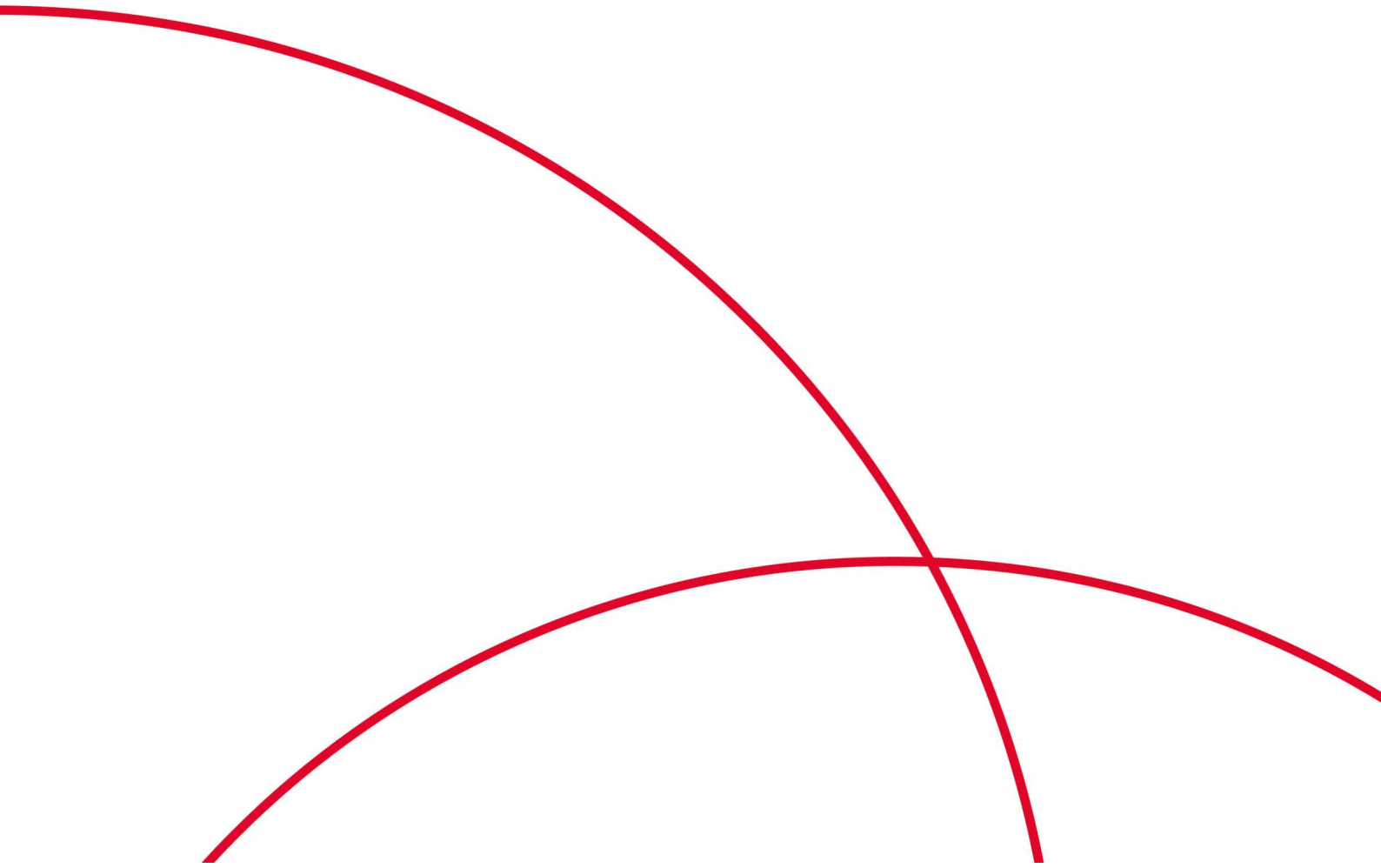
天翼云•对象存储 OBS

API 接口参考

天翼云科技有限公司



目 录

1 使用前必读 6
1.1 概述 6
1.2 基本概念 6
2 API 概览 8
3 如何调用 API 13
3.1 构造请求 13
3.2 认证鉴权 18
3.2.1 用户签名验证 18
3.2.2 Header 中携带签名 19
3.2.3 URL 中携带签名 29
3.2.4 基于浏览器上传的表单中携带签名 37
3.3 返回结果 43
4 快速入门 45
4.1 创建桶 45
4.2 获取桶列表 48
4.3 上传对象 51
5 API 56
5.1 桶的基础操作 56
5.1.1 获取桶列表 56
5.1.2 创建桶 60
5.1.3 列举桶内对象 69
5.1.4 获取桶元数据 89
5.1.5 获取桶区域位置 95
5.1.6 删除桶 96
5.2 桶的高级配置 98
5.2.1 设置桶策略 98
5.2.2 获取桶策略 102

5.2.3 删除桶策略 103
5.2.4 设置桶 ACL 105
5.2.5 获取桶 ACL 109
5.2.6 设置桶日志管理配置 112
5.2.7 获取桶日志管理配置 118
5.2.8 设置桶的生命周期配置 121
5.2.9 获取桶的生命周期配置 131
5.2.10 删除桶的生命周期配置 137
5.2.11 设置桶的多版本状态 138
5.2.12 获取桶的多版本状态 141
5.2.13 设置桶默认存储类型 143
5.2.14 获取桶默认存储类型 144
5.2.15 设置桶的跨区域复制配置 146
5.2.16 获取桶的跨区域复制配置 153
5.2.17 删除桶的跨区域复制配置 156
5.2.18 设置桶标签 158
5.2.19 获取桶标签 161
5.2.20 删除桶标签 165
5.2.21 设置桶配额 166
5.2.22 获取桶配额 168
5.2.23 获取桶存量信息 169
5.2.24 设置桶清单 171
5.2.25 获取桶清单 176
5.2.26 列举桶清单 181
5.2.27 删除桶清单 183
5.2.28 设置桶的自定义域名 185
5.2.29 获取桶的自定义域名 187
5.2.30 删除桶的自定义域名 190
5.2.31 设置桶的加密配置 191
5.2.32 获取桶的加密配置 195
5.2.33 删除桶的加密配置 199
5.2.34 设置桶归档存储对象直读策略 200
5.2.35 获取桶归档存储对象直读策略 203
5.2.36 删除桶归档存储对象直读策略 205
5.2.37 配置桶级默认 WORM 策略 206
5.2.38 获取桶级默认 WORM 策略 211
5.3 静态网站托管 215
5.3.1 设置桶的网站配置 215
5.3.2 获取桶的网站配置 220

