <Owner>
      <ID>73410125</ID>
      <DisplayName>ks3@kingsoft.com</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <Initiated>2014-11-10T20:48:33.000Z</Initiated>
  </Upload>
  <Upload>
    <Key>my-movie.m2ts</Key>
    <UploadId>f9957b016aaf37c7569c91fd14501847</UploadId>
    <Initiator>
      <ID>73404060</ID>
      <DisplayName>ks3_dt@kingsoft.com</DisplayName>
    </Initiator>
    <Owner>
      <ID>73410125</ID>
      <DisplayName>ks3@kingsoft.com</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <Initiated>2014-11-10T20:49:33.000Z</Initiated>
  </Upload>
</ListMultipartUploadsResult>
```

接口细节分析

- 通过该接口可以把bucket下正在进行的分块上传罗列出来。对于很久之前初始化，且无人再使用的分块上传，建议调用Abort Multipart Upload接口删除。
- 通过该接口可以把bucket下正在进行的分块上传罗列出来。对于很久之前初始化，且无人再使用的分块上传，建议调用Abort Multipart Upload接口删除。

Put Bucket Policy

描述

此PUT接口可以添加一个 Bucket Policy 到某个 bucket。 如果某个接口之前已经有Bucket Policy，新添加的Bucket Policy将完全替换旧的。 如果要使用此接口，你需要是这个Bucket的所有者。 若接口调用成功，则返回204，如调用失败，则会返回错误码及具体的错误信息。

> 注意：Bucket Policy 规则总大小不得超过20KB

请求

语法

```
PUT /?policy HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
Policy written in JSON
```

> 注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

一段用于描述bucket policy的Json字符串。

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
PUT /?policy HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Content-Length: 225
Authorization: authorization string
Content-Md5: 8evRehlmPHkf+VuSe8k6Rg==
Date: Tue, 19 Jul 2016 09:12:54 GMT
{"Version":"2008-10-17","Statement": [{"Sid":"1","Effect":"Allow","Principal":{"KSC":["krn:ksc:iam::{userid}:root" ,krn:ksc:iam::(acountid):user/(userName)]},"Action":["ks3:*"],"Resource":["krn:ksc:ks3:::ks3-example","krn:ksc:ks3:::ks3-example/*"],"Condition":{"IpAddress": {"ksc:SourceIp": "54.240.143.1"}}}]}
```

响应示例

```
HTTP/1.1 204 No Content
Content-Length: 0
Connection: keep-alive
Date: Tue, 19 Jul 2016 09:14:23 GMT
Server: Tengine
X-Application-Context: application
X-Kss-Request-Id: 54a47bda18ac4e6e91de369add54218e
```

Get Bucket Policy

描述

此GET接口可以获取某个Bucket的BucketPolicy配置 如果要使用此接口，你需要是这个Bucket的所有者。 若接口调用成功，则返回200以及BucketPolicy配置信息；若无权限获取BucketPolicy，则返回403 Access Denied；若不存在BucketPolicy配置，则返回404 Policy Not Found

请求

语法

```
GET /?policy HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部

请求内容

一段用于描述bucket policy的Json字符串。

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

BucketPolicy配置信息，Json格式的字符串。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET http://ks3-example.ks3-cn-beijing.ksyun.com/?policy HTTP/1.1
Host: ks3-cn-beijing.ksyun.com
Authorization: authorization string
Date: Tue, 19 Jul 2016 09:12:54 GMT
```

响应示例

Bucket Policy存在时：

```
HTTP/1.1 200 OK
Content-Length: 232
Content-Type: application/json; charset=UTF-8
Date: Tue, 19 Jul 2016 09:25:29 GMT
```

```
Server: Tengine
X-Kss-Request-Id: 708e01a0b42642cd94611f33a2a96874
{"Version":"2008-10-17","Statement":[{"Sid":"1","Effect":"Allow","Principal":{"KSC":["krn:ksc:iam::{userid}:root"
]},"Action":["ks3:PutObject"],"Resource":["krn:ksc:ks3:::ks3-example/*","krn:ksc:ks3:::ks3-example"]}]}
```

Bucket Policy不存在时:

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Date: Tue, 19 Jul 2016 09:28:59 GMT
Server: Tengine
X-Kss-Request-Id: 1f807bb266854e4487ce27857001ba38
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><Error><Code>NoSuchBucketPolicy</Code>
<Message>The bucket policy does not exist.</Message><Resource>/ks3-example/?policy</Resource>
<RequestId>1f807bb266854e4487ce27857001ba38</RequestId></Error>
```

Delete Bucket Policy

描述

此Delete接口可以删除某个Bucket的BucketPolicy配置 如果要使用此接口，你需要是这个Bucket的所有者。 该接口调用成功后将返回204

请求

语法

```
DELETE /?policy HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口只使用常用请求头部。获取更多信息，请点击常用请求头部

请求内容

一段用于描述bucket policy的Json字符串。

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

BucketPolicy配置信息，Json格式的字符串。

特殊错误

该接口不返回任何特殊错误。

示例

```
DELETE http://ks3-example.ks3-cn-beijing.ksyun.com/?policy HTTP/1.1
Host: ks3-cn-beijing.ksyun.com
Authorization: authorization string
Date: Tue, 19 Jul 2016 09:12:54 GMT
```

响应

```
HTTP/1.1 204 No Content
Date: Tue, 19 Jul 2016 09:28:34 GMT
Server: Tengine
X-Kss-Request-Id: a5398b800e3a4d498bb4a7e017ee9259
Content-Length: 0
```

Put Bucket Lifecycle

描述

此PUT接口会设置一个Bucket的lifecyle规则。如果某个Bucket之前已经有lifecycle规则，新添加的规则将完全覆盖旧的，请注意这一点以免误删。

提示：为避免出错，推荐用户在控制台进行生命周期规则的设置，更加简单高效。

用户设置规则后，ks3内部会按照用户的设置，自动的将匹配到的object删除或者转化存储类型。

如果要使用此接口，您需要是这个Bucket的owner或者拥有设置生命周期管理规则的权限，即ks3:PutBucketLifecycle。

注意：当在生命周期规则中指定对象标签时，无论是一个还是多个均需要And节点。

请求

语法

```
PUT /?lifecycle HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
Content-length: {length}
Content-MD5: {md5}

Lifecycle configuration in the request body
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

除公共头部外，本接口必须有下面一项

名称	描述	必须
Content-MD5	body里data的128位md5 digest，再用base64编码。这个header必须存在，以便检查body是否损坏。详见RFC-1864 类型：String 默认值：无	是

请求Body

一段描述lifecyle configuration的xml。

```
<LifecycleConfiguration>
  <Rule>
    <ID>id1</ID>
    <Filter>
      <Prefix>documents</Prefix>
    </Filter>
    <Expiration>
      <Date>2016-12-31T00:00:00.000Z</Date>
    </Expiration>
    <Status>Enabled</Status>
  </Rule>
  <Rule>
    <ID>id2</ID>
    <Filter>
      <Prefix>logs</Prefix>
    </Filter>
    <Expiration>
      <Days>130</Days>
    </Expiration>
    <Transition>
      <Days>10</Days>
      <StorageClass>STANDARD_IA</StorageClass>
    </Transition>
    <Transition>
      <Days>40</Days>
      <StorageClass>ARCHIVE</StorageClass>
    </Transition>
    <Status>Enabled</Status>
  </Rule>
  <Rule>
    <ID>id3</ID>
    <Filter>
      <Prefix>pic</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Date>2018-01-01T00:00:00.000Z</Date>
    </Expiration>
  </Rule>
  <Rule>
    <ID>id4</ID>
        <Filter>
         <And>
            <Prefix>123</Prefix>
            <Tag>
               <Key>age</Key>
               <Value>21</Value>
            </Tag>
            <Tag>
              <Key>name</Key>
              <Value>li</Value>
             </Tag>
         </And>
      </Filter>
  <Status>Enabled</Status>
  <Expiration>
    <Date>2021-01-01T00:00:00.000Z</Date>
  </Expiration>
        </Rule>

</LifecycleConfiguration>
```

Rule就是规则，主要包含：

- Filter：规定匹配规则，支持分别设置筛选条件为前缀Prefix或标签Tag，也可组合使用。设置文件前缀时，单条规则仅容许设置一个。如规定了log的前缀的文件的删除操作，那object key是log1,log2,log/2016开头的都会被删，Tag为对象标签，单条规则可设置最多10个标签。
- Status：指定这条规则是启用还是禁用。
- Expiration：object过期的描述，哪个时间之前过期，或存在多少天就过期。
- Transition：object转换存储类型的描述。

注意：

- 规则的执行在每天0点。
- 一个Bukcet rule总条数 <= 1000。
- Rule的ID在一个bucket内必须唯一，不同bucket可以相同。ID可以是任意字符串，包括中文，但不能超过255字符(utf8编码)。
- 一个bucket里可以有多条rule，仅设置前缀匹配时，每条rule可以设置不重叠的的prefix，如logs和docs，如logs和logs2016就是重叠的，导致冲突不可设置。规则中同时指定Tag 与prefix同时设置时可支持重叠前缀 。
- Days指定相对时间，是相对object的last modfiy time，如object在2017-01-02 15:05被modify，Days是2，删除发生在2017-01-05 00:00，即顺延2天再找到下一个0点。 注意：这里说的object last modify time，是上一次PUT，POST，COPY的时间。
- Date指定绝对时间，必须是2017-01-01T00:00:00+08:00 这样ISO 8601格式，北京时间，时分秒必须写成0点，传非0点会报错。该规则会在Date指定的那天0点执行，把last modify time<=该Date的object删除；如果Date是过去时间，该规则在当天深夜就会执行，执行时判断last modify time <= 该Date；如果是未来时间，该规则在到达那个Date才会被执行，执行时仍然判断last modify time <= 该Date。
- StorageClass 指文件要转换哪种存储类型，取值包括：STANDARD_IA，ARCHIVE。
- **在您设置完对象标签相关的生命周期规则之后，请在控制台或使用Get Bucket Lifecycle接口再次检查，确认设置结果是否与您的预期相符。**
  xml中的节点具体如下：

名称	描述	是否必须
LifecycleConfiguration	包含一堆Rule的容器，一个Bucket最多1000条Rule。 类型：Container 子节点：Rule 父节点：无	是
Rule	包含一条规则 类型：Container 父节点：LifecycleConfiguration	是
ID	Rule的唯一标识，一个Bucket内ID不能重复。ID长度<=255字符，注意是utf8编码字符，不是字节 类型：String 父节点：Rule	是
Filter	规定前缀，一个Rule只能有一个Filter，不同rule的prefix不能冲突。 类型：Container 子节点：Prefix 父节点：Rule	否
And	对象筛选器中的一个子集，当指定tag时需要此元素， 包括同时指定 Prefix 和 Tag 筛选，以及指定一个或多个 Tag 筛选。 类型：Container 父节点：LifecycleConfiguration.Rule.Filter	否

Prefix	符合这个前缀的object才会被删。一个Rule只能有一个Filter和一个Prefix 类型：String 父节点：Filter	否
Tag	标签集合，最多支持10个标签 类型：Container	否
Key	标签的 Key，长度不超过128字节，支持大小写字母、数字、空格和符号 + - = . _ : / 类型：String	否
Value	标签的 Value，长度不超过256字节，支持大小写字母、数字、空格和符号 + - = . _ : / 类型：String	否
Status	Enabled状态，该Rule就定期被执行；Disabled，该Rule被忽略，但该Rule随时能被Enable而不是被删。 类型：String 父节点：Rule 取值：Enabled, Disabled	是
Expiration	规定对应的object何时被删。 类型：Container 子节点：Days，Date 父节点：Rule	是
Date	last modify day<这个数的object被删。必须是ISO 8601格式的北京时间。时分秒必须填0，即必须是0点。 类型：String 父节点：Expiration，Transition	是，如果没有Days
Days	规定一个正数，对应object在last modify多少天之后被删。 类型：整数 父节点：Expiration，Transition	是，如果没有Date
Transition	指定Object在有效生命周期中，何时将对象转储为IA或者Archive存储类型 类型：Container 子节点：Days，Date，StorageClass	否
StorageClass	指定对象转储到目标存储类型。 父节点：Transition 取值：STANDARD_IA,ARCHIVE（控制台使用aws sdk，不支持ARCHIVE）	否，如果Transition有的话，则必须包含

其中，

- ID必须有。不同Rule的ID不能重复。
- Date和Days只能二选一，并且不能不选，必须填写其中一个。
- Status节点一定要有。
- Expiration一定要有。
- Filter和Prefix可以没有。但如果有prefix，不同rule之间不能冲突，如log和log2016就是冲突的。
- Prefix与Tag可搭配使用，单条规则仅能有一个Prefix，Tag可以设置多个。
- Rule的子节点里，应该出现的，都只能出现一次。如ID，只能有一个；Status，只能有一个。

响应头部

本接口只带有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

不返回内容。

特殊错误

不返回任何特殊错误。

示例

简单请求示例

```
PUT /?lifecycle HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 14 May 2014 02:11:21 GMT
Content-MD5: xxx
Authorization: authorization string
Content-Length: yyy
Content-type: application/xml

<LifecycleConfiguration>
  <Rule>
    <ID>id1</ID>
    <Filter>
      <And>
          <Tag>
             <Key>age</Key>
             <Value>21</Value>
          </Tag>
      </And>
    </Filter>
    <Expiration>
      <Date>2016-12-31T00:00:00+08:00</Date>
    </Expiration>
    <Status>Enabled</Status>
  </Rule>
</LifecycleConfiguration>
```

复杂请求示例

```
PUT /?lifecycle HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 14 May 2014 02:11:21 GMT
Content-MD5: xxx
Authorization: authorization string
Content-Length: yyy
Content-type: application/xml

<LifecycleConfiguration>
  <Rule>
    <ID>id1</ID>
    <Filter>
      <Prefix>documents</Prefix>
    </Filter>
    <Expiration>
      <Date>2016-12-31T00:00:00+08:00</Date>
    </Expiration>
    <Status>Enabled</Status>
  </Rule>
 <Rule>
    <ID>id2</ID>
    <Filter>
      <Prefix>logs</Prefix>
    </Filter>
    <Expiration>
      <Days>130</Days>
    </Expiration>
    <Transition>
      <Days>10</Days>
      <StorageClass>STANDARD_IA</StorageClass>
    </Transition>
    <Transition>
      <Days>40</Days>
```

```
      <StorageClass>ARCHIVE</StorageClass>
    </Transition>
    <Status>Enabled</Status>
  </Rule>
  <Rule>
    <ID>id3</ID>
     <Filter>
       <And>
          <Prefix>docs</Prefix>
          <Tag>
             <Key>age</Key>
             <Value>21</Value>
          </Tag>
          <Tag>
            <Key>name</Key>
            <Value>li</Value>
           </Tag>
       </And>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Date>2021-01-01T00:00:00.000Z</Date>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

以上在某个bucket里设3条规则。

- 第一条删除2016-12-31 0点前的以documents开头的object key。
- 第二条让logs开头的在最终modify后3天删除。
- 第三条是未来时间，在2021.1.1才会执行，执行效果是删除lastmodify < 2021-01-01 且符合以docs开头，具有age=21，name=li标签的文件。

响应示例

```
HTTP/1.1 200 OK
Content-Length: 0
Date: Tue, 19 Jul 2017 09:14:23 GMT
Server: Tengine
x-kss-request-id: 54a47bda18ac4e6e91de369add54218***
```

说明： 更多信息请参见管理文件生命周期。

Get Bucket Lifecycle

描述

此GET操作返回bucket的lifecyle配置，即描述各条Rule的一个xml。 使用此接口，您需要是bucket的所有者或者具有获取生命周期管理规则的权限，即ks3:GetBucketLifecycle。

请求

语法

```
GET /?lifecycle HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

本接口不使用请求参数。

请求头部

只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

不使用请求内容。 KS3在策略中指定的一组权限。每一个权限，都会映射到特定 KS3 操作。

响应

响应头部

只带有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

返回一个xml，可能有下列节点：

名称	描述	是否返回
LifecycleConfiguration	包含一堆Rule的容器，一个Bucket最多1000条Rule。 类型：Container 子节点：Rule 父节点：无	是
Rule	包含一条规则 类型：Container 父节点：LifecycleConfiguration	是
ID	Rule的唯一标识，一个Bucket内ID不能重复。ID长度<=255字符，注意是utf8编码字符，不是字节。 类型：String 父节点：Rule	是
Filter	规定前缀，一个Rule只能有一个Filter，不同rule的prefix不能冲突。 类型：Container 子节点：Prefix 父节点：Rule	否
And	对象筛选器中的一个子集，仅当需要指定多种筛选规则时才需要此元素。 例如：同时指定 Prefix 和 Tag 筛选，或同时指定多个 Tag 筛选。 类型：Container 父节点：LifecycleConfiguration.Rule.Filter	否
Prefix	符合这个前缀的object才会被删。一个Rule只能有一个Filter和一个Prefix。 类型：String 父节点：Filter	否
Tag	标签集合，最多支持10个标签 类型：Container	否

Key	标签的 Key，长度不超过128字节，支持英文字母、数字、空格、加号、减号、下划线、等号、点号、冒号、正斜线、反斜线。 类型：String	否
Value	标签的 Value，长度不超过256字节，支持英文字母、数字、空格、加号、减号、下划线、等号、点号、冒号、正斜线、反斜线。 类型：String	否
Status	Enabled状态，该Rule就定期被执行；Disabled，该Rule被忽略，但该Rule随时能被Enable而不是被删。 类型：String 父节点：Rule 取值：Enabled，Disabled	是
Expiration	规定对应的object何时被删。 类型：Container 子节点：Days，Date 父节点：Rule	否
Days	规定一个正数，对应object在last modify多少天之后被删。 类型：整数 父节点：Expiration，Transition	是，如果没有Days
Date	last modify day<这个数的object被删。必须是ISO 8601格式的北京时间。时分秒必须填0，即必须是0点。 类型：String 父节点：Expiration，Transition	是，如果没有Date
Transition	指定Object在有效生命周期中，何时将对象转储为IA或者Archive存储类型。 类型：Container 子节点：Days，Date，StorageClassg	否
StorageClass	指定对象转储到目标存储类型。 父节点：Transition 取值：STANDARD_IA,ARCHIVE（控制台使用aws sdk，不支持ARCHIVE）	否，如果Transition有的话，则必须包含。

示例

请求示例

```
GET /?lifecycle HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2016 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 312
Date: Wed, 16 Aug 2017 12:23:54 GMT
Server: tencent-cos
x-kss-request-id: NTk5NDM5NWFfMjQ4OGY3Xzc3NGRf****

<LifecycleConfiguration>
  <Rule>
    <ID>id1</ID>
    <Filter>
       <Prefix>documents/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>100</Days>
      <StorageClass>STANDARD_IA</StorageClass>
    </Transition>
  </Rule>
  <Rule>
    <ID>id2</ID>
    <Filter>
       <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>10</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

Delete Bucket Lifecycle

描述

此Delete接口可以删除某个Bucket的lifecycle配置。如果要使用此接口，你需要是这个Bucket的所有者。
该接口调用成功后将返回204，表示该配置已成功删除。
因为每天0点执行删除，如果你发现某些东西不能删，那最好提前用Delete接口去掉配置，以免丢失数据。

请求

语法

```
DELETE /?lifecycle HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

不使用请求内容

响应

响应头部

只带有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

无

特殊错误

不返回任何特殊错误。如果该bucket没有配置lifecycle，也是返回204，无任何错误。

示例

请求

```
DELETE /?lifecycle HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Authorization: authorization string
Date: Tue, 19 Jul 2016 09:12:54 GMT
```

响应