5.3.3 删除桶的网站配置 222
5.3.4 设置桶的 CORS 配置 223
5.3.5 获取桶的 CORS 配置 228
5.3.6 删除桶的 CORS 配置 231
5.3.7 OPTIONS 桶-OptionsBucket 233
5.3.8 OPTIONS 对象-OptionsObject 236
5.4 对象操作 239
5.4.1 PUT 上传 239
5.4.2 POST 上传 253
5.4.3 复制对象 278
5.4.4 下载对象 294
5.4.5 获取对象元数据 311
5.4.6 删除对象 321
5.4.7 批量删除对象 323
5.4.8 恢复归档存储对象 330
5.4.9 追加写对象 333
5.4.10 设置对象 ACL 344
5.4.11 获取对象 ACL 350
5.4.12 修改对象元数据 354
5.4.13 修改写对象 364
5.4.14 截断对象 366
5.4.15 重命名对象 368
5.4.16 配置对象级 WORM 保护策略 370
5.5 多段操作 374
5.5.1 列举桶中已初始化多段任务 374
5.5.2 初始化上传段任务 382
5.5.3 上传段 393
5.5.4 拷贝段 399
5.5.5 列举已上传未合并的段 409
5.5.6 合并段 415
5.5.7 取消多段上传任务 422
5.6 服务端加密 424
5.6.1 服务端加密简介 424
5.6.2 服务端加密 SSE-KMS 方式 425
5.6.3 服务端加密 SSE-OBS 方式 428
5.6.4 服务端加密 SSE-C 方式 431
5.6.5 与服务端加密相关的接口 435

6 错误码 438

7 IAM 策略和授权项 446
7.1 IAM 策略及授权项说明 446
7.2 桶相关授权项 447
7.3 对象相关授权项 451
8 附录 453
8.1 状态码 453
8.2 获取访问密钥（AK/SK） 453
8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID 454
8.4 并发一致性说明 455
A 修订记录 458

1 使用前必读

1.1 概述

欢迎使用对象存储服务 OBS（Object Storage Service）。对象存储服务提供海量、安全、高可靠、低成本的数据存储能力，可供用户存储任意类型和大小的数据。适合企业备份/归档、视频点播、视频监控等多种数据存储场景。

您可以使用本文档提供的 API 对 OBS 进行相关操作，如创建、修改、删除桶，上传、下载、删除对象等。支持的全部操作请参见 2 API 概览。

在调用 OBS API 之前，请确保已经充分了解 OBS 相关概念。详细信息请参见 1.2 基本概念。

OBS 提供了 REST（Representational State Transfer）风格 API，支持您通过 HTTP/HTTPS 请求调用，调用方法请参见 3 如何调用 API。

说明

TLS1.0 /TLS1.1 协议版本存在安全漏洞，建议使用更高级的安全版本。

终端节点

终端节点即调用 API 的**请求地址**，不同服务不同区域的终端节点不同，您可以向企业管理员获取区域和终端节点信息。

OBS 在每个区域都提供独立的二级域名，访问 OBS 服务既可以使用 OBS 提供的域名，也可以使用自定义域名。

1.2 基本概念

使用 OBS API 涉及的常用概念

- 账号

用户注册时的账号，账号对其所拥有的资源及云服务具有完全的访问权限，可以重置用户密码、分配用户权限等。由于账号是付费主体，为了确保账号安全，建议您不要直接使用账号进行日常管理工作，而是创建用户并使用他们进行日常管理工作。

- 用户

  由账号在 IAM 中创建的用户，是云服务的使用人员，具有身份凭证（密码和访问密钥）。

  在控制台“我的凭证”下，您可以查看账号 ID 和用户 ID，管理账号或 IAM 用户的访问密钥。

  通常在调用 API 的鉴权过程中，您需要用到账号和 IAM 用户的访问密钥。

- 桶

  桶是用于存放对象的容器，是 OBS 中最高等级的命名空间。每个对象都存放在一个桶中。例如，如果名为“picture.jpg”的对象存放在“photo”桶中，则可使用 URL（http://photo.obs.*region*.*example*.com/picture.jpg）对该对象进行寻址。

- 对象

  对象在 OBS 中是最基本的实体。在一个桶中可以存放多个对象，OBS 系统并不能区分对象的文件类型。在 OBS 系统中存储的对象是被序列化了的，因此它可能是一个文本文件或者一个视频文件。OBS 支持的数据大小范围可以是 0B 到 48.8TB（包含 0B 和 48.8TB），PutObject 接口上传对象时对象最大为 5GB，超过 5GB 对象需要使用多段方式上传。

- 区域

  指云资源所在的物理位置，同一区域（region）内可用区间内网互通，不同区域间内网不互通。通过在不同地区创建云资源，可以将应用程序设计的更接近特定客户的要求，或满足不同地区的法律或其他要求。

  OBS 中的桶归属于某个区域，这个区域是您在创建桶时选定的，桶一旦创建，其归属区域信息不能改变。您可以根据地理位置、成本、满足法规要求等标准来选择桶的区域。可以选择的区域请参见终端节点。

- 企业项目

  企业项目是项目的升级版，针对企业不同项目间资源的分组和管理，是逻辑隔离。企业项目中可以包含多个区域的资源，且项目中的资源可以迁入迁出。关于企业项目 ID 的获取及企业项目特性的详细信息，请参见《企业项目管理服务用户指南》。

2 API 概览

桶基础操作接口

表 2-1 桶基础操作接口

接口	说明
5.1.1 获取桶列表	查询自己创建的桶列表。
5.1.2 创建桶	创建一个新桶。可以在创建时添加不同的请求消息头来指定桶的区域、权限控制策略、存储类型等信息。
5.1.3 列举桶内对象	获取桶内对象列表。可以在创建时添加不同的请求消息头来获取符合指定前缀、标识符等要求的对象。
5.1.4 获取桶元数据	查询桶元数据是否存在。可以查询桶的区域、存储类型、OBS 服务版本号、企业项目 id、CORS 配置等信息。
5.1.5 获取桶区域位置	获取桶区域位置信息。
5.1.6 删除桶	删除指定的桶。删除之前需要确保桶内无对象。

桶高级配置接口

表 2-2 桶高级配置接口

接口	说明
5.2.1 设置桶策略	创建或者修改一个桶的策略。如果桶已经存在一个策略，那么当前请求中的策略将完全覆盖桶中现存的策略。
5.2.2 获取桶策略	获取指定桶的策略信息。

接口	说明
5.2.3 删除桶策略	删除一个指定桶上的策略。
5.2.4 设置桶 ACL	设置一个指定桶的 ACL 信息。通过 ACL 可以控制桶的读写权限。
5.2.5 获取桶 ACL	获取一个指定桶的 ACL 信息。
5.2.6 设置桶日志管理配置	开启或关闭桶的日志管理功能。开启后，桶的每次操作将会产生一条日志，并将多条日志打包成一个日志文件存放在指定的位置。
5.2.7 获取桶日志管理配置	获取指定桶的日志管理配置信息。
5.2.8 设置桶的生命周期配置	指定规则来实现定时删除或迁移桶中对象。
5.2.9 获取桶的生命周期配置	获取指定桶已配置的生命周期规则。
5.2.10 删除桶的生命周期配置	删除指定桶的生命周期配置信息。
5.2.11 设置桶的多版本状态	开启或暂停桶的多版本功能。开启后，可以检索和还原各个版本的对象，在意外操作或应用程序故障时快速恢复数据。
5.2.12 获取桶的多版本状态	获取指定桶的多版本功能状态。
5.2.13 设置桶默认存储类型	创建或更新桶的默认存储类型配置信息。
5.2.14 获取桶默认存储类型	获取桶的默认存储类型配置信息。
5.2.15 设置桶的跨区域复制配置	设置桶的跨区域复制功能。通过激活跨区域复制，OBS 可将新创建的对象及修改的对象从一个源桶复制到不同区域中的目标桶。
5.2.16 获取桶的跨区域复制配置	获取指定桶的跨区域复制配置信息。
5.2.17 删除桶的跨区域复制配置	删除指定桶的跨区域复制配置信息。
5.2.18 设置桶标签	添加标签至一个已存在的桶。为桶添加标签后，该桶上所有请求产生的计费话单里都会带上这些标签，从而可以针对话单报表做分类筛选，进行更详细的成本分析。
5.2.19 获取桶标签	获取指定桶的标签。
5.2.20 删除桶标签	删除指定桶的标签。
5.2.21 设置桶配额	设置桶的空间配额，用以限制桶的最大存储容量。
5.2.22 获取桶配额	获取桶的空间配额。
5.2.23 获取桶存量信息	获取桶中的对象个数及对象占用空间。
5.2.24 设置桶清单	为一个桶配置清单规则。桶清单可以用来帮助您

接口	说明
	管理桶内对象，它可以定期列举桶内对象，并将对象元数据的相关信息保存在 CSV 格式的文件中，上传到您指定的桶中。
5.2.25 获取桶清单	获取指定桶的某个清单规则。
5.2.26 列举桶清单	获取指定桶的所有清单规则。
5.2.27 删除桶清单	删除指定桶的某个清单规则。
5.2.28 设置桶的自定义域名	为桶设置自定义域名。设置成功之后，用户可以通过桶的自定义域名访问桶。
5.2.29 获取桶的自定义域名	查询桶已设置的自定义域名。
5.2.30 删除桶的自定义域名	删除桶已设置的自定义域名。
5.2.31 设置桶的加密配置	为桶创建或更新默认服务端加密配置信息。设置桶加密配置后，在该桶中上传对象时，会采用桶的默认加密配置对数据进行加密。
5.2.32 获取桶的加密配置	查询桶的默认服务端加密配置信息。
5.2.33 删除桶的加密配置	删除桶的默认服务端加密配置信息。
5.2.34 设置桶归档存储对象直读策略	开启或关闭桶的归档存储对象直读功能。开启后，归档存储对象不需要恢复便可以直接下载。
5.2.35 获取桶归档存储对象直读策略	获取指定桶的归档存储对象直读状态。
5.2.36 删除桶归档存储对象直读策略	删除指定桶的归档存储对象直读配置信息。
5.2.37 配置桶级默认 WORM 策略	桶的 WORM 开关开启后，支持配置默认保护策略和保护期限。
5.2.38 获取桶级默认 WORM 策略	获取指定桶设置的桶级默认 WORM 策略。

静态网站托管接口

表 2-3 静态网站托管接口

接口	说明
5.3.1 设置桶的网站配置	创建或更新桶的网站配置信息。OBS 允许在桶内保存静态的网页资源，如.html 网页文件、flash 文件、音视频文件等，当客户端通过桶的 Website 接入点访问这些对象资源时，浏览器可以直接解析出这些支持的网页资源，呈现给最终用户。

接口	说明
5.3.2 获取桶的网站配置	获取桶的网站配置信息。
5.3.3 删除桶的网站配置	删除桶的网站配置信息。
5.3.4 设置桶的 CORS 配置	设置桶的跨域资源共享配置信息。OBS 允许在桶内保存静态的网页资源，在正确的使用下，OBS 的桶可以成为网站资源。只有进行了适当的 CORS 配置，OBS 中的网站才能响应另一个网站的跨域请求。
5.3.5 获取桶的 CORS 配置	获取桶的跨域资源共享配置信息。
5.3.6 删除桶的 CORS 配置	删除桶的跨域资源共享配置信息。
5.3.7 OPTIONS 桶-OptionsBucket	检测客户端是否具有对服务端进行操作的权限。通常用于跨域访问之前。
5.3.8 OPTIONS 对象-OptionsObject	检测客户端是否具有对服务端进行操作的权限。通常用于跨域访问之前。

对象操作接口

表 2-4 对象操作接口

接口	说明
5.4.1 PUT 上传	上传简单对象到指定的桶。
5.4.2 POST 上传	基于表单上传对象到指定的桶。
5.4.3 复制对象	为 OBS 上已经存在的对象创建一个副本。
5.4.4 下载对象	下载对象。
5.4.5 获取对象元数据	获取对象的元数据信息。包括对象的过期时间、版本号、CORS 配置等信息。
5.4.6 删除对象	删除指定的对象。也可以携带 versionId 删除指定版本的对象。
5.4.7 批量删除对象	将一个桶内的一部分对象一次性删除，删除后不可恢复。
5.4.8 恢复归档存储对象	将归档存储对象的内容恢复，恢复后才能下载。
5.4.9 追加写对象	在指定桶内的一个对象尾追加上传数据，不存在相同对象键值的对象则创建新对象。
5.4.10 设置对象 ACL	设置一个指定对象的 ACL 信息。通过 ACL 可以控制对象的读写权限。
5.4.11 获取对象 ACL	获取一个指定对象的 ACL 信息。

接口	说明
5.4.12 修改对象元数据	添加、修改或删除桶中已经上传的对象的元数据。
5.4.13 修改写对象	将指定并行文件系统内的一个对象从指定位置起修改为其他内容。
5.4.14 截断对象	将指定并行文件系统内的一个对象截断到指定大小。
5.4.15 重命名对象	将指定并行文件系统内的一个对象重命名为其他对象名。
5.4.16 配置对象级 WORM 保护策略	开启了 WORM 开关的桶，上传的对象支持配置或修改对象保护期限。

多段操作接口

表 2-5 多段操作接口

接口	说明
5.5.1 列举桶中已初始化多段任务	查询一个桶中所有的初始化后还未合并以及未取消的多段上传任务。
5.5.2 初始化上传段任务	使用多段上传特性时，必须首先调用此接口初始化上传段任务，获取全局唯一的多段上传任务号，用于后续的上传段、合并段、列举段等操作。
5.5.3 上传段	为特定的任务上传段。
5.5.4 拷贝段	将已上传对象的一部分或全部拷贝为段。
5.5.5 列举已上传未合并的段	查询一个任务所属的所有段信息。
5.5.6 合并段	将指定的段合并成一个完整的对象。
5.5.7 取消多段上传任务	取消一个多段上传的任务。

3 如何调用 API

3.1 构造请求

本节介绍 REST API 请求的组成。

请求 URI

OBS 根据桶和对象及携带的资源参数来确定具体的 URI，当需要进行资源操作时，可以使用这个 URI 地址。

URI 的一般格式为（方括号内为可选项）：

protocol://[bucket.]domain[:port][/object][?param]

表 3-1 URI 中的参数

参数名称	参数类型	是否必选	描述
protocol	String	必选	**参数解释：** 请求使用的协议类型。 **约束限制：** 不涉及 **取值范围：** • HTTP：超文本传输协议，使用明文传输数据，容易被窃听和篡改。通常用于传输不包含敏感信息的网站，如新闻网站等。 • HTTPS：安全超文本传输协议，使用 SSL/TLS 协议对数据进行加密，提供安全可靠的通信，用于传输需要保护的敏感信息。 **默认取值：** 无