```
HTTP/1.1 204 No Content
Date: Tue, 19 Jul 2016 09:28:34 GMT
Server: Tengine
X-Kss-Request-Id: a5398b800e3a4d498bb4a7e017ee9259
Content-Length: 0
```

PUT Bucket Replication

描述

此接口会在源存储空间（Bucket）上设置跨区域复制规则。每个源桶只能设置一条复制规则，如果某个存储空间之前已经有跨区域复制规则，则会提示已存在跨区域复制规则。用户设置规则后，KS3内部会按照用户的设置，自动的将匹配到的Object复制到目标存储空间。

权限

如果要使用此接口，您需要对这个存储空间具有PUT BucketReplication权限。

请求

语法

```
PUT /?crr HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
Content-length: {length}
Content-MD5: {md5}

<Replication xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <prefix>prefix1</prefix>
    <prefix>prefix2</prefix>
    <DeleteMarkerStatus>Enabled</DeleteMarkerStatus>
    <targetBucket>bucketname</targetBucket>
</Replication>
```

请求参数

该接口不使用请求参数。

请求头部

名称	描述	必须
Content-MD5	body里data的128位md5 digest，再用base64编码。这个header必须存在，以便检查body是否损坏。详见RFC-1864 类型：String 默认值：无	是

请求体

一段描述跨区域复制configuration的xml。

```
<Replication xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <prefix>prefix1</prefix>
    <prefix>prefix2</prefix>
    <DeleteMarkerStatus>Enabled</DeleteMarkerStatus>
    <targetBucket>bucketname</targetBucket>
</Replication>
```

xml主要参数：

名称	描述	是否必须
Replication	包含跨区域复制规则的容器，一个源存储空间只能有一条规则 类型：Container 子节点：无 父节点：无	是
prefix	前缀匹配，如果object匹配了前缀规则才会对该对象进行复制，每条跨区域复制规则最多添加5条前缀匹配规则，且前缀之间不同重叠 类型：String 父节点：Replication	否
DeleteMarkerStatus	指明是否开始删除复制，若显式指定为Enabled为开启，若为Disabled或不指定均为关闭状态，若开启删除复制，则当源Bucket删除一个对象时，该对象在目标Bucket的副本也会删除 类型：String 父节点：Replication	否
targetBucket	跨区域复制的目标存储空间 类型：String 父节点：Replication	是

注意：

- 每个源存储空间只能设置一条跨区域复制规则，但KS3支持双向复制，即一个源存储空间可以作为当前目标空间的目标空间。
- 当存储空间没有开启与其他存储空间的复制时才能开始跨区域复制。例如空间A开启了到空间B的复制，那么就不能再为空间A开启到空间C的复制，除非先删除空间A到空间B的复制配置。同理，若空间A开启了到空间B的复制，此时再开启空间C到空间B的复制也是不允许的。
- 跨区域复制规则最多添加5条前缀匹配规则，且前缀之间不同重叠。
- 目标存储空间中的对象是源空间的精确副本，它们具有相同的对象名、元数据以及内容，例如创建时 间、拥有者、存储类型、用户定义的元数据、Object ACL、对象内容，对象加密方式(KS3托管秘钥的 加密方式)，因此以上任何数据发生变化都会将变化内容同步到目标端。
- 复制到目标端的文件遵循目标存储空间的生命周期规则。
- 要删除处于复制关系中的存储空间，必须先关闭该复制关系才能将存储空间删除。
- 跨区域复制的优先级高于生命周期管理，当一个对象开始执行生命周期管理操作时，首先将其复制到目 标存储空间然后再执行生命周期管理操作，以满足数据安全高于数据管理的核心原则。
- 请注意，以下文件或数据将不会被复制：
    - 采用客户端加密的数据；
    - 源空间中新增的数据是来自其它空间复制的数据；
    - 存储空间级别的配置更新行为，不会进行文件复制，因为存储空间的配置不会作用到文件上；
    - 如果源文件的存储类型为归档存储，除非文件内容发生变化，否则不会对其进行复制。

响应

响应头部

本接口只带有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

不返回内容。

特殊错误

Error Code	描述	HTTP Status Code
NoSuchBucketReplicationConfiguration	此存储空间没有跨区域复制规则配置	404 Not Found
Invalid Argument	该存储空间已配置跨区域复制规则 请求体格式错误 前缀超过5条 前缀超出长度限制 前缀含有特殊字符等	400 Invalid Argument

示例

请求示例

```
PUT /?crr HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 14 May 2014 02:11:21 GMT
Content-MD5: xxx
Authorization: authorization string
Content-Length: yyy
Content-type: application/xml

<Replication>
    <prefix>abc</prefix>
    <prefix>xyz</prefix>
    <DeleteMarkerStatus>Enabled</DeleteMarkerStatus>
    <targetBucket>targetbucket</targetBucket>
</Replication>
```

以上在某个存储空间上设置跨区域复制规则，其中复制规则匹配两条前缀规则（“abc”和“xyz”），且启用删除同步功能，目标存储空间为targetbucket。

响应示例

```
HTTP/1.1 200 OK
Content-Length: 0
Date: Tue, 19 Jul 2017 09:14:23 GMT
Server: Tengine
x-kss-request-id: 6af24440694b4d00b8de063ccbe86336
```

GET Bucket Replication

描述

此接口返回源存储空间的跨区域复制配置，即描述规则的一个xml。

权限

如果要使用此接口，您需要对该存储空间具有GET BucketReplication权限。

请求

语法

```
GET /?crr HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

请求参数

该接口不使用请求参数。

请求头部

只使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

不使用请求内容。

响应

响应头部

本接口只带有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

返回一个xml，可能有下列节点：

名称	描述	是否必须
Replication	包含跨区域复制规则的容器，一个源存储空间只能有一条规则。 类型：Container 子节点：无 父节点：无	是
prefix	前缀匹配，如果object匹配了前缀规则才会对该对象进行复制，每条跨区域复制规则最多添加5条前缀匹配规则，且前缀之间不同重叠。 类型：String 父节点：Replication	否
DeleteMarkerStatus	指明是否开始删除复制，若显式指定为Enabled为开启，若为Disabled或不指定均为关闭状态，若开启删除复制，则当源Bucket删除一个对象时，该对象在目标Bucket的副本也会删除。 类型：String 父节点：Replication	否

targetBucket	跨区域复制的目标存储空间 类型：String 父节点：Replication	是

特殊错误

Error Code	描述	HTTP Status Code
NoSuchBucketReplicationConfiguration	此复制没有跨区域复制规则配置	404 Not Found

示例

请求示例

```
GET /?crr HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 14 May 2014 02:11:21 GMT
Content-MD5: xxx
Authorization: authorization string
Content-Length: yyy
Content-type: application/xml
```

以上在某个存储空间上设置跨区域复制规则，其中复制规则匹配两条前缀规则（“abc”和“xyz”），且启用删除同步功能，目标存储空间为targetbucket。

响应示例

```
HTTP/1.1 200 OK
Content-Length: 0
Date: Tue, 19 Jul 2017 09:14:23 GMT
Server: Tengine
x-kss-request-id: 6af24440694b4d00b8de063ccbe86336
<Replication xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <prefix>abc</prefix>
    <prefix>xyz</prefix>
    <DeleteMarkerStatus>Enabled</DeleteMarkerStatus>
    <targetBucket>bucketname</targetBucket>
</Replication>
```

Delete Bucket Replication

描述

此接口会关闭源存储空间上设置的跨区域复制规则，如果源存储空间没有配置跨区域复制则返回204。

权限

如果要使用此接口，您需要是这个对该桶具有DeleteBucketReplication权限。

请求

语法

```
Delete /?crr HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

请求参数

该接口不使用请求参数。

请求头部

特殊错误

不返回任何特殊错误。如果该存储空间没有配置跨区域复制，则返回204，无任何错误。

响应

响应头部

只带有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

无

示例

请求示例

```
Delete /?crr HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2016 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Tue, 19 Jul 2016 09:28:34 GMT
Server: Tengine
X-Kss-Request-Id: 54a47bda18ac4e6e91de369add54218e
Content-Length: 0
```

DELETE Object

描述

此接口主要实现删除对象操作（如果存在）。如果不存在，则不做任何操作。

请求

语法

```
DELETE /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
```

```
Content-Length: {length}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该请求使用常用请求头部。获取更多信息，请点击常用请求头部。

响应

响应头部

响应内容

该接口不返回响应内容。

特殊错误

该请求不返回任何特殊错误。

示例

请求示例

```
DELETE /my-image.jpg HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Fri, 26 Dec 2014 06:52:15 GMT
Authorization: authorization string
Content-Type: text/plain
```

响应示例

```
HTTP/1.1 204 NoContent
Date: Fri, 26 Dec 2014 06:52:15 GMT
Content-Length: 0
Connection: close
Server: Tengine
```

接口细节分析

- 该操作是不可逆的，删除后数据将无法恢复。

GET Object

描述

此GET操作将从KS3获取object。你需要具有对文件的读权限。如果你授予匿名用户读权限，那么该操作可以不做签名。如果期望生成一个下载链接，可以参考 通过 URL QueryString 发送签名。

KS3的bucket并不具有目录层次，类似于桌面电脑的文件目录。 但是，你可以通过为object key赋予一个具有文件结构的值来实现逻辑分层。例如，对于sample.jpg，你可以命名它为 photos/2006/February/sample.jpg。

此GET操作，你可以通过指定object的全称来获取一个逻辑分层的object。例如，如果你拥有一个名称为 photos/2006/February/sample.jpg 的object，它放在名字为 examplebucket 的 bucket 中，那么你可以认为资源逻辑名称为 /examplebucket/photos/2006/February/sample.jpg。

权限

该接口操作需要用户对Object拥有READ权限。如果请求的 object 不存在，KS3将根据你是否拥有读权限返回相应信息。

- 如果你拥有该Bucket的READ权限，KS3将会返回的 HTTP 状态码为404(no such key)错误。
- 如果你并不拥有该Bucket的READ权限，KS3将会返回的 HTTP 状态码为403(access denied)错误。
- 响应头x kss tagging count的返回需要访问者具有读该对象标签的权限（ks3:GetObjectTagging ）。若当用户仅有ks3:GetObject权限，但没有ks3:GetObjectTagging权限时，GetObject请求的响应头不返回示x-kss-tagging-count。

请求

语法

```
GET /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
Range:bytes=byte_range
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

有时也许你需要在 GET 响应中返回某一个确定的响应头部值，比如，你可能需要在你的 GET 请求中设置响应头部 Content-Disposition 的值。

你可以使用下表中所列出的查询参数来设置响应头部的值。只有当KS3返回200状态码时，设置的header才会生效。可以通过以下参数设置返回的Content-Type, Content-Language, Expires, Cache-Control, Content-Disposition, 和 Content-Encoding。

参数	描述	必须
response-content-type	设置响应头部 Content-Type 类型：String 默认值：None	否
response-content-language	设置响应头部 Content-Language 类型：String 默认值：None	否
response-expires	设置响应头部 Expires 类型：String 默认值：None	否
response-cache-control	设置响应头部 Cache-Control 类型：String 默认值：None	否

response-content-disposition	设置响应头部 Content-Disposition 类型：String 默认值：None	否
response-content-encoding	设置响应头部 Content-Encoding 类型：String 默认值：None	否

请求头部

该接口可以使用所有常用请求头部，此外，也可以使用下表所列请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必须
Range	下载指定 range 字节的 object。更多关于HTTP Range 头部信息，请访问http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.35 类型：String 默认值：None 约束条件：无	否
If-Modified-Since	如果 object 在指定时间后被改变，则返回 object。否则，返回304状态码 类型：String 默认值：None 约束条件：无	否
If-Unmodified-Since	如果 object 在指定时间后没有被改变，则返回 object。否则，返回412状态码 类型：String 默认值：None 约束条件：无	否
If-Match	如果 object 的 ETag(entity tag)与指定值一致，则返回 object。否则，返回412状态码 类型：String 默认值：None 约束条件：无	否
If-None-Match	如果 object 的 ETag(entity tag)与指定值不一致，则返回 object。否则，返回304状态码 类型：String 默认值：None 约束条件：无	否

请求内容

该接口不使用请求内容。

响应

响应头部

名称	描述
Content-MD5	返回文件md5值的base64编码，前提条件为文件是通过PUT或POST上传到KS3；如果是分块上传的文件，将不会返回此响应头。
x-kss-meta-*	如果你在PUT Object中使用了用户元数据，格式为前缀 x-kss-meta- 后缀为你自定的字段，那么响应头部会返回它，并不解析。 类型：String
x-kss-storage-class	如果文件存储类型为低频存储，值为STANDARD_IA；如果文件存储类型为归档存储，值为ARCHIVE；如果文件存储类型为标准存储，不返回此响应头。 类型：String
ETag	用于标识Object内容的32位十六进制字符串，不同内容的Object对应着不同的ETag。对于Put Object或Post Object请求创建的Object，ETag值是其内容的MD5值；对于分块上传方式创建的Object，ETag值是每个分块的MD5值进行字符串拼接后再次计算所得到的MD5值。ETag值可以用于检查Object内容是否发生变化。 类型：String
x kss tagging count	对象关联的标签的个数。仅当用户有读取标签权限时返回。 类型：String

响应内容

该接口不返回响应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
GET /my-image.jpg HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Fri, 26 Dec 2014 06:48:45 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-kss-request-id: 5a868ca0ebd74bcc8eff1f1a7c9bcd6c
Date: Fri, 26 Dec 2014 06:48:46 GMT
Last-Modified: Wed, 12 Oct 2009 17:50:00 GMT
ETag: "798308f9638ea5b28d5f25e4d677ffac"
Content-Length: 434234
Content-Type: text/plain
Connection: close
Server: Tengine
Content-MD5:eYMI+WOOpbKNXyXk1nf/rA==
[434234 bytes of object data]
```

注意：

1、如果文件通过分块上传保存在KS3，对文件调用GET接口时将不会返回Content_MD5响应头；

2、如果文件通过PUT或POST接口上传到KS3，对文件调用GET接口将会返回Content_MD5响应头。

接口细节分析

- 当使用参数设置返回的header时，只有KS3返回200时才会生效。
- 当设置Range下载时，注意Range的格式不能错，否则KS3会返回416。
- x kss tagging count请求头的返回需要访问者具有读取标签的权限（ks3:GetObjectTagging ）。即当用户仅有ks3:GetObject权限，但没有ks3:GetObjectTagging权限时，GetObject请求的响应头不返回示x-kss-tagging-count。
- 对于私有文件想通过浏览器下载的需求，可以参考通过 URL QueryString 发送签名。

HEAD Object

描述

此HEAD操作将会在不返回 object 的情况下获取对象的元数据信息。如果你只需要对象的元数据信息，那么这个方法非常合适。 使用此接口，你需要具有对对象的 READ 权限。

一个对象的HEAD请求与GET请求具有相同的操作，唯一的区别是响应回复中HEAD请求不具有响应体。

权限

该接口操作需要用户对Object拥有READ权限。如果请求的 object 不存在，KS3将根据你是否拥有对权限返回相应信息。

- 如果你拥有该Bucket的READ权限，KS3将会返回的 HTTP 状态码为404(no such key)错误。
- 如果你并不拥有该Bucket的READ权限，KS3将会返回的 HTTP 状态码为403(access denied)错误。

请求

语法

```
HEAD /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求头部。

请求头部

该接口可以使用所有常用请求头部，此外，也可以使用下表所列请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必须
Range	下载指定 range 字节的 object。更多关于HTTP Range 头部信息，请访问http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.35 类型：String 默认值：None 约束条件：无	否
If-Modified-Since	如果 object 在指定时间后被改变，则返回 object。否则，返回304状态码 类型：String 默认值：None 约束条件：无	否
If-Unmodified-Since	如果 object 在指定时间后没有被改变，则返回 object。否则，返回412状态码 类型：String 默认值：None 约束条件：无	否
If-Match	如果 object 的 ETag(entity tag)与指定值一致，则返回 object。否则，返回412状态码 类型：String 默认值：None 约束条件：无	否
If-None-Match	如果 object 的 ETag(entity tag)与指定值不一致，则返回 object。否则，返回304状态码 类型：String 默认值：None 约束条件：无	否

请求内容

该接口不使用请求内容。

响应

响应头部

名称	描述
Content-MD5	返回文件md5值的base64编码，前提条件为文件是通过PUT或POST上传到KS3；如果是分块上传的文件，将不会返回此响应头。
x-kss-meta-*	如果你在PUT Object中使用了用户元数据，格式为前缀 x-kss-meta- 后缀为你自定的字段，那么响应头部会返回它，并不解析。 类型：String
x-kss-restore	对解冻中的或已解冻的归档文件进行head请求时，会增加x-kss-restore请求头，请求头的内容指明解冻的状态以及解冻过期时间，解冻中的文件会返回x-kss-restore: ongoing-request="true"，已解冻的文件会返回，例如：x-kss-restore: ongoing-request="false", expiry-date="Sun, 16 Apr 2017 08:12:33 GMT"；对标准存储文件、低频存储文件和未解冻的归档文件，进行Head请求时，不会增加x-kss-restore请求头。 类型：String
x-kss-storage-class	如果文件存储类型为低频存储，值为STANDARD_IA；如果文件存储类型为归档存储，值为ARCHIVE；如果文件存储类型为标准存储，不返回此响应头。 类型：String
ETag	用于标识Object内容的32位十六进制字符串，不同内容的Object对应着不同的ETag。对于Put Object或Post Object请求创建的Object，ETag值是其内容的MD5值；对于分块上传方式创建的Object，ETag值是每个分块的MD5值进行字符串拼接后再次计算所得到的MD5值。ETag值可以用于检查Object内容是否发生变化。 类型：String
x kss tagging count	对象关联的标签的个数。仅当用户有读取标签权限时返回。 类型：String

响应内容

该接口不返回响应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
HEAD /my-image.jpg HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2006 12:00:00 GMT
ETag: "54a3be97af36cdc9f2516c74550fa95d"
x kss tagging count:2
Content-Length: 434234
Content-Type: text/plain
Connection: close
Server: Tengine
Content-MD5:VKO+1682zcnyUWxOVQ+pXQ==
```

注意：

1、如果文件通过分块上传保存在KS3，对文件调用GET接口时将不会返回Content_MD5响应头；

2、如果文件通过PUT或POST接口上传到KS3，对文件调用GET接口将会返回Content_MD5响应头。

接口细节分析

- HEAD Object不论请求成功与否，都不会返回body。
- 使用该接口可以用来判断object是否存在。
- 使用该接口可以用来获取object的元数据。
- x kss tagging count请求头的返回需要访问者具有读取标签的权限（ks3:GetObjectTagging ）。即当用户仅有ks3:GetObject权限，但没有ks3:GetObjectTagging权限时，HeadObject请求的响应头不返回示x-kss-tagging-count。

PUT Object

描述

此PUT接口可以添加一个 object 到某个 bucket。 如果要使用此接口，你需要具有对要添加对象的空间的 WRITE 权限。

KS3不会添加不完整对象，如果你收到成功的响应，那么KS3已经成功添加对象到相应空间中。

KS3是一个分布式系统。如果同时受到多个相同的对象的写请求，它会覆盖所有相同对象，只保留最后一个对象。KS3不提供在写对象时，为对象加锁服务，如果你确实需要，请在你的应用层实现它。

为了保证数据在传输过程中没有损坏，请使用 Content-MD5 头部。当使用此头部时，KS3会自动计算出MD5，并根据用户提供的MD5进行校验，如果不匹配，将会返回错误信息。另外，当用户上传一个对象到KS3时，可以根据返回的 ETag 计算出对象的MD5值。

注意：使用此接口只能上传5G以内的文件，超过5G的文件请使用分块上传 。

权限

当你上传某个对象时，你可以随意指定任意数量的组或个人，授予权限。使用请求头部，有两种途径能够赋予有效权限。

- 使用 x-kss-acl 请求头部，指定一个预定义的 ACL。
- 使用 x-kss-grant-read,x-kss-grant-write,x-kss-grant-full-control 请求头部，来明确指定具体的访问权限。

对象标签权限

- 若您需要在上传某个对象的同时为其指定标签，那您还需要具有ks3:PutObjectTagging权限，用来添加/更新对象的标签，对象写权限与写对象标签权限相互独立。

请求

语法

```
PUT /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部，此外，也可以使用下表所列请求头部。获取更多信息，请点击常用请求头部 。

名称	描述	必须
Cache-Control	告诉所有的缓存机制是否可以缓存及哪种类型。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.9 类型：String 默认值：None 约束条件：无	否
Content-Disposition	指定对象的表达信息。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec19.html##sec19.5.1 类型：String 默认值：None 约束条件：无	否
Content-Encoding	指定文件内容编码格式。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.11 类型：String 默认值：None 约束条件：无	否
Content-Length	指明对象的大小，按字节。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.13 类型：String 默认值：None 约束条件：无	是
Content-MD5	对消息内容（不包括头部）计算MD5值，获得128比特位数字，然后对该数字进行base64编码，用于对象完整性校验。 类型：String 默认值：None 约束条件：无	否
Content-Type	用于描述文件内容MIME格式。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.17 类型：String 默认值：binary/octet-stream 有效值：MIME types 约束条件：无	否
Expect	当你使用 100-continue 时，直到收到确认时才会发送请求体。如果头部信息被拒绝，请求体不会被发送。 类型：String 默认值：None 有效值：100-continue 约束条件：无	否
Expires	对象存在于缓存的有效时间日期。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.21 类型：String 默认值：None 约束条件：无	否
x-kss-meta-	用户元数据前缀标识。若某个头部前缀为 x-kss-meta-， 则为用户自定义元数据。 类型：String 默认值：None 约束条件：无	否

x-kss-storage-class	设置存储方式。 类型：String 默认值：None 有效值：STANDARD/STANDARD_IA/ARCHIVE 说明：当不指定x-kss-storage-class时，如果Bucket是归档类型，Object自动为归档类型，如果Bucket是非归档类型，Object自动为标准类型；如果指定x-kss-storageClass，则为指定存储类型。 约束条件：无	否
x-kss-content-maxlength	设置上传文件最大允许大小。当Content-Length大于该值时，KS3将会返回403。 类型：String 默认值：None 约束条件：无	否
x-kss-newfilename-in-body	控制台设置文件名 后，指定返回的文件名字是否出现body中。true表示在header和body中返回；false表示只在header中返回。 类型：boolean 默认值：None 有效值：false\true 约束条件：无	否
x-kss-tagging	指定目标Object对象标签，可同时设置多个标签， 如：TagA=A&TagB=B。 **说明：Key和Value需要先进行URL编码** 如果某项没有”=“，则看作Value为空字符串,详见对象标签。	否

ACL 特殊头部

用户可以通过以下的header为Object设置预设的ACL

名称	描述	必须
x-kss-acl	用于对象的预定义权限。 类型：String 默认值：private 有效值：private | public-read 约束条件：无	否

如果用户期望为Bucket设置详细的ACL，可以通过以下header设置

名称	描述	必须
x-kss-grant-read	为若干用户授予READ权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-write	为若干用户授予WRITE权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限。 类型：String 默认值：无 约束条件：无	否

以上header值的值为以一个逗号",”分割的授权列表。每个授权信息的格式为type=value,当前type支持id：

- id:被授权者的用户id

例如，要给id为1234578和3344211的两个用户授予READ权限: x-kss-grant-read:id="1234578",id="3344211"

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String
newfilename	在控制台设置文件名 后的新文件名。 类型：String

响应内容

该接口不返回响应内容。

特殊错误

该请求不返回任何特殊错误。

示例

请求示例

```
PUT /my-image.jpg HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 12 Oct 2009 17:50:00 GMT
Authorization: authorization string
Content-Type: text/plain
x-kss-tagging:TagA=A&TagB=B
Content-Length: 11434
Expect: 100-continue
[11434 bytes of object data]
```

响应示例

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
Date: Wed, 12 Oct 2009 17:50:00 GMT
ETag: "1b2cf535f27731c974343645a3985328"
Content-Length: 0
Connection: close
Server: Tengine
```

接口细节分析

- 用户可以在请求头中添加Content-MD5,要求KS3对文件的完整性做校验。
- 当服务器上文件已存在时，若上传成功，将会导致覆盖。
- 用户可以在上传的时候添加x-kss-meta-*的header，为Object添加用户元数据。
- 支持在上传时指定对象tag，请求者（主账号，子用户，角色）需要具有ks3:GetObjectTagging操作授权。
- Content-Length是必须的，且不能大于body中的实际数据大小。
- 当访问者具有ks3:PutObject权限，但没有ks3:PutObjectTagging权限时，PutObject仅允许上传不带tagging的对象。
- 关于如何下载上传上去的object，请详见GET Object

POST Object

描述

此POST操作将使用HTML表单为指定的空间添加一个对象。POST是PUT的另外一种选择，为了方便用户空间可以基于浏览器的上传对象。 参数通过POST以表单域的形式将数据编码封装到消息体传递，而不是PUT方式。用户需要拥有对空间的写权限才能添加新的对象。

KS3不会添加不完整对象，如果你收到成功的响应，那么KS3已经成功添加对象到响应空间中。

KS3是一个分布式系统。如果同时受到多个相同的对象的写请求，它会覆盖所有相同对象，只保留最后一个对象。KS3不提供在写对象时，为对象加锁服务，如果你确实需要，请在你的应用层实现它。

请求

语法

```
POST / HTTP/1.1
Host: {BucketName}.{endpoint}
User-Agent: {browser_data}
Accept: {file_types}
Accept-Language: {Regions}
Accept-Charset: {character_set}
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: {length}

--9431149156168
Content-Disposition: form-data; name="key"

{ObjectKey}
--9431149156168
Content-Disposition: form-data; name="KSSAccessKeyId"

{AccessKey}
--9431149156168
Content-Disposition: form-data; name="Policy"

{Policy}
--9431149156168
Content-Disposition: form-data; name="Signature"

{Signature}
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg

file_content
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to KS3
--9431149156168--
```

请求参数

该请求不使用请求参数。

请求头部

该请求只使用常用请求头部。获取更多信息，请点击常用请求头部

表单域

名称	描述	必需
acl	指定访问控制权限，如果指定的访问权限列表无效，则会返回错误。 类型: String 默认值: private 有效值: private &##124; public-read	否
Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires	REST 特别头部. 更多信息，请查看PUT Object。 类型: String 默认值: 无	否
file	文件或文本内容。用户每次只能上传一个文件，且内容不能放到key元素的前面，否则返回400。 类型: 文件或文本内容 默认值: 无	是
key	ObjectKey。如果用户想要使用文件名作为Key，可以使用${filename} 变量。例如：如果用户想要上传文件local.jpg，需要指明specify /user/betty/${filename}，那么键值就会为/user/betty/local.jpg。 类型: String 默认值: 无	是
KSSAccessKeyId	KSSAccessKeyId。 类型：String 默认值:无 约束条件:当 bucket 非 public-read-write 或者提供了policy表单域时，必须提供该表单域。	视情况而定
policy	请求中用于描述获准行为的安全策略。没有安全策略的请求被认为是匿名请求，只能访问公共可写空间。 类型: String 默认值: 无 约束条件: 当 bucket 非 public-read-write.必须提供该表单域。 详见：Post Policy	视情况而定
signature	根据Access Key Secret和policy计算的签名信息，KS3验证该签名信息从而验证该Post请求的合法性。 类型: String 默认值: 无 约束条件： 当 bucket 非 public-read-write 或者提供了policy表单域时，必须提供该表单域。详见：Post Policy	视情况而定
success_action_redirect, redirect	成功上传后客户端的重定向URL。如果用户没有指定success_action_redirect，KS3将会返回空文件类型。 如果KS3无法解析URL地址，那么会无视此表单域。如果上传失败，KS3不会将客户端重定向。 类型: String 默认值: 无	否
success_action_status	返回的状态码，如果没有指定，则依赖于上传的成功状态。允许值为200，,201，204(默认值)。如果是200或204,KS3将返回一个状态为200或204的空文件。如果状态码为201，那么KS3将会返回一个状态码为201的XML文档。如果值无效，或者没有设定，KS3将会返回一个状态码为204的空文档。 类型: String 默认值: 无	否
x-kss-meta-	用户元数据前缀标识。若某个头部前缀为 x-kss-meta-， 则为用户自定义元数据。 类型: String 默认值: 无 约束条件：无	否

x-kss-storage-class	设置文件的存储类型。 类型：String 默认值：无 有效值：STANDARD/STANDARD_IA/ARCHIVE 说明：当不指定x-kss-storage-class时，如果Bucket是归档类型，Object自动为归档类型，如果Bucket是非归档类型，Object自动为标准类型；如果指定x-kss-storageClass，则为指定存储类型 约束条件：无	否	
x-kss-newfilename-in-body	控制台设置文件名后，指定返回的文件名字是否出现body中。true表示在header和body中返回；false表示只在header中返回。 类型：boolean 默认值：None 有效值：false	true 约束条件：无	否
tagging	指定标签添加到对象 Default: None	否	

服务端加密请求则需要以下表单项

如果用户需要服务器使用默认加密，需要以下表单项

名称	描述	必需
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则需要配置该表单项，值为使用的加密算法，目前支持AES256。 类型：String	是

如果用户需要服务器按照用户提供的秘钥加密，需要以下头部

名称	描述	必需
x-kss-server-side-encryption-customer-key	由用户指定KS3加密时使用的 base64-encoded 加密秘钥。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-algorithm,x-kss-server-side-encryption-customer-key-MD5 同时使用。	是
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-key-MD5 同时使用。	是
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-algorithm 同时使用。	是

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
success_action_redirect, redirect	成功上传后客户端的重定向URL。 类型:String 父节点：PostResponse
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String
newfilename	在控制台设置文件名后的新文件名。 类型：String

特殊错误

该请求不返回任何特殊错误。

接口细节分析

- 用户需要对Bucket拥有写权限。
- 除非匿名用户对Bucket有写权限，否则需要在表单域中提供KSSAccessKeyId、policy和signature三项，具体构造方法见Post Policy。
- POST Object时，Content-Type必须为 multipart/form-data。
- policy中规定了表单需要满足的规则，表单必须满足policy中定义的规则，方可上传成功。policy中必须包含表单中除KSSAccessKeyId、policy、signature和file外的所有表单项。
- 支持在上传时指定对象tagging，请求者（主账号，子用户，角色）需要具有ks3:PutObjectTagging操作授权,若访问者仅具有ks3:PutObject权限，但没有ks3:PutObjectTagging权限时，PutObject仅允许上传不带tagging的对象。
- 用户可以在表单项中添加以x-kss-meta开头的表单项，视为添加用户元数据。注意，这些表单项也是需要在policy中定义的。
- 当KS3返回403时，应该首先检查表单是否满足policy。

GET Object ACL

描述

此GET操作使用 acl 子资源来返回 object 的 ACL(access control list)。 只有bucket的所有者拥有调用该接口的权限。

请求

语法

```
GET /{ObjectKey}?acl HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口可以使用所有常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

名称	描述
AccessControlList	包含 Grant,Grantee,Permission 的容器 类型：Container 父节点：AccessControlPolicy
AccessControlPolicy	包含了每一个 Grantee 对于某个对象的 ACL 权限设置信息 类型：Container 父节点：无
Grant	包含被授权者和其权限信息。 类型：String 父节点：AccessControlPolicy.AccessControlList
Grantee	被授权者 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant
DisplayName	Bucket拥有者的名称 类型：String 父节点：AccessControlPolicy.Owner
ID	Bucket拥有者的用户ID，或者是被授权者的ID 类型：String 父节点：AccessControlPolicy.Owner 或者 AccessControlPolicy.AccessControlList.Grant
Owner	包含bucket拥有者信息（DisplayName,ID）的容器 类型：Container 父节点：AccessControlPolicy
Permission	指明授予被授权者的权限信息（FULL_CONTROL,READ,WRITE） 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant

特殊错误

该接口不返回任何特殊错误。

示例

下面的请求将会返回相应object的 ACL 信息，my-image.jpg。

请求示例

```
GET /my-image.jpg?acl HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Fri, 26 Dec 2014 07:14:18 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Fri, 26 Dec 2014 07:14:18 GMT
Last-Modified: Sun, 1 Jan 2009 12:00:00 GMT
Content-Length: 124
Content-Type: text/plain
Connection: close
Server: Tengine

<AccessControlPolicy>
<Owner>
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Owner>
<AccessControlList>
<Grant>
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
<ID>73404060</ID>
<DisplayName>ks3_dt@kingsoft.com</DisplayName>
</Grantee>
<Permission>FULL_CONTROL</Permission>
</Grant>
</AccessControlList>
</AccessControlPolicy>
```

PUT Object ACL

描述

此PUT操作使用 acl 的子集为已存在与 bucket 中的 object 设置访问控制权限(ACL)。

只有bucket的所有者可以调用该接口。

你可以使用下面两种方式来设置对象的权限。

- 在请求体中指定 ACL。
- 使用请求头部来设置访问权限。

注意 *不能同时使用以上两种方式。*

请求

语法

下面展示的是通过在请求体中指定 ACL 的方式进行设定。

```
PUT /{ObjectKey}?acl HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}

<AccessControlPolicy>
<Owner>
<ID>{Base64EncodeUserId}</ID>
<DisplayName>{Base64EncodeUserId}</DisplayName>
</Owner>
<AccessControlList>
<Grant>
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
<ID>{Base64EncodeUserId}</ID>
<DisplayName>{Base64EncodeUserId}</DisplayName>
</Grantee>
<Permission>{Permission}</Permission>
</Grant>
...
</AccessControlList>
</AccessControlPolicy>
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该接口不使用请求参数。

请求头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。 用户可以通过以下的header为Object设置预设的ACL

名称	描述	必须
x-kss-acl	用于对象的预定义权限。 类型：String 默认值：private 有效值：private public-read 约束条件：无	否

如果用户期望为Object设置详细的ACL，可以通过以下header设置

名称	描述	必须
x-kss-grant-read	为若干用户授予READ权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限。 类型：String 默认值：无 约束条件：无	否

以上header值的值为以一个逗号","分割的授权列表。每个授权信息的格式为type=value,当前type支持id：

- id:被授权者的用户id

例如，要给id为1234578和3344211的两个用户授予READ权限： x-kss-grant-read:id="1234578",id="3344211"

请求内容

如果用户决定使用请求体来指定访问权限列表，需要使用下表元素。

注意 *如果你使用请求体设置ACL，你不能再通过请求头部设置ACL*

名称	描述
AccessControlList	包含 Grant,Grantee,Permission 的容器 类型：Container 父节点：AccessControlPolicy
AccessControlPolicy	包含了每一个 Grantee 对于某个对象的 ACL 权限设置信息 类型：Container 父节点：无
Grant	包含被授权者和其权限信息。 类型：String 父节点：AccessControlPolicy.AccessControlList
Grantee	被授权者，参考授予权限方式 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant
DisplayName	Bucket拥有者的名称 类型：String 父节点：AccessControlPolicy.Owner
ID	Bucket拥有者的用户ID，或者是被授权者的ID 类型：String 父节点：AccessControlPolicy.Owner 或者 AccessControlPolicy.AccessControlList.Grant
Owner	包含bucket拥有者信息（DisplayName,ID）的容器 类型：Container 父节点：AccessControlPolicy
Permission	指明授予被授权者的权限信息（FULL_CONTROL,READ,WRITE） 类型：String 父节点：AccessControlPolicy.AccessControlList.Grant

授予权限方式

用户可以通过以下方式来授予某个用户对用户空间的权限。

通过用户ID,即根据用户ID授予特定用户权限

{Base64EncodeUserId}{Base64EncodeUserId}

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回相应内容。

特殊错误

该接口不返回任何特殊错误。

示例

请求示例

```
PUT /my-image.jpg ?acl HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Content-Length: 1660
Date: Fri, 26 Dec 2014 06:34:32 GMT
Authorization: authorization string

<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Owner>
<ID>73410125</ID>
<DisplayName>ks3@kingsoft.com</DisplayName>
</Owner>
<AccessControlList>
<Grant>
```

```
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
<ID>73404060</ID>
<DisplayName>ks3_dt@kingsoft.com</DisplayName>
</Grantee>
<Permission>FULL_CONTROL</Permission>
</Grant>
<Grant>
<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
<ID>73406240</ID>
<DisplayName>ks3_op@kingsoft.com</DisplayName>
</Grantee>
<Permission xmlns="">READ</Permission>
</Grant>
</AccessControlList>
</AccessControlPolicy>
```

响应示例

```
HTTP/1.1 200 OK
Date: Fri, 26 Dec 2014 06:34:32 GMT
x-kss-request-id: dbea4ce4ec23415b9e454ecfa25ec4d9
Content-Length: 0
Server: Tengine
```

接口细节分析

- Object的权限含义详见 ACL
- 当同时在header中和Body中设置了ACL，最后只有header中的会生效。当同时在header中设置了x-kss-acl和x-kss-grant-*时，后者生效。
- 对于大部分用户，使用x-kss-acl在header中设置预设的ACL就可以满足大部分需求。

PUT Object Copy

描述

此PUT接口可以拷贝一个在KS3中已经存在 object 到某个 bucket。用户通过在请求中配置请求头部 x-kss-copy-source 来指定要拷贝的数据源。