参数名称	参数类型	是否必选	描述
bucket	String	可选	**参数解释：** 桶名。 **约束限制：** • 桶的名字需全局唯一，不能与已有的任何桶名称重复，包括其他用户创建的桶。 • 桶命名规则如下： – 3～63 个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。 – 禁止使用 IP 地址。 – 禁止以“-”或“.”开头及结尾。 – 禁止两个“.”相邻（如：“my..bucket”）。 – 禁止“.”和“-”相邻（如：“my-.bucket”和“my.-bucket”）。 • 同一用户在同一个区域多次创建同名桶不会报错，创建的桶属性以第一次请求为准。 **默认取值：** 无
domain	String	必选	**参数解释：** 各区域的终端节点（Endpoint），例如 obs.*region*.example.com **约束限制：** 不涉及 **默认取值：** 不涉及
port	String	可选	**参数解释：** 请求使用的端口号。根据软件服务器的部署不同而不同。 **约束限制：** 不涉及 **取值范围：** OBS 对象存储服务的 HTTP 方式访问端口为 80，HTTPS 方式访问端口为 443。 **默认取值：** 缺省时使用默认端口，各种传输协议都有默认的端口号，如 HTTP 的默认端口为 80，HTTPS 的默认端口为 443。
object	String	可选	**参数解释：** 对象名。对象名是对象在存储桶中的唯一标识。对象名是

参数名称	参数类型	是否必选	描述
			对象在桶中的完整路径，路径中不包含桶名。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
param	String	可选	**参数解释：** 请求使用的桶和对象的具体资源。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不设置默认为请求桶或对象自身资源。

须知

除获取桶列表之外的所有接口，都应当包含桶名。OBS 基于 DNS 解析性能和可靠性的考虑，要求凡是携带桶名的请求，在构造 URL 的时候都必须将桶名放在 domain 前面，形成三级域名形式，又称为虚拟主机访问域名。

例如，如果您有一个位于 a1 区域的名为 test-bucket 的桶，期望访问桶中一个名为 test-object 对象的 acl，正确的访问 URL 为 https://test-bucket.obs.a1.example.com/test-object?acl

请求方法

HTTP 方法（也称为操作或动词），它告诉服务您正在请求什么类型的操作。

表 3-2 对象存储支持的 REST 请求方法

方法	说明
GET	请求服务器返回指定资源，如获取桶列表、下载对象等。
PUT	请求服务器更新指定资源，如创建桶、上传对象等。
POST	请求服务器新增资源或执行特殊操作，如初始化上传段任务、合并段等。
DELETE	请求服务器删除指定资源，如删除对象等。
HEAD	请求服务器返回指定资源的概要，如获取对象元数据等。
OPTIONS	请求服务器检查是否具有某个资源的操作权限，需要桶配置 CORS。

请求消息头

可选的附加请求头字段，如指定的 URI 和 HTTP 方法所要求的字段。详细的公共请求消息头字段请参见表 3-3。

表 3-3 公共请求消息头

消息头名称	参数类型	是否必选	描述
Authorization	String	有条件必选	**参数解释：** 请求消息中可带的签名信息。 **约束限制：** 匿名请求不需要带，其他请求必选。 **取值范围：** 不涉及 **默认取值：** 不涉及
Content-Length	String	有条件必选	**参数解释：** 消息（不包含消息头）长度。 **约束限制：** PUT 操作可选，加载 XML 的操作必须带。 **取值范围：** 不涉及 **默认取值：** 不涉及
Content-Type	String	否	**参数解释：** 资源内容的类型，例如： text/plain。 **约束限制：** 不涉及 **默认取值：** 如果上传对象时未携带该字段，下载对象时默认返回的 Content-Type 为 binary/octet-stream。
Date	String	有条件必选	**参数解释：** 请求发起端的日期和时间，例如：Wed, 27 Jun 2018 13:39:15 +0000。 **约束限制：** 如果是匿名请求或者消息头中带了 x-obs-date 字段，则可以不带该字段，其他情况下必选。 **取值范围：**

消息头名称	参数类型	是否必选	描述
			不涉及 **默认取值：** 不涉及
Host	String	是	**参数解释：** 桶的访问域名，格式为*桶名.终端节点*，即*BucketName.Endpoint*，例如 obs.*region*.example.com 如 examplebucketname.obs.*region*.example.com。 **约束限制：** 桶名的约束限制如下： • 桶的名字需全局唯一，不能与已有的任何桶名称重复，包括其他用户创建的桶。 • 桶命名规则如下： – 3～63 个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。 – 禁止使用 IP 地址。 – 禁止以“-”或“.”开头及结尾。 – 禁止两个“.”相邻（如：“my..bucket”）。 – 禁止“.”和“-”相邻（如：“my-.bucket”和“my.-bucket”）。 • 同一用户在同一个区域多次创建同名桶不会报错，创建的桶属性以第一次请求为准。 **默认取值：** 不涉及

请求消息体（可选）

请求消息体通常以结构化格式（如 JSON 或 XML）发出，与请求消息头中 Content-type 对应，传递除请求消息头之外的内容。如果请求消息体中参数支持中文，则中文字符必须为 UTF-8 编码。

每个接口的请求消息体内容不同，也并不是每个接口都需要有请求消息体（或者说消息体为空），GET、DELETE 操作类型的接口就不需要消息体，消息体具体内容需要根据具体接口而定。

发起请求

共有两种方式可以基于已构建好的请求消息发起请求，分别为：

- cURL

  cURL 是一个命令行工具，用来执行各种 URL 操作和信息传输。cURL 充当的是 HTTP 客户端，可以发送 HTTP 请求给服务端，并接收响应消息。cURL 适用于接

口调试。关于 cURL 详细信息请参见 https://curl.haxx.se/。由于 cURL 无法计算签名，使用 cURL 时仅支持访问匿名的公共 OBS 资源。

- 编码

  通过编码调用接口，组装请求消息，并发送处理请求消息。

3.2 认证鉴权

3.2.1 用户签名验证

OBS 通过 AK/SK 对请求进行签名，在向 OBS 发送请求时，客户端发送的每个消息头需要包含由 SK、请求时间、请求类型等信息生成的签名信息。

- AK(Access Key ID)：访问密钥 ID。与私有访问密钥关联的唯一标识符；访问密钥 ID 和私有访问密钥一起使用，对请求进行加密签名。
- SK(Secret Access Key)：与访问密钥 ID 结合使用的密钥，对请求进行加密签名，可标识发送方，并防止请求被修改。

用户可以在 IAM 服务中获取 AK 和 SK，获取的方法请参见 8.2 获取访问密钥（AK/SK）。

OBS 根据应用场景，提供了 3.2.2 Header 中携带签名、3.2.3 URL 中携带签名和 3.2.4 基于浏览器上传的表单中携带签名 3 种签名计算方式。

以 Header 中携带签名为例，用户签名验证流程如表 3-4 所示。Header 中携带签名方法的具体参数说明及代码示例，请参见 3.2.2 Header 中携带签名。

表 3-4 OBS 签名计算和验证步骤

步骤		示例
签名计算	1. 构造 HTTP 消息	PUT /object HTTP/1.1 Host: bucket.obs.region.example.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913
	2. 按照签名规则计算 **StringToSign**	StringToSign = HTTP-Verb + "\n" + Content-MD5 + "\n" + Content-Type + "\n" + Date + "\n" + CanonicalizedHeaders + CanonicalizedResource
	3. 准备 AK 和 SK	AK: ****** SK: ******
	4. 计算签名 **Signature**	Signature = Base64(HMAC-SHA1(**SecretAccessKeyID**, UTF-8-Encoding-Of(**StringToSign**)))
	5. 添加签名头域发送到 OBS 服务	PUT /object HTTP/1.1 Host: bucket.obs.region.example.com