PUT Object Copy接口能拷贝的源文件大小最大为1GB，如需要拷贝的源文件超过1GB，请使用Upload Part Copy接口 进行分块拷贝。

PUT Object Copy接口的目标桶和源桶必须在同一个region。

用户可以通过此接口实现对象移动、重命名、修改对象元数据和修改存储类型。

除非在此接口请求中显式指定存储类型，否则无论何种处理方式，目标对象均保持为目标bucket的默认存储类型，详情请参考存储空间类型 。

PUT Object Copy接口支持跨账户复制，但用户需要具有拷贝对象的读权限，以及目标空间的写权限。

请求

语法

```
PUT /{destinationObject} HTTP/1.1
Host: {destinationBucket}.{endpoint}
x-kss-copy-source: {/source_bucket/sourceObject}
Authorization: {SignatureValue}
Date: {date}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部，此外，也可以使用下表所列请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必需
x-kss-copy-source	空间名称与对象的object key名称的组合，通过斜杠分隔('/')。 类型：String 默认值：None 约束条件:其值必须使用URL编码。另外，空间名称必须有效，同时用户需要拥有对拷贝对象的读权限。	是
x-kss-metadata-directive	指定是否拷贝源Object的元数据信息，有效值：COPY, REPLACE。 如果此选项不指定，默认为COPY。 COPY：目标Object的元信息会从源Object拷贝（包括用户自定义x-kss-meta-*头和标准HTTP Header头），请求中新指定的信息不生效。 REPLACE：忽略源Object的元信息，目标Object直接采用请求中指定的元信息。 **说明**：如果仅需要修改Object元信息而不拷贝，需将请求的object key和x-kss-copy-source设置为相同并且设置本选项为REPLACE，则会修改为请求中新指定的信息。 另外需要注意如果原来object的元信息有多个，则除 Content-MD5， content-part-md5， Content-Length， server-side-encryption-customer-key-MD5之外，object其他无需修改的元信息也需要加上； 如果源Object采用了加密存储，则同时需要指定x-kss-service-side-encryption-*和x-kss-copy-source-server-side-encryption-*为源文件加密信息。	否
x-kss-storage-class	设置存储类型。 类型：String 默认值：None 有效值：STANDARD/STANDARD_IA/ARCHIVE 存储类型参考[存储类型介绍]。(https://docs.ksyun.com/documents/6756) 。 说明： 1.当不上传x-kss-storage-class时，如果Bucket是归档类型，Object自动为归档类型，如果Bucket是非归档类型，Object自动为标准类型； 2.若原文件存储类型为ARCHIVE，必须为解冻状态才可修改成功	否
x-kss-tagging	指定目标Object对象标签，可同时设置多个标签，如：TagA=A&TagB=B。 说明 Key和Value需要先进行URL编码，如果某项没有“=”，则看作Value为空字符串。详情请见对象标签。	否
x-kss-tagging-directive	指定如何设置目标Object的对象标签。 默认值：COPY 有效值： 1. COPY（默认值）：复制源Object的对象标签到目标 Object。 2. REPLACE：忽略源Object的对象标签，直接采用请求中指定的对象标签。	否

加密相关请求头部

注意：如果源Object没有进行过加密，不支持对目标Object进行加密。且目标Object的加密方式要与源Object的加密方式一致，如果源Object是客户提供的密钥加密，目标Object应该使用同样的加密密钥。

如果用户需要对目标**Object**使用默认加密，需要以下头部

名称	描述	必需

x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String	是

如果用户需要对目标**Object**使用用户提供的秘钥加密，需要以下头部

名称	描述	必需
x-kss-server-side-encryption-customer-key	由用户指定KS3加密时使用的 base64-encoded 加密秘钥。其值必须与源Object创建时使用的秘钥一致。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-algorithm,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-algorithm 同时使用。	是
x-kss-copy-source-server-side-encryption-customer-key	由用户指定KS3解密时使用的 base64-encoded 加密秘钥，其值必须与源Object创建时使用的秘钥一致。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-algorithm,x-kss-server-side-encryption-customer-key-MD5 同时使用。	是
x-kss-copy-source-server-side-encryption-customer-algorithm	指定数据源对象解密使用的解密算法。 类型：String 有效值：AES256 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-copy-source-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-algorithm 同时使用	是
x-kss-tagging	指定Object的对象标签，可同时设置多个标签，如：TagA=A&TagB=B。 说明 Key和Value需要先进行URL编码，如果某项没有“=”，则看作Value为空字符串。	否
x-kss-tagging-directive	指定如何设置目标Object的对象标签。 默认值：Copy 有效值： 1. COPY（默认值）：复制源Object的对象标签到目标 Object。 2. REPLACE：忽略源Object的对象标签，直接采用请求中指定的对象标签。	否

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String

响应内容

名称	描述
CopyObjectResult	响应内容的容器。 类型：Container 父节点：无
ETag	返回一个新对象的实体标签。影响因素为对象的内容与，对象名称或元数据无关。 类型：String 父节点：CopyObjectResult
LastModified	返回最后被修改的时间日期。 类型：String 父节点：CopyObjectResult

特殊错误

该请求不返回任何特殊错误。

示例

普通复制请求示例

```
PUT /rename-image.jpg HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
x-kss-copy-source: /SourceBucket/SourceObjectKey
x-kss-tagging-directive:REPLACE
x-kss-tagging:TagA=A&TagB=B//需要URLencode
Authorization: authorization string
```

普通响应示例

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2009 22:32:00 GMT
Connection: close
Server: Tengine
```

修改元数据请求示例

```
PUT /modifymetakey HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
x-kss-copy-source: /ks3-example/modifymetakey
x-kss-metadata-directive:REPLACE
x-kss-meta-youmetakey:yourmetavalue
Authorization: authorization string
```

修改元数据响应示例

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2009 22:32:00 GMT
Connection: close
Server: Tengine
```

```
<CopyObjectResult>
<LastModified>2009-10-28T22:32:00</LastModified>
<ETag>"9b2cf535f27731c974343645a3985328"</ETag>
</CopyObjectResult>
```

修改Object存储类型请求示例

```
PUT /yourobjectkey HTTP/1.1
Host: yourbucket.ks3-cn-beijing.ksyun.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
x-kss-copy-source: /yourbucket/yourobjectkey
x-kss-storage-class:STANDARD_IA
Authorization: authorization string
```

修改Object存储类型响应示例

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2009 22:32:00 GMT
Connection: close
Server: Tengine

<CopyObjectResult>
<LastModified>2009-10-28T22:32:00</LastModified>
<ETag>"9b2cf535f27731c974343645a3985328"</ETag>
</CopyObjectResult>
```

接口细节分析

- 用户必须对源bucket的Object具有读权限，对目标Bucket具有写权限。
- 如果您要复制的目标存储中同名key已经存在，则KS3会返回400。
- 复制默认会将源Object的对象标签到目标 Object，操作成功的权限是调用者，需要具有源文件ks3:GetObjectTagging 与目标桶的写文件与写标签权限。
- 当源与目标为同一个桶且Key相同时，支持该接口实现对同一文件的修改存储类型与元数据的目的。 object 可修改的元数据包括 ：
  1. HTTP Header头：Content-Type，Content-Length， Cache-Control， Content-Disposition， Content-Encoding，Expires
  2. x-kss-meta-
- 如果复制一个带有标签的对象，则同时需要对源文件有ks3:GetObject与ks3:GetObjectTagging权限 同时对目标文件ks3:PutObject与 ks3:PutObjectTagging 权限，但是如果指定x-kss-tagging-directive为REPLACE且新指定tag为空，则目标权限只需ks3:PutObject权限即可。
- 如果需要覆盖源文件对象标签，则x-kss-tagging-directive 需设置为 Replace 且指定x-kss-tagging 合法，这时将采用 x-kss-tagging 指定的对象标签作为目标对象的新标签。若仅 x-kss-tagging 时无法完成重置目标对象的标签的覆盖。

Initiate Multipart Upload

描述

此操作将启动一个分块上传任务并返回 upload ID。在一个确定的分块上传任务中，upload ID用于关联所有分块。连续分块上传请求中的 upload ID由用户指定。在Complete Multipart Upload 和 Abort Multipart Upload请求中同样包含 upload ID。 关于请求签名的问题，分块上传为一系列的请求（初始化分块上传，上传块，完成分块上传，终止分块上传），用户启动任务，发送一个或多个分块，最终完成任务。用户需要对每一个请求单独签名。

> 注意：当你启动分块上传后，并开始上传分块，你必须完成或者放弃上传任务，才能终止因为存储造成的收费。

请求

语法

```
POST /{ObjectKey}?uploads HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

> 注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部，此外，也可以使用下表所列请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必须
Cache-Control	告诉所有的缓存机制是否可以缓存及哪种类型。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.9 类型：String 默认值：None 约束条件：无	否
Content-Disposition	指定对象的表达信息。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec19.html##sec19.5.1 类型：String 默认值：None 约束条件：无	否
Content-Encoding	指定文件内容编码格式。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.11 类型：String 默认值：None 约束条件：无	否
Content-Type	用于描述文件内容MIME格式。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.17 类型：String 默认值：binary/octet-stream 有效值：MIME types 约束条件：无	否
Expires	对象存在于缓存的有效时间日期。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html##sec14.21 类型：String 默认值：None 约束条件：无	否
x-kss-meta-	用户元数据前缀标识。若某个头部前缀为 x-kss-meta-， 则为用户自定义元数据。 类型：String 默认值：None 约束条件：无	否
x-kss-storage-class	设置存储方式。 类型：String 默认值：None 有效值：STANDARD/STANDARD_IA/ARCHIVE 说明：当不指定x-kss-storage-class时，如果Bucket是归档类型，Object自动为归档类型，如果Bucket是非归档类型，Object自动为标准类型；如果指定x-kss-storageClass，则为指定存储类型。 约束条件：无	否
x-kss-tagging	指定Object的标签，可同时设置多个标签， 如：TagA=A&TagB=B。**说明 Key和Value需要先进 URL编码**，如果某项没有”=“，则看作Value为空字符 。	否

ACL 特殊头部

用户可以通过以下的header为Object设置预设的ACL

名称	描述	必须
x-kss-acl	用于对象的预定义权限。 类型：String 默认值：private 有效值：private | public-read 约束条件：无	否

如果用户期望为Bucket设置详细的ACL，可以通过以下header设置。

名称	描述	必须
x-kss-grant-read	为若干用户授予READ权限。 类型：String 默认值：无 约束条件：无	否
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限。 类型：String 默认值：无 约束条件：无	否

以上header值的值为以一个逗号","分割的授权列表。每个授权信息的格式为type=value,当前type支持id：

- id:被授权者的用户id。
- uri:被授权的用户组，暂时value仅支持http://acs.ksyun.com/groups/global/AllUsers，表示所有用户(包括匿名用户) 例如，要给所有用户授予READ权限: x-kss-grant-read:uri="http://acs.ksyun.com/groups/global/AllUsers"

例如，要给id为1234578和3344211的两个用户授予READ权限: x-kss-grant-read:id="1234578",id="3344211"

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String

响应内容

名称	描述
InitiateMultipartUploadResult	包含响应信息的容器 类型：Container 子节点：Bucket, Key, UploadId 父节点：无
Bucket	启动分块上传任务的 bucket 的名字 类型：String 父节点：InitiateMultipartUploadResult
Key	分块上传对象的 key 类型：String 父节点：InitiateMultipartUploadResult
UploadId	分块上传任务的ID 类型：String 父节点：InitiateMultipartUploadResult

特殊错误

该请求不返回任何特殊错误。

示例

请求示例

```
POST /my-video.rm ?uploads HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Mon, 1 Nov 2010 20:34:56 GMT
Content-Length: 197
Connection: keep-alive
Server: Tengine

<?xml version="1.0" encoding="UTF-8"?>
<InitiateMultipartUploadResult xmlns="http://ks3-cn-beijing.ksyun.com/doc/2006-03-01/">
<Bucket>ks3-example</Bucket>
<Key>my-video.rm</Key>
<UploadId>1aa9cfad5e2e405c8f27965feb8b60cc</UploadId>
</InitiateMultipartUploadRes
```

接口细节分析

- 初始化分块上传不会对现有的同名文件造成影响，只有在Complete Multipart Upload之后才会覆盖同名文件。
- 初始化分块上传会在body中返回UploadId，在之后的一些列操作中都会用到。
- 支持在上传时指定对象tagging，请求者（主账号，子用户，角色）需要具有ks3:PutObjectTagging操作授权。
- 当访问者具有ks3:PutObject权限，但没有ks3:PutObjectTagging权限时，PutObject仅允许上传不带tagging的对象。

Upload Part

描述

此操作将在分块上传任务中上传一个块。

在你上传任一块之前你必须先要启动一个分块上传任务。在你发送一个启动请求后，KS3会给你一个唯一的 upload ID。每次上传块时，都需要将上传ID包含在请求中。

块的数量可以是1到10,000中的任意一个（包含1和10,000）。块序号用于标识一个块以及其在对象创建时的位置。如果你上传一个新的块，使用之前已经使用的序列号，那么之前的那个块将会被覆盖。当所有块总大小大于5M时，除了最后一个块没有大小限制外，其余的块的大小均要求在5MB以上。当所有块总大小小于5M时，除了最后一个块没有大小限制外，其余的块的大小均要求在100K以上。如果不符合上述要求，会返回413状态码。

为了保证数据在传输过程中没有损坏，请使用 Content-MD5 头部。当使用此头部时，KS3会自动计算出MD5，并根据用户提供的MD5进行校验，如果不匹配，将会返回错误信息。

注意：当你启动分块上传后，并开始上传分块，你必须完成或者放弃上传任务，才能终止因为存储造成的收费。

请求

语法

```
PUT /{ObjectKey}?partNumber={PartNumber}&uploadId={UploadId} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Content-Length: {Size}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必须
Content-Length	指明对象的大小，按字节。更多信息，请点击http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html## sec14.13 类型：String 默认值：None 约束条件：无	是
Content-MD5	base64加密MD5信息，128位，用于对象完整性校验。 类型：String 默认值：None 约束条件：无	否
Expect	当你使用 100-continue 时，直到收到确认时才会发送请求体。如果头部信息被拒绝，请求体不会被发送 类型：String 默认值：None 有效值：100-continue 约束条件：无	否

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String

响应内容

该接口不使用响应内容。

特殊错误

错误代码	描述	HTTP状态码	SOAP错误代码的前缀
NoSuchUpload	指定的分块上传任务不存在。可能是上传ID无效，也可能是分块上传任务已经完成或放弃。	404 Not Found	Client

示例

请求示例

```
PUT /my-video.rm ?partNumber=2 &uploadId= 1aa9cfad5e2e405c8f27965feb8b60cc HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Mon, 1 Nov 2010 20:34:56 GMT
Content-Length: 10485760
Content-MD5: pUNXr/BjKK5G2UKvaRRrOA==
Authorization: authorization string
```

part data omitted

响应示例

```
HTTP/1.1 200 OK
Date: Mon, 1 Nov 2010 20:34:56 GMT
ETag: "b54357faf0632cce46e942fa68356b38"
Content-Length: 0
Connection: keep-alive
Server: Tengine
```

接口细节分析

- UploadId是在初始化分块上传时获取的
- PartNumber需要是升序连续的数字，且范围在1到10,000之间。否则最后在Complete Multipart Upload时，可能会导致KS3返回错误的状态码。

Complete Multipart Upload

描述

此操作将完成对象装配之前的块上传任务 。

用户启动一个分块上传任务后，会使用 Upload Parts 接口上传所有的块。成功上传所有相关块之后，用户需要调用此接口来完成分块上传。收到完成请求后，KS3将会根据块序号将所有的块组装起来创建一个新的对象。在用户的完成任务请求中需要用户提供分块列表，由于KS3将会按照列表将所有块连接起来，所以要求用户保证所有的块已经完成上传。对于分块列表中的每一个块，用户需要在上传块时添加块序号以及对象的 ETag 头部，KS3则会在块完成上传后回复完成响应。

请注意，如果 Complete Multipart Upload 请求失败了，用户应用应当能够进行重试操作。

请求

语法

```
POST /{ObjectKey}?uploadId={UploadId} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {Date}
Content-Length: {Size}
Authorization: {SignatureValue}

<CompleteMultipartUpload>
<Part>
<PartNumber>{PartNumber}</PartNumber>
<ETag>{ETag}</ETag>
</Part>
...
</CompleteMultipartUpload>
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该请求仅使用常用请求头部。获取更多信息，请点击常用请求头部

请求内容

名称	描述	必须
CompleteMultipartUpload	请求信息容器。 父节点：无 类型：Container 子节点：One or more Part elements	是
Part	某一特定的已上传的块信息容器。 父节点：CompleteMultipartUpload 类型：Container 子节点：PartNumber, ETag	是
PartNumber	用于标识块的块序列号。 父节点：Part 类型：Integer	是
ETag	块上传后返回的实体标签。 父节点：Part 类型：String	是

响应

响应头部

该接口使用常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256

响应内容

名称	描述
CompleteMultipartUploadResult	响应信息的容器。 类型：Container 子节点：Location, Bucket, Key, ETag 父节点：None
Location	用于标识新创建对象的URI。 类型：URI 父节点：CompleteMultipartUploadResult
Bucket	包含新创建对象的空间的名称。 类型：String 父节点：CompleteMultipartUploadResult
Key	新创建对象的 object key。 类型：String 父节点：CompleteMultipartUploadResult
ETag	用于标识Object内容的32位十六进制字符串，不同内容的Object对应着不同的ETag。对于Put Object或Post Object请求创建的Object，ETag值是其内容的MD5值；对于分块上传方式创建的Object，ETag值是每个分块的MD5值进行字符串拼接后再次计算所得到的MD5值。ETag值可以用于检查Object内容是否发生变化。 类型：String 父节点：CompleteMultipartUploadResult

特殊错误

错误代码	描述	HTTP状态码
EntityTooSmall	用户拟上传的块大小小于对象所允许的最小值。除了最后一个块之外，每一个块至少在5MB以上。当文件总大小在5M以内的话，可以允许除了最后一个块之外，每一个块至少在100K以上。	400 Bad Request
InvalidPart	一个或多个指定的块没有找到。块可能没有被上传，也可能因为上传了但实体标签不匹配。	400 Bad Request
InvalidPartOrder	分块列表没有按照升序排列。分块必须按序指定分块序号。	400 Bad Request
NoSuchUpload	指定的分块上传任务不存在。可能是上传ID无效，也可能是分块上传任务已经完成或放弃。	404 Not Found

示例

请求示例

```
POST /my-viedo.rm ?uploadId=1aa9cfad5e2e405c8f27965feb8b60cc HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Mon, 1 Nov 2010 20:34:56 GMT
Content-Length: 391
Authorization: authorization string

<CompleteMultipartUpload>
    <Part>
        <PartNumber>1</PartNumber>
        <ETag>"a54357aff0632cce46d942af68356b38"</ETag>
    </Part>
    <Part>
        <PartNumber>2</PartNumber>
        <ETag>"0c78aef83f66abc1fa1e8477f296d394"</ETag>
    </Part>
    <Part>
        <PartNumber>3</PartNumber>
        <ETag>"acbd18db4cc2f85cedef654fccc4a4d8"</ETag>
    </Part>
</CompleteMultipartUpload>
```

响应示例

```
HTTP/1.1 200 OK
Date: Mon, 1 Nov 2010 20:34:56 GMT
Connection: close
Server: Tengine

<?xml version="1.0" encoding="UTF-8"?>
<CompleteMultipartUploadResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Location>http://ks3-example.ks3-cn-beijing.ksyun.com/my-video.rm</Location>
    <Bucket>ks3-example</Bucket>
    <Key>my-video.rm</Key>
    <ETag>"3858f62230ac3c915f300c664312c11f"</ETag>
</CompleteMultipartUploadResult>
```

接口细节分析

- 除了最后一个块之外，每一个块至少在5MB以上。当文件总大小在5M以内的话，可以允许除了最后一个块之外，每一个块至少在100K以上。
- 用户提交的part列表必须是升序连续的。
- 当执行完该操作后，对应的uploadId将会失效
- 同一个Object可以同时拥有不同的Upload Id，当Complete一个Upload ID后，该Object的其他Upload ID不受影响。
- 提交的part列表中的每个part必须是存在的。否则会报InvalidPartOrder

Abort Multipart Upload

描述

此操作将会放弃一个分块上传任务。当任务被放弃后，使用相应的上传ID则会无法实现任何块的上传。存储中已经上传的块将会被释放。

请求

语法

```
DELETE /{ObjectKey}?uploadId={UploadId} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该请求只使用常用请求头部。获取更多信息，请点击常用请求头部

响应

响应头部

该接口只使用常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不使用响应内容。

特殊错误

错误代码	描述	HTTP状态码	SOAP错误代码的前缀
NoSuchUpload	指定的分块上传任务不存在。可能是上传ID无效，也可能是分块上传任务已经完成或放弃。	404 Not Found	Client

示例

请求示例

```
DELETE /my-video.rm ?uploadId=1aa9cfad5e2e405c8f27965feb8b60cc HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 204 OK
Date: Mon, 1 Nov 2010 20:34:56 GMT
Content-Length: 0
Connection: keep-alive
Server: Tengine
```

接口细节分析

- 对于不再使用且没有执行Complete Multipart Upload的分块上传，建议调用该接口取消分块上传任务。

List Parts

描述

此接口将会列出指定上传任务中所有已上传的块。

使用此接口，必须包含发送启动分块任务请求时KS3返回的 upload ID。 由于默认最大分块数为1000，所以该请求最多返回1000个已上传的块。用户可以指定 max-parts 来限制返回的块的数量。如果用户已上传的块数量超过1000，KS3返回的响应中 IsTruncated 的值将会为 true 并增加一个 NextPartNumberMarker 元素。在一个连续 List Parts 请求中，用户可以设置 part-number-marker 参数为之前一个请求返回响应的 NextPartNumberMarker 的值，完成连续列出分块。

请求

语法

```
GET /{ObjectKey}?uploadId={UploadId} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {Date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

参数	描述	必须
encoding-type	指明请求KS3与KS3响应使用的编码方式。object key 可以包含任意Unicode字符。然而，XML 1.0解析器无法解析某些字符，如ASCII码中的0到10。对于这些不能被解析的字符可以添加到请求中，KS3会在响应中对他们进行编码。 类型：String 默认值：无 有效值：url	否
uploadId	用于标识分块上传任务。 类型：String 默认值：无	是
max-parts	设置响应体中块的最大数量。 类型：String 默认值：1,000	否
part-number-marker	指定应该从哪个块开始列举。只有比设定值大的块才会被列举。 类型：String 默认值：无	否

请求头部

该请求只使用常用请求头部。获取更多信息，请点击常用请求头部

响应

响应头部

该接口使用常用响应头部。获取更多信息，请点击常用响应头部。

除常用响应头部外，还返回storage-class响应头，表示文件的类型。

名称	描述
storage-class	用于说明Bucket文件类型。 标准存储不返回此响应头；低频存储此响应头内容为STANDARD_IA ；归档存储此响应头内容为ARCHIVE

响应内容

名称	描述
ListPartsResult	响应信息的容器。 子节点：Bucket, Key, UploadId, Initiator, Owner, StorageClass,PartNumberMarker, NextPartNumberMarker, MaxParts, IsTruncated, Part 类型：Container
Bucket	启动分块上传任务的空间的名称。 类型：String 父节点：ListPartsResult
Encoding-Type	KS3发送的XML响应中 object key的编码方式。如果用户指定了一种编码方式，那么KS3响应中将会对 object key 使用用户指定的编码方式进行编码。 类型：String 父节点：ListBucketResult
Key	启动分块上传任务的对象的 object key。 类型：String 父节点：ListPartsResult
UploadId	用于标识分块上传任务。 类型：String 父节点：ListPartsResult
Initiator	包含分块上传任务发起人的信息的容器。 子节点：ID, DisplayName 类型：Container 父节点：ListPartsResult
ID	用户ID 类型：String 父节点：Initiator
DisplayName	用户的显示名称。 类型：String 父节点：Initiator
Owner	标识对象拥有者信息的容器。 子节点：ID, DisplayName 类型：Container 父节点：ListPartsResult
StorageClass	上传对象的存储方式。 类型：String 说明：标准存储返回STANDARD；低频存储此响应内容值为STANDARD_IA ；归档存储此响应内容值为ARCHIVE 父节点：ListPartsResult
PartNumberMarker	列举块的开始位置。 类型：Integer 父节点：ListPartsResult
NextPartNumberMarker	当一个列表被截断了，该参数指定最后一个块的序号，可以用来在连续列举块请求中设置 part-number-marker 值。 类型：Integer 父节点：ListPartsResult
MaxParts	响应体中所允许的最大块数。 类型：Integer 父节点：ListPartsResult
IsTruncated	标识列表是否完整。如果上传的块数超过了 MaxParts，那么这个列表应该是被截断的。 类型：Boolean 父节点：ListPartsResult

Part	包含某个指定块的信息的容器。响应中可以包含0个或多个块容器。 子节点：PartNumber，LastModified，ETag，Size 类型：String 父节点：ListPartsResult
PartNumber	标识块的块序列号。 类型：Integer 父节点：Part
LastModified	指定块最后一个完成上传的时间。 类型：Date 父节点：Part
ETag	块上传完成后返回的实体标签。 类型：String 父节点：Part
Size	块的大小。 类型：Integer 父节点：Part

特殊错误

该请求不返回任何特殊错误。

示例

请求示例

```
GET /my-video.rm ?uploadId=1aa9cfad5e2e405c8f27965feb8b60cc &max-parts=2 &part-number-marker=1 HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Mon, 1 Nov 2010 20:34:56 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Mon, 1 Nov 2010 20:34:56 GMT
Content-Length: 985
Connection: keep-alive
Server: Tengine

<?xml version="1.0" encoding="UTF-8"?>
<ListPartsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Bucket>ks3-example</Bucket>
    <Key>my-video.rm</Key>
    <UploadId>1aa9cfad5e2e405c8f27965feb8b60cc</UploadId>
    <Initiator>
        <ID>73410125</ID>
        <DisplayName>ks3@kingsoft.com</DisplayName>
    </Initiator>
    <Owner>
        <ID>73410125</ID>
        <DisplayName>ks3@kingsoft.com</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <PartNumberMarker>1</PartNumberMarker>
    <NextPartNumberMarker>3</NextPartNumberMarker>
    <MaxParts>2</MaxParts>
    <IsTruncated>true</IsTruncated>
    <Part>
        <PartNumber>2</PartNumber>
        <LastModified>2010-11-10T20:48:34.000Z</LastModified>
        <ETag>7778aef83f66abc1fa1e8477f296d394</ETag>
        <Size>10485760</Size>
    </Part>
    <Part>
        <PartNumber>3</PartNumber>
        <LastModified>2010-11-10T20:48:33.000Z</LastModified>
        <ETag>aaaa18db4cc2f85cedef654fccc4a4x8</ETag>
        <Size>10485760</Size>
    </Part>
</ListPartsResult>
```

Upload Part Copy

描述

此操作将可以通过拷贝已存在的对象的方式实现上传一个块。用户通过在请求中配置请求头部 x-kss-copy-source 来指定数据源，通过请求头部 x-kss-copy-source-range 来指定字节范围。

注意：此外你还可以通过在请求中包含数据的方式来实现块的上传，更多信息，请查看接口 Upload Part

在你上传任一块之前你必须先要启动一个分块上传任务。在你发送一个启动请求后，KS3会给你一个唯一的 upload ID。每次上传块时，都需要将上传ID包含在请求中。

块的数量可以是1到10,000中的任意一个（包含1和10,000）。块序号用于标识一个块以及其在对象创建时的位置。如果你上传一个新的块，使用之前已经使用的序列号，那么之前的那个块将会被覆盖。除了最后一个块没有大小限制外，其余的块的大小均要求在5MB以上。

请求

语法

```
PUT /{ObjectName}?partNumber={PartNumber}&uploadId={UploadId} HTTP/1.1
Host: {BucketName}.{endpoint}
x-kss-copy-source: {/source_bucket/sourceObject}
x-kss-copy-source-range:bytes={first}-{last}
x-kss-copy-source-if-match: {etag}
x-kss-copy-source-if-none-match: {etag}
x-kss-copy-source-if-unmodified-since: {time_stamp}
x-kss-copy-source-if-modified-since: {time_stamp}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必需
x-kss-copy-source	空间名称与对象的object key名称的组合，通过斜杠分隔('/')。 类型：String 默认值：无	是

x-kss-copy-source-range	从数据源拷贝的字节范围。范围设定必需使用 bytes=first-last，其值是基于0的。例如：bytes=0-9 表示用户将复制前10个字节。当拷贝整个对象时，不需要此头部。 类型：Integer 默认值：无	否

下表所列头部基于指定的 x-kss-copy-source 头部

名称	描述	必需
x-kss-copy-source-if-match	如果 object 的 ETag(entity tag)与指定值一致，则拷贝 object。否则，返回412状态码 类型：String 默认值：None 约束条件：无	否
x-kss-copy-source-if-none-match	如果 object 的 ETag(entity tag)与指定值不一致，则拷贝 object。否则，返回304状态码 类型：String 默认值：None 约束条件：无	否
x-kss-copy-source-if-unmodified-since	如果 object 在指定时间后没有被改变，则拷贝 object。否则，返回412状态码 类型：String 默认值：None 约束条件：无	否
x-kss-copy-source-if-modified-since	如果 object 在指定时间后被改变，则拷贝 object。否则，返回304状态码 类型：String 默认值：None 约束条件：无	否

若要求服务端加密则需要以下头部

名称	描述	必需
x-kss-server-side-encryption-customer-key	由用户指定KS3加密时使用的 base64-encoded 加密秘钥，其值必须与启动分块上传任务时的秘钥一致。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-algorithm,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-algorithm 同时使用	是

如果作为数据源的对象使用用户秘钥进行服务端加密，用户需要使用下列头部使KS3能够对对象解密。

名称	描述	必需
x-kss-copy-source-server-side-encryption-customer-key	由用户指定KS3解密时使用的 base64-encoded 加密秘钥，其值必须数据源对象创建时使用的秘钥一致。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-algorithm,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-copy-source-server-side-encryption-customer-algorithm	指定数据源对象解密使用的解密算法。 类型：String 有效值：AES256 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-key-MD5 同时使用	是
x-kss-copy-source-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String 约束：需要和有效的 x-kss-server-side-encryption-customer-key,x-kss-server-side-encryption-customer-algorithm 同时使用	是

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则响应会包含该头部，值为使用的加密算法。 类型：String
x-kss-server-side-encryption-customer-algorithm	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来确认使用的解密算法。 类型：String 有效值：AES256
x-kss-server-side-encryption-customer-key-MD5	如果服务端使用了用户提供的加密秘钥加密，在请求解密时，响应将会包含该头部来提供用户提供加密秘钥的数据一致性验证信息。 类型：String

响应内容

名称	描述
CopyPartResult	响应内容的容器。 类型：Container 父节点：无
ETag	返回一个新块的实体标签。 类型：String 父节点：CopyPartResult
LastModified	返回最后被修改的时间日期。 类型：String 父节点：CopyPartResult

特殊错误

错误代码	描述	HTTP状态码
NoSuchUpload	指定的分块上传任务不存在。可能是上传ID无效，也可能是分块上传任务已经完成或放弃。	404 Not Found
InvalidRequest	指定数据源对象不支持按字节范围进行拷贝。	400 Bad Request

示例

请求示例

```
PUT /new-video.rm?partNumber=2 &uploadId=6aaf37c7501847569c91f9957b01fd14 HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Mon, 11 Apr 2011 20:34:56 GMT
x-kss-copy-source: /source-bucket/sourceobject
x-kss-copy-source-range:bytes=500-6291456
```

```
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Mon, 11 Apr 2011 20:34:56 GMT
Server: Tengine

<CopyPartResult>
<LastModified>2009-10-28T22:32:00</LastModified>
<ETag>"9b2cf535f27731c974343645a3985328"</ETag>
</CopyPartResult>
```

POST Policy

Policy 构建方法

描述

Policy是使用 UTF-8 和 Base64 编码的 JSON 文档，它指定了请求必须满足的条件并且用于对内容进行身份验证。根据您设计Policy文档的方式，您可以对每次上传、每个用户、所有上传或根据其他能够满足您需要的设计来使用它们。 Policy文档示例

```
{
    "expiration": "2015-01-01T12:00:00.000Z",
     "conditions": [
        ["eq","$acl", "public-read" ],
        ["eq","$bucket", "mybucket" ],
        ["starts-with", "$key", "2015/01/"]
    ]
}
```

Expiration

expiration元素采用 ISO 8601 UTC 日期格式来指定策略的过期日期，即只能在指定时间之前进行上传，在这个时间之后的上传都将被认为是非法的，KS3将返回403。注意：时区为格林时间(零时区)而非北京时间(东八区)

Conditions

Policy文档中对上传内容的验证条件。您在表单中指定的每个表单字段（KSSAccessKeyId、Signature、file、policy除外）以及bucket必须包含在条件列表中。
注意：表单中的所有变量会在验证Policy之前进行扩展。因此，应该针对扩展的字段执行所有条件匹配。例如，如果您将key字段设置为 2015/01/${filename}，您Policy中的condition可能是 ["starts-with", "$key", "2015/01/"]。请勿输入 ["starts-with", "$key", "2015/01/${filename}"]

元素名称	描述
bucket	指定bucket必须满足的条件，支持精确匹配和starts-with
acl	指定acl必须满足的条件，支持精确匹配和starts-with
content-length-range	指定上传内容(不止是文件，还有其他表单项)Content-Length的最小和最大值,支持范围匹配
Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires	REST 特别头部。支持精确匹配和starts-with
key	上传的key,支持精确匹配和starts-with
success_action_redirect, redirect	成功上传之后重定向的路径。支持精确匹配和starts-with
success_action_status	成功上传之后返回的状态码(没有指定success_action_redirect, redirect的情况下)，仅支持精确匹配
其他以x-kss-meta开头的表单项	用户元数据.支持精确匹配和starts-with
其他KSSAccessKeyId、Signature、file、policy外的表单项	KS3服务器不会对这些表单项做处理。但是必须在Policy文档中包含。支持精确匹配和starts-with

Condition匹配规则

匹配规则	示例
精确匹配	指定acl必须等于public-read。{"acl": "public-read" }或者["eq", "$acl", "public-read"]
Starts with	指定key必须以2015/01/开头。["starts-with", "$key", "2015/01/"]
通配	指定success_action_redirect可以是任意值（注意:success_action_status不支持通配）。["starts-with", "$success_action_redirect", ""]
指定Content-Length范围	指定上传内容(不止是文件，还有其他表单项)Content-Length的范围为1048579-10485760。["content-length-range", 1048579, 10485760]

字符转义

下表中的字符必须在Policy文档中做转化

转义字符	描述
\\	反斜杠
\$	美元符号
\b	退格键
\f	换页
\n	新建行
\r	回车
\t	水平选项卡
\v	垂直选项卡
\uxxxx	Unicode 字符

Signature 构建方法

对于验证的 Post 请求，HTML 表单中必须包含 Policy 和 Signature 信息。
Policy 控制请求中哪些值是允许的。
计算 Signature 的具体流程为：

1. 创建一个 UTF-8 编码的 Policy文档，详见Policy构建方法。
2. 将 Policy 进行 base64 编码，其值即为 Policy 表单域该填入的值，将该值作为将要签名的字符串(StringToSign)。
3. 签名Signature = Base64(HMAC-SHA1(YourSecretKey, UTF-8-Encoding-Of(StringToSign)));

PUT Fetch

描述

此PUT接口从第三方URL拉取文件，并上传至KS3某个 bucket 中存储成名为 object 的文件。用户通过在请求中配置请求头部 x-kss-sourceurl 来指定第三方URL。也可以通过配置请求头 x-kss-callbackurl 来指定上传成功或失败时的回调URL。还可以通过 x-kss-acl 来设置上传KS3后的文件权限。

注意：

该接口执行过程分为两个部分：

1.发起PUT Object Fetch请求；

2.执行拉取文件并上传到KS3。这两个过程异步执行，先发起PUT Object Fetch请求，成功后再执行Fetch拉取文件和上传到KS3的过程。两个过程有不同的返回信息，其中，发起PUT Object Fetch请求完成会返回http状态码200；如果用户配置了回调url，则根据不同的回调状态返回不同的状态码。

如果fetch的文件和KS3已存在的文件同名，KS3服务端会校验请求头中的Content-Md5值和KS3同名文件的md5值。如果不相同，才会拉取文件进行覆盖。

KS3会尝试访问 x-kss-sourceurl 指定的URL，会重试三次。如果拉取文件成功，会尝试往 bucket 上传名为 object 的文件，会重试三次。

如果使用回调接口，KS3在拉取成功或失败时，会通过POST方法向用户的回调地址POST一段json数据。用户服务端正确处理回调后通过响应HTTP状态码200表示正确处理，对于其它状态码，KS3认为客户服务端处理异常，会进行重试。超时时间设置为3秒，重试三次。

使用此接口，用户需要具有目标空间的写权限。并且所有的拉取请求均需要用户进行身份验证且不包含信息体。

请求

语法

```
PUT /{ObjectKey}?fetch HTTP/1.1
Host: {BucketName}.{endpoint}
x-kss-sourceurl: {sourceurl}
x-kss-callbackurl: {callbackurl}
x-kss-acl: {acl}
x-kss-tagging:TagA=A&TagB=B
Authorization: {SignatureValue}
Date: {date}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部，此外，也可以使用下表所列请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必需
x-kss-sourceurl	第三方URL地址，需要进行URLEncode。	是
x-kss-callbackurl	拉取成功或失败的回调URL，需要进行URLEncode。	否
x-kss-acl	用于对象的预定义权限。 类型：String 默认值：private 有效值：private | public-read	否
x-kss-tagging	指定目标Object对象标签，可同时设置多个标签，　如：TagA=A&TagB=B。 **说明 Key和Value需要先进　URL编码，** 如果某项没有“=”，则看作Value为空字符　。详见对象标签。	否
Content-MD5	base64加密MD5信息，128位，用于对象完整性校验。 类型：String 默认值：None 如果这个值与上传到KS3文件的MD5不一致，回调会返回上传KS3失败（status = 3）。	否

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不返回响应内容。

特殊错误

该请求不返回任何特殊错误。

回调

回调内容

如果在请求中配置了回调url（x-kss-callbackurl），则在拉取、上传过程后会执行回调返回拉取、上传的执行结果。回调内容是一段JSON。顶层包括 status 字段，用来表示拉取与上传的状态。

如果 status 为 0，表示拉取成功、上传KS3成功。同时会包括一些文件信息。

如果 status 为 1，表示拉取失败，需要检查指定的第三方URL。

如果 status 为 2，表示上传KS3失败，请稍后重试。

如果 status 为 3，表示上传的md5值与从源站拉取下来的文件md5值不一致。

如果 status 为 4，表示文件已经存在。

如果 status 为 5，表示相同的objectKey的仍在执行中。

回调内容还会包含请求接口时返回的requestId，用于建立请求的对应关系。

文件信息

参数	说明	备注
bucket	文件上传的Bucket	Utf-8编码
key	文件的名称	Utf-8编码
objectSize	文件大小	以字节标识
sourceUrl	拉取资源URL	Utf-8编码

示例

请求示例

```
PUT /my-image.jpg?fetch HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
x-kss-sourceurl: http://example.com/some-resource-url
x-kss-callbackurl: http://example.com/some-callbackurl
x-kss-acl: public-read
x-kss-tagging:TagA=A&TagB=B
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2009 22:32:00 GMT
Connection: close
Server: Tengine
```

回调示例

拉取上传成功后的回调内容：

```
{"status":0,"key":"test-object-key","bucket":"test-bucket-name","objectSize":20597555,"sourceUrl":"http://example.com/some-resource-url","requestId":"d089cd7e583a4f9d95626de6695279db"}
```

拉取失败时的回调内容：

```
{"status":1,"key":"test-object-key","bucket":"test-bucket-name","sourceUrl":"http://example.com/some-resource-url","requestId":"47661b13ce8c4c4aabbcda3a1a2dfe14"}
```

拉取成功，但上传失败时的回调内容：

```
{"status":2,"key":"test-object-key","bucket":"test-bucket-name","sourceUrl":"http://example.com/some-resource-url","requestId":"f28d01ef77274c0c8f40489a41f367f4"}
```

接口细节分析

- 此接口返回 200 仅代表发起PUT Object Fetch请求成功。不代表文件已经上传到KS3，甚至不保证文件会上传至KS3。如果需要了解拉取是否成功，需要注册回调。
- 用户必须对目标Bucket具有写权限。
- 仅支持对存储到KS3的目标文件设置tag，且需要具有ks3:PutObjectTagging操作授权。
- 如果目标key已经存在，则此接口会报错。
- 请不要多次对同一个目标文件发起fetch请求。如果文件正在fetch中，再次发起请求不会有任何回调。
- 当访问者具有ks3:PutObject权限，但没有PutObjectTagging权限时，该接口仅允许上传不带tagging的对象。

Infrequent Access

接口描述

KS3标准低频存储用于保存不频繁访问但在需要时也要求快速访问的数据。

用户可以在上传文件过程中，通过HTTP头指定存储类型；相应的，在下载文件过程中我们会通过HTTP响应头指示文件的存储类型。

除此之外，在用户枚举文件时也会返回文件的存储类型信息。

上传接口

Put Object 上传文件

用户上传过程中，可以通过设置 x-kss-storage-class 头来指定存储类型。 有效值为： STANDARD 或 STANDARD_IA。 对于无效的存储类型，KS3会拒绝本次请求。

请求语法：