步骤		示例
		Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913 Authorization: OBS **AccessKeyID**:**Signature**
签名验证	6. 接收 HTTP 消息	PUT /object HTTP/1.1 Host: bucket.obs.region.example.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913 Authorization: OBS **AccessKeyID**:**Signature**
	7. 根据请求中的 AK 获取 SK	从头域 Authorization 中取出 AK，去 IAM 取回用户的 SK
	8. 按照签名规则计算 **StringToSign**	StringToSign = HTTP-Verb + "\n" + Content-MD5 + "\n" + Content-Type + "\n" + Date + "\n" + CanonicalizedHeaders + CanonicalizedResource
	9. 计算签名 **Signature**	Signature = Base64(HMAC-SHA1(**SecretAccessKeyID**, UTF-8-Encoding-Of(**StringToSign**)))
	10. 验证签名	验证头域 Authorization 中的 **Signature** 与服务端计算的 **Signature** 是否相等 相等：签名验证通过 不相等：签名验证失败

3.2.2 Header 中携带签名

OBS 的所有 API 接口都可以通过在 header 中携带签名方式来进行身份认证，也是最常用的身份认证方式。

在 Header 中携带签名是指将通过 HTTP 消息中 Authorization header 头域携带签名信息，消息头域的格式为：

```
Authorization: OBS AccessKeyID:Signature
```

签名的计算过程如下：

1. 构造请求字符串(StringToSign)。
2. 对第一步的结果进行 UTF-8 编码。
3. 使用 SK 对第二步的结果进行 HMAC-SHA1 签名计算。
4. 对第三步的结果进行 Base64 编码，得到签名。

请求字符串(StringToSign)按照如下规则进行构造，各个参数的含义如表 3-5 所示。

```
StringToSign =
   HTTP-Verb + "\n" +
```

```
Content-MD5 + "\n" +
Content-Type + "\n" +
Date + "\n" +
CanonicalizedHeaders + CanonicalizedResource
```

表 3-5 构造 StringToSign 所需参数说明

参数	描述
HTTP-Verb	指接口操作的方法，对 REST 接口而言，即为 http 请求操作的 VERB，如："PUT"，"GET"，"DELETE"等字符串。
Content-MD5	按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128-bit MD5 值经过 base64 编码后得到的字符串，可以为空。具体请参见表 3-10 以及表下方的计算方法示例。
Content-Type	内容类型，用于指定消息类型，例如： text/plain。 当请求中不带该头域时，该参数按照空字符串处理，见表 3-6。
Date	生成请求的时间，该时间格式遵循 RFC 1123；该时间与当前服务器的时间超过 15 分钟时服务端返回 403。 当有自定义字段 x-obs-date 时，该参数按照空字符串处理；见表 3-10。 如果进行临时授权方式操作（如临时授权方式获取对象内容等操作）时，该参数不需要。
Canonicalized Headers	HTTP 请求头域中的 OBS 请求头字段，即以“x-obs-”作为前缀的头域，如“x-obs-date，x-obs-acl，x-obs-meta-*”。用户在调用 API 时，请根据自身需求，从调用的 API 支持的头域中选取。 1. 请求头字段中关键字的所有字符要转为小写（但内容值需要区分大小写，如“x-obs-storage-class:STANDARD”），需要添加多个字段时，要将所有字段按照关键字的字典序从小到大进行排序。 2. 在添加请求头字段时，如果有重名的字段，则需要进行合并。如：x-obs-meta-name:name1 和 x-obs-meta-name:name2，则需要先将重名字段的值（这里是 name1 和 name2）以逗号分隔，合并成 x-obs-meta-name:name1,name2。 3. 头域中的请求头字段中的关键字不允许含有非 ASCII 码或不可识别字符；请求头字段中的值也不建议使用非 ASCII 码或不可识别字符，如果一定要使用非 ASCII 码或不可识别字符，需要客户端自行做编解码处理，可以采用 URL 编码或者 Base64 编码，服务端不会做解码处理。 4. 当请求头字段中含有无意义空格或 Tab 键时，需要摒弃。例如：x-obs-meta-name: name（name 前带有一个无意义空格），需要转换为：x-obs-meta-name:name 5. 每一个请求头字段最后都需要另起新行，见表 3-8。
Canonicalized Resource	表示 HTTP 请求所指定的 OBS 资源，构造方式如下： <桶名+对象名>+[子资源 1] + [子资源 2] + ... 1. 桶名和对象名，例如：/bucket/object。如果没有对象名，如列举

参数	描述
	桶，则为"/bucket/"，如桶名也没有，则为“/”。 2. 如果有子资源，则将子资源添加进来，例如?acl，?logging。 OBS 支持各种子资源，包括：acl, append, attname, cors, customdomain, delete, deletebucket, directcoldaccess, encryption, inventory, length, lifecycle, location, logging, metadata, modify, name, partNumber, policy, position, quota, rename, replication, response-cache-control, response-content-disposition, response-content-encoding, response-content-language, response-content-type, response-expires, restore, storageClass, storagePolicy, storageinfo, tagging, torrent, truncate, uploadId, uploads, versionId, versioning, versions, website, x-image-process, x-image-save-bucket, x-image-save-object, x-obs-security-token, object-lock, retention。 3. 如果有多个子资源，在包含这些子资源时，需要首先将这些子资源按照其关键字的字典序从小到大排列，并使用**“&”**拼接。 **说明** ● 子资源通常是唯一的，不建议请求的 URL 包含多个相同关键字的子资源（例如，key=value1&key=value2），如果存在这种情况，OBS 服务端签名时只会计算第一个子资源且也只有第一个子资源的值会对实际业务产生作用； ● 以获取对象（GetObject）接口为例，假设桶名为 bucket-test，对象名为 object-test，对象的版本号为 xxx，获取时需要重写 Content-Type 为 text/plain，那么签名计算出的 CanonicalizedResource 为：/bucket-test/object-test?response-content-type=text/plain&versionId=xxx。

下面的几张表提供了一些生成 StringToSign 的例子。

表 3-6 获取对象

请求消息头	StringToSign
GET /object.txt HTTP/1.1 Host: bucket.obs.*region*.example.com Date: Sat, 12 Oct 2015 08:12:38 GMT	GET \n \n \n Sat, 12 Oct 2015 08:12:38 GMT\n /bucket/object.txt

表 3-7 使用临时 AK/SK 和 securitytoken 上传对象

请求消息头	StringToSign
PUT /object.txt HTTP/1.1 User-Agent: curl/7.15.5 Host: bucket.obs.*region*.example.com x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT x-obs-security-token: YwkaRTbdY8g7q.... content-type: text/plain	PUT\n \n text/plain\n \n x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT\n

请求消息头	StringToSign
Content-Length: 5913339	x-obs-security-token:YwkaRTbdY8g7q....\n /bucket/object.txt