```
PUT /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

SDK示例代码：

```
PutObjectRequest request = new PutObjectRequest(bucketName, key, file);
request.setStorageClass(StorageClass.StandardInfrequentAccess);
PutObjectResult result = client.putObject(request); // 假设已经合理初始化Ks3Client
```

Post Object 上传文件

用户可以在表单域中通过设置 x-kss-storage-class 域来指定存储类型。 有效值为： STANDARD 或 STANDARD_IA。 对于无效的存储类型，KS3会拒绝本次请求。

Java SDK不支持POST方式上传文件。

Initiate Multipart Upload 初始化分块上传

用户可以在初始化分块上传时通过 x-kss-storage-class 请求头设置文件的存储类型。 有效值为： STANDARD 或 STANDARD_IA。 对于无效的存储类型，KS3会拒绝本次请求。

请求语法：

```
POST /{ObjectKey}?uploads HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

SDK示例代码：

```
InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest(bucketName, key);
request.setStorageClass(StorageClass.StandardInfrequentAccess);
InitiateMultipartUploadResult result = client.initiateMultipartUpload(request); // 假设已经合理初始化Ks3Client
```

后续的所有分块上传操作(Upload Part、Complete、Abort)都不再接受 x-kss-storage-class 请求头。

Put Object Copy 复制文件

对于复制文件，KS3目前不支持重新指定存储类型。目的文件的存储类型必须与源文件存储类型一致。 如果复制源文件是标准低频存储，则必须设置 x-kss-storage-class 请求头，其值必须为 STANDARD_IA。 如果源文件存储类型与目的文件存储类型不一致，会报错。提示 InvalidStorageClass。

Upload Part Copy 复制分块

对于复制分块，KS3目前不支持重新指定存储类型。目的文件的存储类型必须与源文件存储类型一致。 对于复制分块，KS3会忽略 x-kss-storage-class 请求头。 如果源文件存储类型与目的文件存储类型不一致，会报错。提示 InvalidStorageClass。

下载接口

Get Object 下载文件

用户下载文件时，如果文件是低频存储文件，则在响应头中会出现 x-kss-storage-class 头；标准存储文件下载时没有这个响应头。

Java SDK中，可以通过获取元信息来查看存储类型，示例代码如下：

```
GetObjectRequest request = new GetObjectRequest(bucketName, key);
GetObjectResult result = client.getObject(request);
String storageClass = result.getObject().getObjectMetadata().getStorageClass();
```

对于标准存储文件，getStorageClass 返回 null；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

枚举接口

Get Bucket 枚举bucket下的所有文件

用户获取某个Bucket下文件时，KS3会为每个文件返回其存储类型。

SDK示例代码：

```
ListObjectsRequest request = new ListObjectsRequest(bucketName);
ObjectListing listing = client.listObjects(request);
for (Ks3ObjectSummary summary : listing.getObjectSummaries()) {
    String storageClas = summary.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Multipart Uploads 查看bucket下的分块上传

用户在查看当前bucket下所有的分块上传请求时，KS3会为每个分块上传返回其存储类型。

SDK示例代码：

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest(bucketName);
ListMultipartUploadsResult result = client.listMultipartUploads(request);
for (MultiPartUploadInfo uploadInfo : result.getUploads()) {
    String storageClass = uploadInfo.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Parts 查看已上传的块

用户在列举某个分块上传已经上传的块时，KS3会返回当前这次分块上传的存储类型。

SDK示例代码：

```
ListPartsRequest reqest = new ListPartsRequest(bucketName, key, result.getUploadId());
ListPartsResult result = client.listParts(reqest);
String storageClass = result.getStorageClass();
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

Restore Object

描述

此接口用于对归档Object进行解冻。

此接口只针对归档类型的Object的解冻操作。如果一个Object是标准或者低频访问类型，不要调用该接口。

归档类型Object状态说明

1. 将文件上传到归档存储Bucket，文件自动存储为归档object,处于冷冻状态。
2. 提交一次Restore操作后，Object将处于解冻中的状态，服务端执行解冻。解冻任务完成需要1到10分钟。
3. 待服务端执行完成解冻任务后，Object进入解冻状态，此时用户可以读取Object。
4. 解冻状态默认持续24小时，24小时内再次调用Restore Object接口则解冻状态会自动延长24小时，最多可延长7天，之后，Object又回到初始时的冷冻状态。

解冻操作所产生的费用说明：

- 对一个处于冷冻状态的Object执行Restore操作，会产生数据取回费用。
- 解冻状态最多延长7天。在此期间内不再重复收取数据取回费用。
- 解冻状态结束后，Object又回到冷冻状态，再次解冻的首次读取数据会收取数据取回费用。
- 每调用Restore Object接口，都会产生请求数费用

请求

语法

下面展示的是对某一个归档文件进行解冻请求

```
POST /{ObjectKey}?restore HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

语法

该接口不使用请求参数。

请求头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

说明

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

示例

说明

- 如果是对归档Object第一次调用RestoreObject接口，则返回202。
- 如果已经成功调用过RestoreObject接口，且Object仍处于解冻中，再次调用时返回409，错误码为：RestoreAlreadyInProgress，表示服务端正在执行解冻操作，用户只需要等待解冻完成，最长等待时间10分钟。
- 如果已经成功调用过RestoreObject接口，且服务端解冻已经完成，再次调用时返回200，且会将归档文件的可下载时间延长一天，最多延长7天。
- 如果文件不存在，则返回404。
- 如果针对非归档类型的文件提交解冻操作，则返回400，错误码为：OperationNotSupported。

首次提交请求示例

```
POST /restore_test.txt?restore HTTP/1.1
Host: test-arc-l2/ks3-cn-shanghai-internal.ksyun.com
Date: Fri, 26 Dec 2018 06:34:32 GMT
Authorization:KSS authorization string
```

响应示例

```
HTTP/1.1 202 Accepted
Server: Tengine
Date: Fri, 26 Dec 2018 06:34:32 GMT
Content-Length: 0
Connection: keep-alive
X-Application-Context: application
x-kss-request-id: cb2a95d037794c0293be0a094375c047
x-kss-storage-class: ARCHIVE
```

再解冻操作未完成的情况下，再次提交请求示例

```
POST /restore_test.txt?restore HTTP/1.1
Host: test-arc-l2/ks3-cn-shanghai-internal.ksyun.com
Date: Fri, 26 Dec 2018 06:35:02 GMT
Authorization:KSS authorization string
```

响应示例

```
HTTP/1.1 409 Conflict
Server: Tengine
Date: Fri, 26 Dec 2018 06:35:02 GMT
Content-Type: application/xml
Transfer-Encoding: chunked
Connection: keep-alive
X-Application-Context: application
x-kss-request-id: 79516a04224b423181f20f3dcbf2f416
x-kss-StorageClass: -
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Error>
    <Code>RestoreAlreadyInProgress</Code>
    <Message>Object restore is already in progress.</Message>
    <Resource>/test-arc-l2/restore_test1.txt?restore</Resource>
    <RequestId>79516a04224b423181f20f3dcbf2f416</RequestId>
</Error>
```

解冻后，第三次提交请求示例

```
POST /restore_test.txt?restore HTTP/1.1
Host: test-arc-l2/ks3-cn-shanghai-internal.ksyun.com
Date: Fri, 26 Dec 2018 06:40:02 GMT
Authorization:KSS authorization string
```

响应示例

```
HTTP/1.1 200 OK
Server: Tengine
Date: Fri, 26 Dec 2018 06:40:02 GMT
Content-Length: 0
Connection: keep-alive
X-Application-Context: application
x-kss-request-id: ld147c34a9d542aa8dc7f0f70d3db893
x-kss-storage-class: ARCHIVE
```

ARCHIVE

接口描述

KS3归档存储（ACHIVE)适合需要长期保存（建议3个月以上）的归档数据，在存储周期内很少被访问，数据进入到可读取状态需要1分钟到10分钟的解冻时间。适合需要长期保存的档案数据、医疗影像、科学资料、影视素材。

您想采取归档存储类型来存储文件，您可以通过以下方式：

1. 利用上传接口，指定请求头x-kss-storage-class为ARCHIVE
2. 创建一个归档类型Bucket，无需指定存储类型，即可按照默认归档类型存储。
3. 通过生命周期转化，设定存储类型转化规则，按照规则系统自动到期转化为归档存储类型

上传接口

Put Object 上传归档文件

上传文件时，如果不指定请求头x-kss-storage-class，如果Bucket是归档类型，Object自动为归档类型，如果Bucket是非归档类型，Object自动为标准类型；如果指定Object 的x-kss-storageClass则以用户指定为准，例如在非归档存储空间中，指定Object 的x-kss-storageClass为ARCHIVE时，则该文件为归档存储类型。

bucket类型	x-kss-storage-class请求头	文件类型
归档存储类型	不上传	归档类型
归档存储类型	ARCHIVE	归档类型
归档存储类型	STANDAND	标准类型
归档存储类型	STANDARD_IA	低频类型
非归档存储类型	不上传	标准类型
非归档存储类型	STANDAND	标准类型
非归档存储类型	STANDARD_IA	低频类型
非归档存储类型	ARCHIVE	归档类型

请求语法：

```
PUT /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

Post Object 上传文件

用户可以在表单域中通过设置 x-kss-storage-class 域来指定存储类型。规则和Put Object上传文件一致。

Initiate Multipart Upload 初始化分块上传

用户可以在初始化分块上传时通过 x-kss-storage-class 请求头设置文件的存储类型。 有效值为： STANDARD 、STANDARD_IA 、ARCHIVE。规则和Put Object上传文件一致。 对于无效的存储类型，KS3会拒绝本次请求。

请求语法：

```
POST /{ObjectKey}?uploads HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

后续的所有分块上传操作(Upload Part、Complete、Abort)都不再接受 x-kss-storage-class 请求头。

Put Object Copy 复制文件

对于复制文件，KS3支持重新指定存储类型。 如果复制源文件是归档存储文件，成功复制的前提是，源文件必须完成解冻，为可读状态。

Upload Part Copy 复制分块

对于复制分块，KS3目前不支持重新指定存储类型。目的文件的存储类型必须与源文件存储类型一致。 对于复制分块，KS3会忽略 x-kss-storage-class 请求头。 如果源文件存储类型与目的文件存储类型不一致，会报错。提示 InvalidStorageClass。

解冻接口

详解解冻接口

下载接口

说明 归档文件必须完成解冻，才能正常下载。

Get Object 下载文件

```
GetObjectRequest request = new GetObjectRequest(bucketName, key);
GetObjectResult result = client.getObject(request);
Ks3Object object = result.getObject();
object.getObjectContent();
```

Head Object 查看文件元信息

Java SDK中，可以通过获取元信息来查看存储类型，示例代码如下：

```
HeadObjectRequest request = new HeadObjectRequest(bucketName, key);
HeadObjectResult result =client.headObject(request);
String storageClass = result.getObjectMetadata().getStorageClass();
```

对于标准存储文件，getStorageClass 返回 null；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

枚举接口

Get Bucket 枚举bucket下的所有文件

用户获取某个Bucket下文件时，KS3会为每个文件返回其存储类型。

SDK示例代码：

```
ListObjectsRequest request = new ListObjectsRequest(bucketName);
ObjectListing listing = client.listObjects(request);
for (Ks3ObjectSummary summary : listing.getObjectSummaries()) {
    String storageClas = summary.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Multipart Uploads 查看bucket下的分块上传

用户在查看当前bucket下所有的分块上传请求时，KS3会为每个分块上传返回其存储类型。

SDK示例代码：

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest(bucketName);
ListMultipartUploadsResult result = client.listMultipartUploads(request);
for (MultiPartUploadInfo uploadInfo : result.getUploads()) {
    String storageClass = uploadInfo.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Parts 查看已上传的块

用户在列举某个分块上传已经上传的块时，KS3会返回当前这次分块上传的存储类型。

SDK示例代码：

```
ListPartsRequest reqest = new ListPartsRequest(bucketName, key, result.getUploadId());
ListPartsResult result = client.listParts(reqest);
String storageClass = result.getStorageClass();
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

Delete Object Tagging

DeleteObjectTagging接口用于删除指定对象（Object）的所有标签（Tagging）信息。

权限

该接口操作需要用户对Object拥有DeleteObjectTagging权限。KS3将根据你是否拥有写权限返回相应信息。 如果你并不拥有该Object的DeleteObjectTagging权限，KS3将会返回的 HTTP 状态码为403(access denied)错误。

请求语法

```
DELETE /<ObjectKey>?tagging
Host: <BucketName>.<Region>.ksyun.com
Date: GMT Date
Authorization: SignatureValue
```

请求头

此接口仅涉及公共请求头

响应头

此接口仅涉及公共响应头。

响应体

无

特殊错误

不返回任何特殊错误。

示例

请求示例

```
DELETE /<ObjectKey>?tagging HTTP 1.1
Host:<BucketName>.<Region>.ksyun.com
Date: GMT Date
Authorization: Auth String
```

返回示例

```
204 (No Content)
content-length: 0
server:ks3
x-kss-request-id: 5CAC0AD16D0232E2051B****
date: Tue, 09 Apr 2020 03:00:33 GMT
```

Get Object Tagging

可通过GetObjectTagging接口获取对象（Object）的标签（Tagging）信息。

权限

该接口操作需要用户对Object拥有GetObjectTagging权限。KS3将根据你是否拥有写权限返回相应信息。 如果你并不拥有该Object的GetObjectTagging权限，KS3将会返回的 HTTP 状态码为403(access denied)错误。

请求语法

```
GET /<ObjectKey>?tagging HTTP 1.1
Host:<BucketName>.<Region>.ksyun.com
Date: GMT Date
Authorization: Auth String
```

请求头

此接口仅涉及公共请求头

响应头

此接口仅涉及公共响应头。

响应元素

名称	类型	描述
Tagging	容器	标签集合。 子节点：TagSet
TagSet	容器	标签集合。 父节点：Tagging 子节点：Tag
Tag	容器	标签集合。 父节点：TagSet 子节点：Key、Value
Key	字符串	标签键。 父节点：Tag 子节点：无
Value	字符串	标签值。 父节点：Tag 子节点：无

特殊错误

不返回任何特殊错误。

示例

- 请求示例

```
GET /<ObjectKey>?tagging HTTP 1.1
Host:<BucketName>.<Region>.ksyun.com
Date: GMT Date
Authorization: Auth String
```

返回示例

```
HTTP/1.1 200 OK
Content-Type: application/xml
Connection: close
Date: Fri, 19 Jan 2020 11:40:22 GMT
Server: ks3
<Tagging>
    <TagSet>
        <Tag>
            <Key>age</Key>
            <Value>18</Value>
        </Tag>
        <Tag>
            <Key>name</Key>
            <Value>xiaoming</Value>
        </Tag>
    </TagSet>
</Tagging>
```

Put Object Tagging

PutObjectTagging接口用于设置或更新对象（Object）的标签（Tagging）信息。

注意事项

对象标签使用一组键值对（Key-Value）标记对象。调用PutObjectTagging接口时，有如下注意事项：

- 单个Object最多可设置10个标签，Key不可重复。

  Key命名规则：

  - 支持大小写字母、数字、空格和符号 + - = . _ : /
  - 1-128字节，区分大小写，不能以空格开头或结尾，不容许为空
  - 不容许设置系统保留字段，ksc:与kss:开头

  value设置规则：

  - 支持大小写字母、数字、空格和符号 + - = . _ : /
  - 1-256字节，区分大小写，不能以空格开头或结尾

- 更改标签信息不会更新Object的Last Modified时间。

  通过HTTP header的方式设置标签且标签中包含任意字符时，您需要对标签的Key和Value做URL编码。

有关对象标签的更多信息，请参见对象标签。

权限

该接口操作需要用户对Object拥有PutObjectTagging权限。KS3将根据你是否拥有写权限返回相应信息。 如果你并不拥有该Object的PutObjectTagging权限，KS3将会返回的 HTTP 状态码为403(access denied)错误。

请求语法

```
PUT /<ObjectKey>?tagging HTTP 1.1
Host:<BucketName>.<Region>.ksyun.com
Date: GMT Date
Authorization: Auth String
<Tagging>
    <TagSet>
        <Tag>
            <Key>string</Key>
            <Value>string</Value>
        </Tag>
    </TagSet>
</Tagging>
```

请求元素

名称	类型	是否必需	描述
Tagging	容器	是	标签集合。 子节点：TagSet
TagSet	容器	是	标签集合。 父节点：Tagging 子节点：Tag
Tag	容器	是	标签集合。 父节点：TagSet 子节点：Key、Value
Key	字符串	是	标签键。长度不超过128字节，支持大小写英文字母、数字、空格、加号、减号、下划线、等号、点号、冒号、正斜线、反斜线。 父节点：Tag 子节点：无
Value	字符串	是	标签值。长度不超过256字节，支持大小写英文字母、数字、空格、加号、减号、下划线、等号、点号、冒号、正斜线、反斜线 父节点：Tag 子节点：无

示例

- 请求示例

  针对存储空间BucketName中的对象ObjectKey，通过PUT请求设置{name:1}和{age:2}两个标签。标签设置成功后返回200 (OK)。

  ```
  PUT /<ObjectKey>?tagging HTTP 1.1
  Host:<BucketName>.<Region>.ksyun.com
  Date: GMT Date
  Authorization: Auth String
  <Tagging>
  <TagSet>
    <Tag>
      <Key>name</Key>
      <Value>1</Value>
    </Tag>
    <Tag>
      <Key>age</Key>
      <Value>2</Value>
    </Tag>
  </TagSet>
  </Tagging>
  ```

  返回示例

  ```
  HTTP/1.1 200 OK
  Date: Fri, 26 Dec 2020 06:34:32 GMT
  x-kss-request-id: dbea4ce4ec23415b9e454ecfa25e****
  Content-Length: 0
  Server: Tengine
  ```

错误码说明

错误码	描述	HTTP状态码
SignatureDoesNotMatch	提供的签名不符合规则，返回该错误码	403 Forbidden
TNoSuchObject	如果对象不存在，则无法添加对象标签，将返回该错误码	404 Not Found
MalformedXML	XML 格式不合法，请跟 Restful API 文档仔细比对	400 Bad Request
InvalidTag	Tag 的 key 和 value 中包含了保留字符串ksc:与kss:	400 InvalidTaggingFormat
BadRequest	超过了一个对象允许设置标签数量的最大值，目前最多支持10个标签	400 Bad Reques

分块上传

接口描述

当文件大于100MB的时候，可以选择分块上传。把大文件进行切割上传到服务器。 分块上传分为三步：

1. Initiate Multipart Upload 初始化分块上传
2. Upload Part 上传文件块
3. Complete Multipart Upload 完成分块上传

上传中，你可以使用Abort Multipart Upload取消上传，或者List Parts查看上传的分块。或者List Multipart Uploads查看当前的bucket下有多少个uploadid。

注意：KS3最多支持10000个文件块

接口包括

- Initiate Multipart Upload 初始化分块上传
- Upload Part 上传文件块
- Complete Multipart Upload 完成分块上传
- Abort Multipart Upload 取消分块上传
- List Parts 查看已上传的块
- List Multipart Uploads 查看bucket下的分块上传

GET Bucket CORS

描述

返回用户 bucket（空间）的 cors （跨源资源共享）信息。

使用此接口，你需要拥有执行 GetBucketCORS 操作的权限。空间拥有者默认具有此权限，并且可以授予他人相应权限。

请求

语法

```
GET /?cors HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口仅使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口仅使用常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

名称	描述	必须
CORSConfiguration	包含 CORSRules 元素的容器，元素上限为100。 类型: Container 子节点: CORSRules 父节点: 无	是
CORSRule	源与方法的集合，最多可以配置100条规则。 类型: Container 子节点: AllowedOrigin, AllowedMethod, MaxAgeSeconds, ExposeHeader, ID. 父节点: CORSConfiguration	是
ID	规则的唯一表示，最多255字符，它可以帮助你快速查找配置中的某一规则。 类型: String 父节点: CORSRule	是
AllowedMethod	用户允许源所能执行的 HTTP 方法，每一条 CORSRule 必须定义至少一个源地址和一种方法。 类型: Enum (GET, PUT, HEAD, POST, DELETE) 父节点: CORSRule	是
AllowedOrigin	用户允许跨源资源共享访问的源地址，其最多含有一个"*"通配符。每一条 CORSRule 必须定义至少一个源地址和一种方法。例如：http://*.example.com。另外，你可以使用"*"来代表全部源。 类型: String 父节点: CORSRule	是
AllowedHeader	指明在预检OPTION中通过 Access-Control-Request-Headers 哪些头部是可以使用的。每一个在 Access-Control-Request-Headers 中指定的头部必须要与发送到KS3请求的头部保持一致，最多使用一个"*" 类型: String 父节点: CORSRule	是
MaxAgeSeconds	指定在 KS3 针对特定资源的预检 OPTIONS 请求作出响应后，浏览器缓存该响应的时间。一个 CORSRule 最多有一个 MaxAgeSeconds 元素. 类型: Integer (seconds) 父节点: CORSRule	是

ExposeHeader	识别可允许客户从应用程序（例如，从 JavaScript XMLHttpRequest 数据元）进行访问的响应标头。 类型：String 父节点：CORSRule	是

特殊错误

该接口不返回错误代码。

示例

请求示例

```
GET /?cors HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 24 Dec 2014 03:08:04 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
Date: Wed, 24 Dec 2014 03:08:04 GMT
Server: Tengine
Content-Type: application/xml
x-kss-request-id: d72a2c2be3ec42aebf5b8c395b6cb8e7

<CORSConfiguration>
<CORSRule>
<AllowedOrigin>http://www.example.com</AllowedOrigin>
<AllowedMethod>GET</AllowedMethod>
<MaxAgeSeconds>3000</MaxAgeSec>
<ExposeHeader>x-kss-server-side-encryption</ExposeHeader>
</CORSRule>
</CORSConfiguration>
```

PUT Bucket CORS

描述

设置用户 bucket（空间)的 cors （跨源资源共享）。如果配置已存在，那么KS3将会替换它。

使用此接口，你需要拥有执行 PutBucketCORS 操作的权限。空间拥有者默认具有此权限，并且可以授予他人相应权限。

用户通过设置此配置，可以实现指定空间响应来自不同源的请求。例如，你也许想要从一个源 http://www.example.com 以 XMLHttpRequest 的方式请求你在KS3的某个空间 my.example.bucket.com，可以通过此操作实现。

要将您的存储段配置为允许跨源请求，您可以创建一个 CORS 配置，即一个 XML 文档，其中包含一些规则，它们能够识别您允许访问存储段的源、每个源支持的操作（HTTP 方法），以及其他特定操作的信息。您可以向配置添加最多 100 条规则。将 XML 文档作为 cors 子资源添加到存储段，文件大小限制在64KB。

例如，以下针对存储段的 cors 配置包含两个指定为 CORSRule 元素的规则：

- 第一个规则允许来自 https://www.example.com 源的跨源 PUT、POST 和 DELETE 请求。该规则还通过 Access-Control-Request-Headers 标头允许预检 OPTIONS 请求中的所有标头。作为对任何预检 OPTIONS 请求的响应，KS3 将返回请求的任意标头。
- 规则二允许任一源通过 GET 方式请求跨源资源共享。“*”通配符字符是指所有的源。

```
<CORSConfiguration>
<CORSRule>
<AllowedOrigin>http://www.example.com</AllowedOrigin>
<AllowedMethod>PUT</AllowedMethod>
<AllowedMethod>POST</AllowedMethod>
<AllowedMethod>DELETE</AllowedMethod>

<AllowedHeader>*</AllowedHeader>
</CORSRule>
<CORSRule>
<AllowedOrigin>*</AllowedOrigin>
<AllowedMethod>GET</AllowedMethod>
</CORSRule>
</CORSConfiguration>
```

用户可以通过其他可选参数来对某个空间进行 cors 配置。例如，设置允许源 http://www.example.com 使用 PUT 和 POST 发送跨源资源共享请求。

```
<CORSConfiguration>
<CORSRule>
<AllowedOrigin>http://www.example.com</AllowedOrigin>
<AllowedMethod>PUT</AllowedMethod>
<AllowedMethod>POST</AllowedMethod>
<AllowedHeader>*</AllowedHeader>
<MaxAgeSeconds>3000</MaxAgeSeconds>
<ExposeHeader>x-kss-server-side-encryption</ExposeHeader>
</CORSRule>
</CORSConfiguration>
```

上面的配置中，CORSRule包含以下可选参数：

- MaxAgeSeconds—指定在 KS3 针对特定资源的预检 OPTIONS 请求作出响应后，浏览器缓存该响应的时间（以秒为单位）（在本示例中，为 3 000 秒）。通过缓存响应，在需要重复原始请求时，浏览器无需向KS3 发送预检请求。
- ExposeHeader - 识别可允许客户从应用程序（例如，从 JavaScript XMLHttpRequest 数据元）进行访问的响应标头（在本示例中，为 x-kss-server-side-encryption）。

当KS3收到针对某一空间的跨源请求，它会根据设定的跨源资源共享规则来进行匹配，为了能够正常匹配，用户提交的规则配置必须满足以下条件。

- 请求源头部必须匹配 AllowedOrigin 配置。
- 在一个预检 OPTIONS 请求中请求方法（如 PUT,GET,HEAD等）或者 Access-Control-Request-Method header 必须是 AllowedMethod 配置中的一个。
- 在一个预检 OPTIONS 请求中每一个在 Access-Control-Request-Headers 指明的头部必须匹配 AllowedHeader 配置。

请求

语法

```
PUT /?cors HTTP/1.1
Host: {BucketName}.{endpoint}
Content-Length: {length}
Date: {date}
Authorization: {SignatureValue}
Content-MD5: {MD5}

<CORSConfiguration>
<CORSRule>
<AllowedOrigin>{Origin you want to allow cross-domain requests from}</AllowedOrigin>
<AllowedOrigin>...</AllowedOrigin>
...
<AllowedMethod>{HTTP method}</AllowedMethod>
<AllowedMethod>...</AllowedMethod>
...
<MaxAgeSeconds>{Time in seconds your browser to cache the pre-flight OPTIONS response for a resource}</MaxAgeSeconds>
<AllowedHeader>{Headers that you want the browser to be allowed to send}</AllowedHeader>
<AllowedHeader>...</AllowedHeader>
...
<ExposeHeader>{Headers in the response that you want accessible from client application}</ExposeHeader>
<ExposeHeader>...</ExposeHeader>
...
</CORSRule>
```

```
<CORSRule>
...
</CORSRule>
...
</CORSConfiguration>
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口仅使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

名称	描述	必须
CORSConfiguration	包含 CORSRules 元素的容器，元素上限为100。 类型：Container 子节点：CORSRules 父节点：无	是
CORSRule	源与方法的集合，最多可以配置100条规则。 类型：Container 子节点：AllowedOrigin, AllowedMethod, MaxAgeSeconds, ExposeHeader, ID. 父节点：CORSConfiguration	是
ID	规则的唯一表示，最多255字符，它可以帮助你快速查找配置中的某一规则。 类型：String 父节点：CORSRule	是
AllowedMethod	用户允许源所能执行的 HTTP 方法，每一条 CORSRule 必须定义至少一个源地址和一种方法。 类型：Enum（GET, PUT, HEAD, POST, DELETE） 父节点：CORSRule	是
AllowedOrigin	用户允许跨源资源共享访问的源地址，其最多含有一个"*"通配符。每一条 CORSRule 必须定义至少一个源地址和一种方法。例如：http://*.example.com。另外，你可以使用"*"来代表全部源。 类型：String 父节点：CORSRule	是
AllowedHeader	指明在预检OPTION中通过 Access-Control-Request-Headers 哪些头部是可以使用的。每一个在 Access-Control-Request-Headers 中指定的头部必须要与发送到KS3请求的头部保持一致，最多使用一个"*" 类型：String 父节点：CORSRule	是
MaxAgeSeconds	指定在 KS3 针对特定资源的预检 OPTIONS 请求作出响应后，浏览器缓存该响应的时间。一个 CORSRule 最多有一个 MaxAgeSeconds 元素. 类型：Integer（seconds） 父节点：CORSRule	是
ExposeHeader	识别可允许客户从应用程序（例如，从 JavaScript XMLHttpRequest 数据元）进行访问的响应标头。 类型：String 父节点：CORSRule	是

响应

响应头部

该接口仅使用常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不使用响应内容。

特殊错误

该接口不返回错误代码。

示例

请求示例

```
PUT /?cors HTTP/1.1
Content-MD5: /Q08JEkpekFaAuWWEKS7Fg==
Date: Thu, 15 Jun 2017 02:32:02 GMT
Authorization: authorization string
Content-Length: 652
Host: ks3-example.ks3-cn-beijing.ksyun.com

<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<CORSRule>
<AllowedMethod>GET</AllowedMethod>
<AllowedOrigin>http://baidu.com</AllowedOrigin>
<AllowedHeader>*</AllowedHeader>
</CORSRule>
<CORSRule>
<AllowedMethod>POST</AllowedMethod>
<AllowedOrigin>http://*ks3.ksyun.com</AllowedOrigin>
<AllowedOrigin>https://*ks3.console.ksyun.com</AllowedOrigin>
<AllowedOrigin>https://*ks3.ksyun.com</AllowedOrigin>
<AllowedOrigin>https://www.example.com</AllowedOrigin>
<AllowedOrigin>http://*ks3.console.ksyun.com</AllowedOrigin>
<AllowedHeader>*</AllowedHeader>
</CORSRule>
</CORSConfiguration>
```

响应示例

```
HTTP/1.1 200 OK
Server: Tengine
Date: Thu, 15 Jun 2017 02:32:56 GMT
Content-Length: 0
X-Application-Context: application
x-kss-request-id: 94def5b819f74135aldf54e6cf422f64
```

DELETE Bucket CORS

描述

此操作将删除指定 bucket（空间）的 cors（跨源资源共享）配置信息。

使用此接口，你需要拥有执行 PutCORSConfiguration 操作的权限。空间拥有者默认具有此权限，并且可以授予他人相应权限。

请求

语法

```
DELETE /?cors HTTP/1.1 Host: {BucketName}.{endpoint} Date: {date} Authorization: {SignatureValue}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

请求参数

该请求不使用请求参数。

请求头部

该接口仅使用常用请求头部。获取更多信息，请点击常用请求头部。

请求内容

该接口不使用请求内容。

响应

响应头部

该接口仅使用常用响应头部。获取更多信息，请点击常用响应头部。

响应内容

该接口不使用响应内容。

示例

请求示例

```
DELETE /?cors HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Date: Wed, 24 Dec 2014 03:12:40 GMT
Authorization: Authorization String
```

响应示例

```
HTTP/1.1 204 No Content
Date: Wed, 24 Dec 2014 03:12:40 GMT
Server: Tengine
Content-Length: 0
x-kss-request-id: 73a50b1275954392bfa62eaf062601af
```

OPTIONS Object

描述

浏览器能够发送预检请求到KS3，来检测它是否能够发送一个包含指定的源，HTTP方法和请求头部的跨源资源共享请求。

通过用户为其某一空间配置 cors 规则，KS3能够支持跨源资源共享服务。当用户发送一个预检请求后，KS3会计算请求是否匹配设定的规则。

如果请求内容与规则不匹配，那么，KS3将返回一个 403 Forbidden 响应。

请求

语法

```
OPTIONS /{ObjectName} HTTP/1.1
Host: {BucketName}.{endpoint}
Origin: {Origin}
Access-Control-Request-Method: {HTTPMethod}
Access-Control-Request-Headers: {RequestHeader}
```

注意：

- endpoint与Region对应关系

请求参数

该请求不使用请求参数。

请求头部

该接口可以使用所有常用请求头部。获取更多信息，请点击常用请求头部。

名称	描述	必需
Origin	标识向KS3请求跨源资源共享的源，例如：http://www.example.com。 类型：String 默认值：无	是
Access-Control-Request-Method	标识实际发送跨源资源请求时所使用的HTTP方法。 类型：String 默认值：无	是
Access-Control-Request-Headers	一个逗点分隔的列表，标识实际发送跨源资源请求时使用的头部。 类型：String 默认值：无	否

请求内容

该接口不使用请求内容。

响应

响应头部

该接口可以使用所有常用响应头部。获取更多信息，请点击常用响应头部。

名称	描述
Access-Control-Allow-Origin	实际请求时的源地址，如果该源不被允许，在响应中，KS3不会包含此头部。 类型：String
Access-Control-Max-Age	预检请求结果能被缓存的时间，以秒为单位。 类型：String

Access-Control-Allow-Methods	源地址请求时使用的HTTP方法，如果此方法不被允许，在响应中，KS3将不包含此头部。 类型：String
Access-Control-Allow-Headers	在一个实际请求中，浏览器能发送的以逗号分隔的HTTP头部列表。如果其中任一个头部不被允许，在响应中，KS3将不会包含那个头部，同时也将不包含任一个 Access-Control 前缀的头部。 类型：String
Access-Control-Expose-Headers	逗号分隔的HTTP头部列表。在响应一个实际请求中，允许JavaScript读取这些头部信息。 类型：String

响应内容

该接口不使用响应内容。

示例

请求示例

```
OPTIONS /my-image.jpg HTTP/1.1
Host: ks3-example.ks3-cn-beijing.ksyun.com
Origin: http://www.example.com
Access-Control-Request-Method: PUT
```

响应示例

```
HTTP/1.1 200 OK
Date: Wed, 21 Aug 2012 23:09:55 GMT
Etag: "1flalaf1f111111111111c11aed1da1"
Access-Control-Allow-Origin: http://www.example.com
Access-Control-Allow-Methods: PUT
Access-Control-Expose-Headers: x-kss-request-id
Content-Length: 0
Server: Tengine
```

上传回调处理（Upload CallBack Processing）

支持上传回调处理的接口有PUT Object、COMPLETE MULTIPART UPLOAD、POST Object

1.PUT Object、COMPLETE MULTIPART UPLOAD

描述

在调用Put Object，Complete Multipart Upload接口时，可携带相关的Callback参数，实现上传回调处理（Upload CallBack Processing，UCP）。

客户端将文件及相关Callback请求头上传到KS3后，KS3会通过POST方法向用户的回调地址POST一段json数据，此json数据为客户端传入的x-kss-callbackbody请求头填充变量后转化得来。用户服务端正确处理回调后返回{"result":true}，如果回调处理错误返回{"result":false}则表示文件上传失败，如果用户服务端没有返回任何消息，则进入超时等待，默认等待时间为3秒，重试两次。只有在KS3服务端收到{"result":true}时，文件才真正上传成功。
支持的头部参数里面加入x-kss-callbackauth 值为”1”时表示开启回调鉴权功能，具体详见第三部分“回调鉴权”。

请求接口

上传回调请求保持原有接口不变，参考Put Object,Complete Multipart Upload。

请求参数

该接口不使用请求参数。

请求头部

名称	描述	必需
x-kss-callbackurl	支持http	是
x-kss-callbackbody	回调参数支持自定义参数返回、常量和魔法变量，自定义变量通过Head传回，例如objectKey=${key}&etag=${etag}&location=${kss-location} &uid=123 回调参数的value值不能包含“=”和“&”	是
x-kss-callbackauth	值为”1”时表示开启回调鉴权功能。回调鉴权功能具体详见第三部分“回调鉴权”	否
kss-location	自定义头以kss-开始	自定义

魔法变量

参数	说明	备注
bucket	文件上传的Bucket	Utf-8编码
key	文件的名称	Utf-8编码
etag	文件Md5值经过base64处理	
objectSize	文件大小	以字节标识
mimeType	文件类型	
createTime	文件创建时间	Unix时间戳表示，1420629372，精确到秒

回调响应

响应内容

上传触发处理与原有Put Object，Complete Multipart Upload返回兼容。

特殊错误

无

2.POST Object

描述

采用POST回调时，需要在请求后面加入一个postcallback的参数，如POST /?postcallback HTTP/1.1，表示这是一个POST回调请求。当加入postcallback参数时，表单项中必须同时存在回调相关的参数，否则请求会判定不合法，具体与回调服务器交互过程与PUT回调一致。其它语法和普通POST请求一样，参考Post Object。

请求接口

上传回调请求保持原有接口不变，参考Post Object。

请求参数

参数	描述	必需
postcallback	表示这是一个POST回调请求	是

表单项

名称	描述	必需
x-kss-callbackurl	支持http(s)	是
x-kss-callbackbody	回调参数支持自定义参数返回、常量和魔法变量，自定义变量通过Head传回，例如objectKey=${key}&etag=${etag}&location=${kss-location} &uid=123 回调参数的value值不能包含“=”和“&”	是
x-kss-callbackauth	值为”1”时表示开启回调鉴权功能。回调鉴权功能具体详见第三部分“回调鉴权”	否
kss-location	自定义头以kss-开始	自定义

魔法变量

参数	说明	备注
bucket	文件上传的Bucket	Utf-8编码
key	文件的名称	Utf-8编码
etag	文件Md5值经过base64处理	
objectSize	文件大小	以字节标识
mimeType	文件类型	
createTime	文件创建时间	Unix时间戳表示，1420629372，精确到秒

回调响应

响应内容

上传触发处理与原有Post Object返回兼容。

特殊错误

调用POST回调方法时，如果未传入回调必填的表单项，请求将返回400；如果传入了回调相关的表单项，但是调用的是普通（非回调）POST方法时，请求也会返回400。

回调鉴权

如果回调时用户开启了鉴权字段x-kss-callbackauth(该字段在PUT和Complete Multipart Upload请求中写在header里，在POST请求中写在表单项里)，则KS3向用户的回调地址返回的header中，就会包含Authorization和x-kss-date字段内容：

```
Authorization: ks3cbauth AKLTxxxxx5IxxxpA7xxxxxx:FIexxyY3jGeO7JxxxxxW5rBejKM=
  x-kss-date：1590132647609
```

其中ks3cbauth为固定值，AKLT开头到冒号前内容为用户此条POST请求使用的AK，冒号后面的部分是用此AK、对应SK和服务端指定时间（此处x-kss-date为回调发起时间即KS3服务器的时间）算出的签名串signature，算签名的方法和发送请求的v2签名计算方法类似，但简化一些：