表 3-8 带请求头字段上传对象

请求消息头	StringToSign
PUT /object.txt HTTP/1.1 User-Agent: curl/7.15.5 Host: bucket.obs.*region*.example.com Date: Mon, 14 Oct 2015 12:08:34 GMT x-obs-acl: public-read content-type: text/plain Content-Length: 5913339	PUT\n \n text/plain\n Mon, 14 Oct 2015 12:08:34 GMT\n x-obs-acl:public-read\n /bucket/object.txt

表 3-9 获取对象 ACL

请求消息头	StringToSign
GET /object.txt?acl HTTP/1.1 Host: bucket.obs.*region*.example.com Date: Sat, 12 Oct 2015 08:12:38 GMT	GET \n \n \n Sat, 12 Oct 2015 08:12:38 GMT\n /bucket/object.txt?acl

表 3-10 上传对象且携带 Content-MD5 头域

请求消息头	StringToSign
PUT /object.txt HTTP/1.1 Host: bucket.obs.*region*.example.com x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT Content-MD5: I5pU0r4+sgO9Emgl1KMQUg== Content-Length: 5913339	PUT\n I5pU0r4+sgO9Emgl1KMQUg==\n \n \n x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT\n /bucket/object.txt

表 3-11 使用自定义域名方式上传对象

请求消息头	StringToSign
PUT /object.txt HTTP/1.1 Host: obs.ccc.com	PUT\n I5pU0r4+sgO9Emgl1KMQUg==\n

请求消息头	StringToSign
x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT Content-MD5: I5pU0r4+sgO9Emgl1KMQUg== Content-Length: 5913339	\n \n x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT\n /obs.ccc.com/object.txt

Java 中 Content-MD5 的计算方法示例

```
import java.security.MessageDigest;
import sun.misc.BASE64Encoder;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;

public class Md5{
    public static void main(String[] args) {
        try {
              String exampleString = "blog";
              MessageDigest messageDigest = MessageDigest.getInstance("MD5");
              BASE64Encoder encoder = new BASE64Encoder();
              String contentMd5 =
encoder.encode(messageDigest.digest(exampleString.getBytes("utf-8")));
              System.out.println("Content-MD5:" + contentMd5);
        } catch (NoSuchAlgorithmException | UnsupportedEncodingException e)
        {
              e.printStackTrace();
        }
    }
}
```

根据请求字符串(StringToSign)和用户 SK 使用如下算法生成 Signature，生成过程使用 HMAC 算法(hash-based authentication code algorithm)。

```
Signature = Base64( HMAC-SHA1( Your_SK, UTF-8-Encoding-Of( StringToSign ) ) )
```

例如在某区域创建桶名为 newbucketname2 的私有桶，客户端请求格式为：

```
PUT / HTTP/1.1
Host: newbucketname2.obs.region.example.com
Content-Length: length
Date: Fri, 06 Jul 2018 03:45:51 GMT
x-obs-acl:private
x-obs-storage-class:STANDARD
Authorization: OBS UDSIAMSTUBTEST000254:ydH8ffpcbS6YpeOMcEZfn0wE90c=
<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>region</Location>
</CreateBucketConfiguration>
```

Java 中签名的计算方法

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.security.InvalidKeyException;
```

```
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Base64;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.TreeMap;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public class SignDemo {

   private static final String SIGN_SEP = "\n";

   private static final String OBS_PREFIX = "x-obs-";

   private static final String DEFAULT_ENCODING = "UTF-8";

   private static final List<String> SUB_RESOURCES =
Collections.unmodifiableList(Arrays.asList(
      "CDNNotifyConfiguration", "acl", "append", "attname",  "cors", "customdomain",
"delete",
    "deletebucket", "directcoldaccess", "encryption", "inventory", "length",
"lifecycle", "location", "logging",
    "metadata", "mirrorBackToSource", "modify", "name",
    "partNumber", "policy", "position", "quota","rename", "replication", "response-
cache-control",
    "response-content-disposition","response-content-encoding", "response-content-
language", "response-content-type",
    "response-expires","restore", "storageClass", "storagePolicy", "storageinfo",
"tagging", "torrent", "truncate",
    "uploadId", "uploads", "versionId", "versioning", "versions", "website", "x-
image-process",
     "x-image-save-bucket", "x-image-save-object", "x-obs-security-token", "object-
lock", "retention"));

   private String ak;

   private String sk;

    public String urlEncode(String input) throws UnsupportedEncodingException {
      return URLEncoder.encode(input, DEFAULT_ENCODING)
          .replaceAll("%7E", "~") //for browser
          .replaceAll("%2F", "/")
          .replaceAll("%20", "+");
   }

   private String join(List<?> items, String delimiter) {
      StringBuilder sb = new StringBuilder();
      for (int i = 0; i < items.size(); i++) {
         String item = items.get(i).toString();
```

```
            sb.append(item);
            if (i < items.size() - 1) {
                sb.append(delimiter);
            }
        }
        return sb.toString();
    }

    private boolean isValid(String input) {
        return input != null && !input.equals("");
    }

    public String hmacSha1(String input) throws NoSuchAlgorithmException,
InvalidKeyException, UnsupportedEncodingException {
        SecretKeySpec signingKey = new
SecretKeySpec(this.sk.getBytes(DEFAULT_ENCODING), "HmacSHA1");
        Mac mac = Mac.getInstance("HmacSHA1");
        mac.init(signingKey);
        return
Base64.getEncoder().encodeToString(mac.doFinal(input.getBytes(DEFAULT_ENCODING)));
    }

    private String stringToSign(String httpMethod, Map<String, String[]> headers,
Map<String, String> queries,
        String bucketName, String objectName) throws Exception{
        String contentMd5 = "";
        String contentType = "";
        String date = "";

        TreeMap<String, String> canonicalizedHeaders = new TreeMap<String, String>();

        String key;
        List<String> temp = new ArrayList<String>();
        for(Map.Entry<String, String[]> entry : headers.entrySet()) {
            key = entry.getKey();
            if(key == null || entry.getValue() == null || entry.getValue().length ==
0) {
                continue;
            }

            key = key.trim().toLowerCase(Locale.ENGLISH);
            if(key.equals("content-md5")) {
                contentMd5 = entry.getValue()[0];
                continue;
            }

            if(key.equals("content-type")) {
                contentType = entry.getValue()[0];
                continue;
            }

            if(key.equals("date")) {
                date = entry.getValue()[0];
                continue;
            }
```

```
            if(key.startsWith(OBS_PREFIX)) {
                for(String value : entry.getValue()) {
                    if(value != null) {
                        temp.add(value.trim());
                    }
                }
                canonicalizedHeaders.put(key, this.join(temp, ","));
                temp.clear();
            }
        }

        if(canonicalizedHeaders.containsKey("x-obs-date")) {
            date = "";
        }

        // handle method/content-md5/content-type/date
        StringBuilder stringToSign = new StringBuilder();
        stringToSign.append(httpMethod).append(SIGN_SEP)
            .append(contentMd5).append(SIGN_SEP)
            .append(contentType).append(SIGN_SEP)
            .append(date).append(SIGN_SEP);

        // handle canonicalizedHeaders
        for(Map.Entry<String, String> entry : canonicalizedHeaders.entrySet()) {

stringToSign.append(entry.getKey()).append(":").append(entry.getValue()).append(SIG
N_SEP);
        }