```
signString = "ksscallback " + {timestamp} //形如"ksscallback 1590132647609"，中间有个空格，ksscallback是个固定值
  signature = Base64(HMAC-SHA1(YourSecretKey, UTF-8-Encoding-Of(signString)));
```

AWS S3协议兼容

目前KS3 API接口覆盖了AWS S3协议大部分的基本功能，并在持续扩充新特性。除少量AWS S3特殊特性接口未覆盖外，用户采用AWS S3协议的软件、服务基本上可以做到无缝迁移。

详细的接口兼容性情况见下表，供有需求的用户参考。

	Desciption	Name	Amazon	Kingsoft
Service Operation	获取所有bucket信息	GET Service	√	√
Bucket Operation	Bucket基本操作	DELETE Bucket	√	√
		GET Bucket (ListObjects)	√	√
		HEAD Bucket	√	√
		PUT Bucket	√	√
	Bucket cors相关操作	DELETE Bucket cors	√	√
		GET Bucket cors	√	√
		PUT Bucket cors	√	√
	Bucket lifecycle相关操作	DELETE Bucket lifecycle	√	×
		GET Bucket lifecycle	√	
		PUT Bucket lifecycle	√	
	Bucket policy相关操作	DELETE Bucket policy	√	×
		GET Bucket policy	√	
		PUT Bucket policy	√	
	Bucket tagging相关操作	DELETE Bucket tagging	√	×
		GET Bucket tagging	√	
		PUT Bucket tagging	√	
	Bucket website相关操作	DELETE Bucket website	√	×
		GET Bucket website	√	
		PUT Bucket website	√	
	Bucket logging相关操作	GET Bucket logging	√	√
		PUT Bucket logging	√	√
	Bucket notification相关操作	GET Bucket notification	√	×

		PUT Bucket notification	√	
		GET Bucket versioning	√	
	Bucket versioning相关操作	GET Bucket Object versions	√	×
		PUT Bucket versioning	√	
	Bucket acl相关操作	PUT Bucket acl	√	√
		GET Bucket acl	√	√
	Bucket requestPayment相关操作	GET Bucket requestPayment	√	×
		PUT Bucket requestPayment	√	
	枚举该Bucket下的所有分块上传	List Multi Part Uploads	√	√
	删除Object	DELETE Object	√	√
	删除多个Object	Delete Multiple Objects	√	×
	下载Object	GET Object	√	√
	获取Object ACL	GET Object ACL	√	√
	获取Object BT 种子	GET Object torrent	√	×
	获取Object 元信息	HEAD Object	√	√
	Object对HTML5浏览器跨域支持	OPTIONS Object	√	×
	浏览器表单上传Object	POST Object	√	√
	Amazon Glacier存储恢复	POST Object restore	√	×
Object Operation	上传Object	PUT Object	√	√
	设置Object ACL	PUT Object acl	√	√
	复制Object	PUT Object - Copy	√	√
		Initiate Multipart Upload	√	√
		Upload Part	√	√
	分块上传相关操作	Upload Part - Copy	√	×
		Complete Multipart Upload	√	√
		Abort Multipart Upload	√	√
		List Parts	√	√
Image Thumbnail			×	√

加密相关

KS3服务端加密使用指南

目录

---

1. 概述

服务器端加密关乎静态数据加密，即 KS3 在将您的数据写入数据中心内的磁盘时会在对象级别上加密这些数据，并在您访问这些数据时为您解密这些数据。只要您验证了您的请求并且拥有访问权限，您访问加密和未加密数据元的方式就没有区别。例如，如果您使用预签名的 URL 来共享您的对象，那么对于加密和解密对象，该 URL 的工作方式是相同的。

2. 使用方式

KS3目前支持两种方式管理加密密钥：

- 使用具有 KS3 托管密钥的服务器端加密 (SSE-S3) - 利用多因素强加密来使用不同密钥加密每个对象。作为额外的保护，它将使用定期轮换的主密钥加密密钥本身。KS3 服务器端加密使用256 位高级加密标准 (AES-256)来加密您的数据。
- 使用具有客户提供的密钥的服务器端加密 (SSE-C) - 您管理数据的加密/解密、加密密钥和相关工具。

3. API支持

3.1 使用具有 KS3 托管密钥的服务器端加密 (SSE-S3)

- PUT 操作
- POST 操作
- Initiate Multipart Upload 操作
- Upload Part 操作

PUT 操作

请求头

名称	描述
x-kss-server-side -encryption	如果请求中包含此头，服务端将对数据进行加密处理，合法值：AES256

响应头

名称	描述
x-kss-server-side -encryption	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
x-kss-server-side -encryption	如果x-kss-server-side -encryption请求头的值不是AES256

POST 操作

表单项

如果用户需要服务器使用默认加密，需要以下表单项

名称	描述
x-kss-server-side-encryption	如果存储 object 时使用了服务端加密，则需要配置该表单项，值为使用的加密算法，目前支持AES256。 类型：String

错误返回

类型	描述
加密类型无效	如果x-kss-server-side -encryption表单项的值不是AES256

Initiate Multipart Upload 操作

请求头

名称	描述
x-kss-server-side -encryption	如果请求中包含此头，服务端将生成加密密钥，并在分块上传时使用该密钥进行加密，合法值：AES256

响应头

名称	描述
x-kss-server-side -encryption	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
加密类型无效	如果x-kss-server-side -encryption请求头的值不是AES256

Upload Part 操作

响应头

名称	描述
x-kss-server-side -encryption	如果数据通过KS3 托管密钥的服务器端加密，则响应头将包含该值

错误返回

类型	描述
非法参数	如果提供x-kss-server-side -encryption请求头，将报400错误，提示参数非法

Complete Multipart Upload 操作

响应头

名称	描述
x-kss-server-side -encryption	如果数据通过KS3 托管密钥的服务器端加密，则响应头将包含该值

错误返回

类型	描述
非法参数	果提供x-kss-server-side -encryption请求头，将报400错误，提示参数非法

COPY 操作

请求头

名称	描述
x-kss-server-side -encryption	合法值：AES256
x-kss-copy-source-server-side-encryption-customer-algorithm	如果被拷贝对象为客户端提供密钥加密方式，需提供该请求头
x-kss-copy-source-server-side-encryption-customer-key	如果被拷贝对象为客户端提供密钥加密方式，需提供该请求头
x-kss-copy-source-server-side-encryption-customer-key-MD5	如果被拷贝对象为客户端提供密钥加密方式，需提供该请求头

注：x-kss-copy-source-server-side-encryption-customer-algorithm 、x-kss-copy-source-server-side-encryption-customer-key 、x-kss-copy-source-server-side-encryption-customer-key-MD5此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
加密类型无效	如果x-kss-server-side -encryption请求头的值不是AES256
MD5值错误	400错误，x-kss-copy-source-server-side-encryption-customer-key-MD5 不是 x-kss-copy-source-server-side-encryption-customer-key的MD5值
加密算法错误	400错误，x-kss-copy-source-server-side-encryption-customer-algorithm不是合法的AES256
非法参数	400错误，x-kss-copy-source-server-side-encryption-customer-algorithm 、x-kss-copy-source-server-side-encryption-customer-key 、x-kss-copy-source-server-side-encryption-customer-key-MD5 3个请求头未同时存在

GET 操作

响应头

名称	描述
x-kss-server-side -encryption	如果数据通过KS3 托管密钥的服务器端加密，则响应头将包含该值

错误返回

类型	描述
非法参数	如果提供x-kss-server-side -encryption请求头，将报400错误，提示参数非法

HEAD 操作

响应头

名称	描述
x-kss-server-side -encryption	如果数据通过KS3 托管密钥的服务器端加密，则响应头将包含该值

错误返回

类型	描述
非法参数	如果提供x-kss-server-side -encryption请求头，将报400错误，提示参数非法

3.2 通过使用客户提供的加密密钥的服务器端加密 (SSE-C) 保护数据

PUT 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值

注：此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求头里包含该值，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss-server-side -encryption -customer-key-MD5 不是 x-kss-server-side -encryption -customer-key 的MD5值
加密算法错误	400错误，x-kss-server-side -encryption -customer-algorithm不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在

POST 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值

注：此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求头里包含该值，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss-server-side -encryption -customer-key-MD5 不是 x-kss-server-side -encryption -customer-key 的MD5值 的MD5值
加密算法错误	400错误，x-kss-server-side -encryption -customer-algorithm 不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在

Initiate Multipart Upload 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值

注：此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求头里包含该值，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss-server-side -encryption -customer-key-MD5 不是 x-kss-server-side -encryption -customer-key 的MD5值
加密算法错误	400错误，x-kss-server-side -encryption -customer-algorithm不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在

Upload Part 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值

注：此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求头里包含该值，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss-server-side -encryption -customer-key-MD5 不是 x-kss-server-side -encryption -customer-key 的MD5值
加密算法错误	400错误，x-kss-server-side -encryption -customer-algorithm不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在
缺少客户密钥	400错误，数据通过客户端提供密钥加密，但是请求中未提供客户密钥
MD5值不一致	400错误，获取数据时提供的密钥的MD5和存储时不一致

Complete Multipart Upload 操作

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求头里包含该值，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
非法参数	400错误，不应该携带客户端提供密钥进行加密的请求头，如果携带，提示参数非法

COPY 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值
x-kss-copy-source-server-side-encryption-customer-algorithm	如果被拷贝对象为客户端提供密钥加密方式，需提供该请求头
x-kss-copy-source-server-side-encryption-customer-key	如果被拷贝对象为客户端提供密钥加密方式，需提供该请求头
x-kss-copy-source-server-side-encryption-customer-key-MD5	如果被拷贝对象为客户端提供密钥加密方式，需提供该请求头

注： x-kss-server...三个请求头以及x-kss-copy-source...三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求头里包含该值，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求头里包含该值，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss(-copy-source)-server-...-MD5 不是 x-kss(-copy-source)-server-...-key 的MD5值
加密算法错误	400错误，x-kss(-copy-source)-server-side -...-algorithm不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在
缺少客户密钥	400错误，数据通过客户端提供密钥加密，但是请求中未提供客户密钥
MD5值不一致	400错误，获取数据时提供的密钥的MD5和存储时不一致

GET 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值

注：此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求数据通过客户提供的加密密钥的服务器端加密，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求数据通过客户提供的加密密钥的服务器端加密，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss-server-...-MD5 不是 x-kss-server-...-key 的MD5值
加密算法错误	400错误，x-kss-server-side -...-algorithm不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在
缺少客户密钥	400错误，数据通过客户端提供密钥加密，但是请求中未提供客户密钥
MD5值不一致	400错误，获取数据时提供的密钥的MD5和存储时不一致

HEAD 操作

请求头

名称	描述
x-kss-server-side -encryption -customer-algorithm	客户端提供的加密算法，合法值：AES256
x-kss-server-side -encryption -customer-key	客户端提供的加密密钥
x-kss-server-side -encryption -customer-key-MD5	客户端提供的通过BASE64编码的通过128位MD5加密的密钥的MD5值

注：此三个请求头必须同时存在

响应头

名称	描述
x-kss-server-side -encryption -customer-algorithm	如果请求数据通过客户提供的加密密钥的服务器端加密，则响应头将包含该值
x-kss-server-side -encryption -customer-key-MD5	如果请求数据通过客户提供的加密密钥的服务器端加密，则响应头将包含该值

错误返回

类型	描述
MD5值错误	400错误，x-kss-server-...-MD5 不是 x-kss-server-...-key 的MD5值
加密算法错误	400错误，x-kss-server-side -...-algorithm不是合法的AES256
非法参数	400错误，上述3个请求头未同时存在
缺少客户密钥	400错误，数据通过客户端提供密钥加密，但是请求中未提供客户密钥
MD5值不一致	400错误，获取数据时提供的密钥的MD5和存储时不一致

低频存储相关

接口描述

KS3标准低频存储用于保存不频繁访问但在需要时也要求快速访问的数据。

用户可以在上传文件过程中，通过HTTP头指定存储类型；相应的，在下载文件过程中我们会通过HTTP响应头指示文件的存储类型。

除此之外，在用户枚举文件时也会返回文件的存储类型信息。

上传接口

Put Object 上传文件

用户上传过程中，可以通过设置 x-kss-storage-class 头来指定存储类型。 有效值为： STANDARD 或 STANDARD_IA。 对于无效的存储类型，KS3会拒绝本次请求。

请求语法：

```
PUT /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

SDK示例代码：

```
PutObjectRequest request = new PutObjectRequest(bucketName, key, file);
request.setStorageClass(StorageClass.StandardInfrequentAccess);
PutObjectResult result = client.putObject(request); // 假设已经合理初始化Ks3Client
```

Post Object 上传文件

用户可以在表单域中通过设置 x-kss-storage-class 域来指定存储类型。 有效值为： STANDARD 或 STANDARD_IA。 对于无效的存储类型，KS3会拒绝本次请求。

Java SDK不支持POST方式上传文件。

Initiate Multipart Upload 初始化分块上传

用户可以在初始化分块上传时通过 x-kss-storage-class 请求头设置文件的存储类型。 有效值为： STANDARD 或 STANDARD_IA。 对于无效的存储类型，KS3会拒绝本次请求。

请求语法：

```
POST /{ObjectKey}?uploads HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
```

```
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

SDK示例代码：

```
InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest(bucketName, key);
request.setStorageClass(StorageClass.StandardInfrequentAccess);
InitiateMultipartUploadResult result = client.initiateMultipartUpload(request); // 假设已经合理初始化Ks3Client
```

后续的所有分块上传操作(Upload Part、Complete、Abort)都不再接受 x-kss-storage-class 请求头。

Put Object Copy 复制文件

对于复制文件，KS3支持重新指定存储类型。若不指定目的文件的存储类型，则按照目标存储的默认存储类型保存。

Upload Part Copy 复制分块

对于复制分块，KS3目前不支持重新指定存储类型。目的文件的存储类型必须与源文件存储类型一致。 对于复制分块，KS3会忽略 x-kss-storage-class 请求头。 如果源文件存储类型与目的文件存储类型不一致，会报错。提示 InvalidStorageClass。

下载接口

Get Object 下载文件

```
GetObjectRequest request = new GetObjectRequest(bucketName, key);
GetObjectResult result = client.getObject(request);
Ks3Object object = result.getObject();
object.getObjectContent();
```

Head Object 查看文件元信息

Java SDK中，可以通过获取元信息来查看存储类型，示例代码如下：

```
HeadObjectRequest request = new HeadObjectRequest(bucketName, key);
HeadObjectResult result =client.headObject(request);
String storageClass = result.getObjectMetadata().getStorageClass();
```

对于标准存储文件，getStorageClass 返回 null；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

枚举接口

Get Bucket 枚举bucket下的所有文件

用户获取某个Bucket下文件时，KS3会为每个文件返回其存储类型。

SDK示例代码：

```
ListObjectsRequest request = new ListObjectsRequest(bucketName);
ObjectListing listing = client.listObjects(request);
for (Ks3ObjectSummary summary : listing.getObjectSummaries()) {
    String storageClas = summary.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Multipart Uploads 查看bucket下的分块上传

用户在查看当前bucket下所有的分块上传请求时，KS3会为每个分块上传返回其存储类型。

SDK示例代码：

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest(bucketName);
ListMultipartUploadsResult result = client.listMultipartUploads(request);
for (MultiPartUploadInfo uploadInfo : result.getUploads()) {
    String storageClass = uploadInfo.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Parts 查看已上传的块

用户在列举某个分块上传已经上传的块时，KS3会返回当前这次分块上传的存储类型。

SDK示例代码：

```
ListPartsRequest reqest = new ListPartsRequest(bucketName, key, result.getUploadId());
ListPartsResult result = client.listParts(reqest);
String storageClass = result.getStorageClass();
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

归档存储相关

KS3归档存储（ACHIVE)适合需要长期保存（建议3个月以上）的归档数据，在存储周期内很少被访问，数据进入到可读取状态需要1分钟到10分钟的解冻时间。适合需要长期保存的档案数据、医疗影像、科学资料、影视素材。

您想采取归档存储类型来存储文件，您可以通过以下方式：

1. 利用上传接口，指定请求头x-kss-storage-class为ARCHIVE
2. 创建一个归档类型Bucket，无需指定存储类型，即可按照默认归档类型存储。
3. 通过生命周期转化，设定存储类型转化规则，按照规则系统自动到期转化为归档存储类型

上传接口

Put Object 上传归档文件

上传文件时，如果不指定请求头x-kss-storage-class，如果Bucket是归档类型，Object自动为归档类型，如果Bucket是非归档类型，Object自动为标准类型；如果指定Object 的x-kss-storageClass则以用户指定为准，例如在非归档存储空间中，指定Object 的x-kss-storageClass为ARCHIVE时，则该文件为归档存储类型。

bucket类型	x-kss-storage-class请求头	文件类型
归档存储类型	不上传	归档类型
归档存储类型	ARCHIVE	归档类型
归档存储类型	STANDAND	标准类型
归档存储类型	STANDARD_IA	低频类型
非归档存储类型	不上传	标准类型

非归档存储类型	STANDAND	标准类型
非归档存储类型	STANDARD_IA	低频类型
非归档存储类型	ARCHIVE	归档类型

请求语法：

```
PUT /{ObjectKey} HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

Post Object 上传文件

用户可以在表单域中通过设置 x-kss-storage-class 域来指定存储类型。规则和Put Object上传文件一致。

Initiate Multipart Upload 初始化分块上传

用户可以在初始化分块上传时通过 x-kss-storage-class 请求头设置文件的存储类型。 有效值为： STANDARD 、STANDARD_IA 、ARCHIVE。规则和Put Object上传文件一致。 对于无效的存储类型，KS3会拒绝本次请求。

请求语法：

```
POST /{ObjectKey}?uploads HTTP/1.1
Host: {BucketName}.{endpoint}
Date: {date}
Authorization: {SignatureValue}
x-kss-storage-class: {StorageClass}
```

注意：

- endpoint与Region对应关系
- SignatureValue签名算法

后续的所有分块上传操作(Upload Part、Complete、Abort)都不再接受 x-kss-storage-class 请求头。

Put Object Copy 复制文件

对于复制文件，KS3支持重新指定存储类型。 如果复制源文件是归档存储文件，成功复制的前提是，源文件必须完成解冻，为可读状态。

Upload Part Copy 复制分块

对于复制分块，KS3目前不支持重新指定存储类型。目的文件的存储类型必须与源文件存储类型一致。 对于复制分块，KS3会忽略 x-kss-storage-class 请求头。 如果源文件存储类型与目的文件存储类型不一致，会报错。提示 InvalidStorageClass。

解冻接口

详解解冻接口

下载接口

说明 归档文件必须完成解冻，才能正常下载。

Get Object 下载文件

```
GetObjectRequest request = new GetObjectRequest(bucketName, key);
GetObjectResult result = client.getObject(request);
Ks3Object object = result.getObject();
object.getObjectContent();
```

Head Object 查看文件元信息

Java SDK中，可以通过获取元信息来查看存储类型，示例代码如下：

```
HeadObjectRequest request = new HeadObjectRequest(bucketName, key);
HeadObjectResult result =client.headObject(request);
String storageClass = result.getObjectMetadata().getStorageClass();
```

对于标准存储文件，getStorageClass 返回 null；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

枚举接口

Get Bucket 枚举bucket下的所有文件

用户获取某个Bucket下文件时，KS3会为每个文件返回其存储类型。

SDK示例代码：

```
ListObjectsRequest request = new ListObjectsRequest(bucketName);
ObjectListing listing = client.listObjects(request);
for (Ks3ObjectSummary summary : listing.getObjectSummaries()) {
    String storageClas = summary.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Multipart Uploads 查看bucket下的分块上传

用户在查看当前bucket下所有的分块上传请求时，KS3会为每个分块上传返回其存储类型。

SDK示例代码：

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest(bucketName);
ListMultipartUploadsResult result = client.listMultipartUploads(request);
for (MultiPartUploadInfo uploadInfo : result.getUploads()) {
    String storageClass = uploadInfo.getStorageClass();
}
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。

List Parts 查看已上传的块

用户在列举某个分块上传已经上传的块时，KS3会返回当前这次分块上传的存储类型。

SDK示例代码：

```
ListPartsRequest reqest = new ListPartsRequest(bucketName, key, result.getUploadId());
ListPartsResult result = client.listParts(reqest);
String storageClass = result.getStorageClass();
```

对于标准存储文件，getStorageClass 返回 STANDARD；对于标准低频存储文件，getStorageClass 返回 STANDARD_IA；对于归档存储文件，getStorageClass 返回 ARCHIVE。