        // handle CanonicalizedResource
        stringToSign.append("/");
        if(this.isValid(bucketName)) {
            stringToSign.append(bucketName).append("/");
            if(this.isValid(objectName)) {
                stringToSign.append(this.urlEncode(objectName));
            }
        }

        TreeMap<String, String> canonicalizedResource = new TreeMap<String,
String>();
        for(Map.Entry<String, String> entry : queries.entrySet()) {
            key = entry.getKey();
            if(key == null) {
                continue;
            }

            if(SUB_RESOURCES.contains(key)) {
                canonicalizedResource.put(key, entry.getValue());
            }
        }

        if(canonicalizedResource.size() > 0) {
            stringToSign.append("?");
            for(Map.Entry<String, String> entry : canonicalizedResource.entrySet()) {
```

```
                stringToSign.append(entry.getKey());
                if(this.isValid(entry.getValue())) {
                    stringToSign.append("=").append(entry.getValue());
                }
                stringToSign.append("&");
            }
            stringToSign.deleteCharAt(stringToSign.length()-1);
        }

        //   System.out.println(String.format("StringToSign:%s%s", SIGN_SEP,
stringToSign.toString()));

        return stringToSign.toString();
    }

    public String headerSignature(String httpMethod, Map<String, String[]> headers,
Map<String, String> queries,
        String bucketName, String objectName) throws Exception {

        //1. stringToSign
        String stringToSign = this.stringToSign(httpMethod, headers, queries,
bucketName, objectName);

        //2. signature
        return String.format("OBS %s:%s", this.ak, this.hmacSha1(stringToSign));
    }

    public String querySignature(String httpMethod, Map<String, String[]> headers,
Map<String, String> queries,
        String bucketName, String objectName, long expires) throws Exception {
        if(headers.containsKey("x-obs-date")) {
            headers.put("x-obs-date", new String[] {String.valueOf(expires)});
        } else {
            headers.put("date", new String[] {String.valueOf(expires)});
        }
        //1. stringToSign
        String stringToSign = this.stringToSign(httpMethod, headers, queries,
bucketName, objectName);

        //2. signature
        return this.urlEncode(this.hmacSha1(stringToSign));
    }

    public static void main(String[] args) throws Exception {
        SignDemo demo = new SignDemo();

        /* 认证用的 ak 和 sk 硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量
中密文存放，使用时解密，确保安全；
        本示例以 ak 和 sk 保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量 YOUR_AK 和
YOUR_SK。*/
        demo.ak = System.getenv("YOUR_AK");
        demo.sk = System.getenv("YOUR_SK");

        String bucketName = "bucket-test";
        String objectName = "hello.jpg";
```

```
        Map<String, String[]> headers = new HashMap<String, String[]>();
        headers.put("date", new String[] {"Sat, 12 Oct 2015 08:12:38 GMT"});
        headers.put("x-obs-acl", new String[] {"public-read"});
        headers.put("x-obs-meta-key1", new String[] {"value1"});
        headers.put("x-obs-meta-key2", new String[] {"value2", "value3"});
        Map<String, String> queries = new HashMap<String, String>();
        queries.put("acl", null);

        System.out.println(demo.headerSignature("PUT", headers, queries, bucketName,
objectName));
    }

}
```

签名计算的样例结果为（按照执行时间的不同变化）：
ydH8ffpcbS6YpeOMcEZfn0wE90c=

Python 中签名的计算方法

```
import os
import sys
import hashlib
import hmac
import binascii
from datetime import datetime
IS_PYTHON2 = sys.version_info.major == 2 or sys.version < '3'

# 认证用的 ak 和 sk 硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以 ak 和 sk 保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量 YOUR_AK 和 YOUR_SK。
Your_SK = os.getenv('YOUR_SK')
httpMethod = "PUT"
contentType = "application/xml"
# "date" is the time when the request was actually generated
date = datetime.utcnow().strftime('%a, %d %b %Y %H:%M:%S GMT')
canonicalizedHeaders = "x-obs-acl:private\n"
CanonicalizedResource = "/newbucketname2"
canonical_string = httpMethod + "\n" + "\n" + contentType + "\n" + date + "\n" +
canonicalizedHeaders + CanonicalizedResource
if IS_PYTHON2:
    hashed = hmac.new(Your_SK, canonical_string, hashlib.sha1)
    encode_canonical = binascii.b2a_base64(hashed.digest())[:-1]
else:
    hashed = hmac.new(Your_SK.encode('UTF-8'), canonical_string.encode('UTF-8'),
hashlib.sha1)
    encode_canonical = binascii.b2a_base64(hashed.digest())[:-1].decode('UTF-8')

print(encode_canonical)
```

签名计算的样例结果为（按照执行时间的不同变化）：
ydH8ffpcbS6YpeOMcEZfn0wE90c=

3.2.3 URL 中携带签名

URL 中携带签名：OBS 服务支持用户构造一个特定操作的 URL，这个 URL 中会包含用户 AK、签名、有效期、资源等信息，任何拿到这个 URL 的人均可执行这个操作，OBS 服务收到这个请求后认为该请求就是签发 URL 用户自己在执行操作。例如构造一个携带签名信息的下载对象的 URL，拿到相应 URL 的人能下载这个对象，但该 URL 只在 Expires 指定的失效时间内有效。URL 中携带签名主要用于在不提供给其他人 Secret Access Key 的情况下，让其他人能用预签发的 URL 来进行身份认证，并执行预定义的操作。

URL 中携带签名请求的消息格式如下：

```
GET /ObjectKey?AccessKeyId=AccessKeyID&Expires=ExpiresValue&Signature=signature
HTTP/1.1
Host: bucketname.obs.region.example.com
```

URL 中使用临时 AK，SK 和 securitytoken 下载对象消息格式如下：

```
GET /ObjectKey?AccessKeyId=AccessKeyID&Expires=ExpiresValue&Signature=signature&x-
obs-security-token=securitytoken HTTP/1.1
Host: bucketname.obs.region.example.com
```

参数具体意义如表 3-12 所示。

表 3-12 请求消息参数

参数名称	描述	是否必选
AccessKeyId	签发者的 AK 信息。OBS 根据 AK 确定签发者的身份，并认为 URL 就是签发者在访问。 类型：String	是
Expires	临时授权失效的时间；此处以 UNIX 时间戳的形式来计量（以 1970 年 1 月 1 日零时为起点往后数秒），超过该时间后，URL 签名失效。 类型：String	是
Signature	根据用户 SK、Expires 等参数计算出的签名信息。 类型：String	是
x-obs-security-token	使用临时 AK/SK 鉴权时，临时 AK/SK 和 securitytoken 必须同时使用，请求头中需要添加“x-obs-security-token”字段；	否

签名的计算过程如下：

1. 构造请求字符串(StringToSign)。
2. 对第一步的结果进行 UTF-8 编码。
3. 使用 SK 对第二步的结果进行 HMAC-SHA1 签名计算。
4. 对第三步的结果进行 Base64 编码。

5. 对第四步的结果进行 URL 编码，得到签名。

请求字符串(StringToSign)按照如下规则进行构造，各个参数的含义如表 3-13 所示：

```
StringToSign =
    HTTP-Verb + "\n" +
    Content-MD5 + "\n" +
    Content-Type + "\n" +
    Expires + "\n" +
    CanonicalizedHeaders + CanonicalizedResource
```

表 3-13 构造 StringToSign 所需参数说明

参数	描述
HTTP-Verb	指接口操作的方法，对 REST 接口而言，即为 http 请求操作的 VERB，如："PUT"，"GET"，"DELETE"等字符串。
Content-MD5	按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128-bit MD5 值经过 base64 编码后得到的字符串，可以为空。
Content-Type	内容类型，用于指定消息类型，例如： text/plain。 当请求中不带该头域时，该参数按照空字符串处理。
Expires	临时授权的失效时间，即请求消息参数 Expires 的值 ExpiresValue。
Canonicalized Headers	HTTP 请求头域中的 OBS 请求头字段，即以“x-obs-”作为前缀的头域，如“x-obs-date，x-obs-acl，x-obs-meta-*”。 1. 请求头字段中关键字的所有字符要转为小写，需要添加多个字段时，要将所有字段按照关键字的字典序从小到大进行排序。 2. 在添加请求头字段时，如果有重名的字段，则需要进行合并。如：x-obs-meta-name:name1 和 x-obs-meta-name:name2，则需要先将重名字段的值（这里是 name1 和 name2）以逗号分隔，合并成 x-obs-meta-name:name1,name2。 3. 头域中的请求头字段中的关键字不允许含有非 ASCII 码或不可识别字符；请求头字段中的值也不建议使用非 ASCII 码或不可识别字符，如果一定要使用非 ASCII 码或不可识别字符，需要客户端自行做编解码处理，可以采用 URL 编码或者 Base64 编码，服务端不会做解码处理。 4. 当请求头字段中含有无意义空格或 Tab 键时，需要摒弃。例如：x-obs-meta-name: name（name 前带有一个无意义空格），需要转换为：x-obs-meta-name:name 5. 每一个请求头字段最后都需要另起新行。
Canonicalized Resource	表示 HTTP 请求所指定的 OBS 资源，构造方式如下： <桶名+对象名>+[子资源]+ [子资源 2] + ... 1. 桶名和对象名，例如：/bucket/object。如果没有对象名，如列举桶，则为"/bucket/"，如桶名也没有，则为“/”。 2. 如果有子资源，则将子资源添加进来，例如?acl，?logging。 OBS 支持各种子资源，包括：acl, append, attname, cors, customdomain, delete, deletebucket, directcoldaccess, encryption,

参数	描述
	inventory, length, lifecycle, location, logging, metadata, modify, name, partNumber, policy, position, quota, rename, replication, response-cache-control, response-content-disposition, response-content-encoding, response-content-language, response-content-type, response-expires, restore, storageClass, storagePolicy, storageinfo, tagging, torrent, truncate, uploadId, uploads, versionId, versioning, versions, website, x-image-process, x-image-save-bucket, x-image-save-object, x-obs-security-token, object-lock, retention。 3. 如果有多个子资源，在包含这些子资源时，需要首先将这些子资源按照其关键字的字典序从小到大排列，并使用**“&”**拼接。 **说明** • 子资源通常是唯一的，不建议请求的 URL 包含多个相同关键字的子资源（例如，key=value1&key=value2），如果存在这种情况，OBS 服务端签名时只会计算第一个子资源且也只有第一个子资源的值会对实际业务产生作用； • 以获取对象（GetObject）接口为例，假设桶名为 bucket-test，对象名为 object-test，对象的版本号为 xxx，获取时需要重写 Content-Type 为 text/plain，那么签名计算出的 CanonicalizedResource 为：/bucket-test/object-test?response-content-type=text/plain&versionId=xxx。

根据请求字符串(StringToSign)和用户 SK 使用如下算法生成 Signature，生成过程使用 HMAC 算法(hash-based authentication code algorithm)。

```
Signature = URL-Encode( Base64( HMAC-SHA1( Your_SK, UTF-8-Encoding-
Of( StringToSign ) ) ) )
```

URL 中的 Signature 计算方法和 Header 中携带的 Authorization 签名计算方法有两处不同：

- URL 中签名在 Base64 编码后还要经过 URL 编码。
- StringToSign 中的 Expires 和原来 Authorization 消息中的消息头 Date 对应。

使用 URL 携带签名方式为浏览器生成预定义的 URL 实例：

表 3-14 下载对象在 URL 中携带签名的请求及 StringToSign

请求消息头	StringToSign
GET /objectkey?AccessKeyId=MFyfvK41ba2giqM7Uio6PznpdUKGpownRZlmVmHc&Expires=1532779451&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D HTTP/1.1 Host: examplebucket.obs.*region*.example.com	GET \n \n \n 1532779451\n /examplebucket/objectkey

表 3-15 在 URL 中使用临时 AK/SK 和 securitytoken 下载对象请求及 StringToSign

请求消息头	StringToSign

请求消息头	StringToSign
GET /objectkey?AccessKeyId=MFyfvK41ba2giqM7Uio6PznpdUKGpownRZlmVmHc&Expires=1532779451&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D&x-obs-security-token=YwkaRTbdY8g7q.... HTTP/1.1 Host: examplebucket.obs.*region*.example.com	GET \n \n \n 1532779451\n /examplebucket/objectkey?x-obs-security-token=YwkaRTbdY8g7q....

根据签名计算规则

```
Signature = URL-Encode( Base64( HMAC-SHA1( Your_SK, UTF-8-Encoding-
Of( StringToSign ) ) ) )
```

计算出签名，然后将 Host 作为 URL 的前缀，可以生成预定义的 URL:

http(s)://examplebucket.obs.*region*.example.com/objectkey?AccessKeyId=AccessKeyID&Expires=1532779451&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D

在浏览器中直接输入该地址则可以下载 examplebucket 桶中的 objectkey 对象。这个链接的有效期是 1532779451(Sat Jul 28 20:04:11 CST 2018)。

在 Linux 环境上使用 curl 命令访问注意&字符需要\转义，如下命令将对象 objectkey 下载到 output 文件中：

curl http(s)://examplebucket.obs.*region*.example.com/objectkey?AccessKeyId=AccessKeyID\&Expires=1532779451\&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D -X GET -o output

说明

如果想要在浏览器中使用 URL 中携带签名生成的预定义 URL，则计算签名时不要使用只能携带在头域部分的“Content-MD5”、“Content-Type”、“CanonicalizedHeaders”来计算签名。否则浏览器不能携带这些参数，请求发送到服务端之后，会提示签名错误。

Java 中签名的计算方法

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Base64;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.TreeMap;
import java.util.regex.Pattern;
```

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public class SignDemo {

   private static final String SIGN_SEP = "\n";

   private static final String OBS_PREFIX = "x-obs-";

   private static final String DEFAULT_ENCODING = "UTF-8";

   private static final List<String> SUB_RESOURCES =
Collections.unmodifiableList(Arrays.asList(
         "CDNNotifyConfiguration", "acl", "append", "attname",  "cors",
"customdomain", "delete",
         "deletebucket", "directcoldaccess", "encryption", "inventory", "length",
"lifecycle", "location", "logging",
         "metadata", "mirrorBackToSource", "modify", "name",
         "partNumber", "policy", "position", "quota","rename", "replication",
"response-cache-control",
         "response-content-disposition","response-content-encoding", "response-
content-language", "response-content-type",
         "response-expires","restore", "storageClass", "storagePolicy",
"storageinfo", "tagging", "torrent", "truncate",
         "uploadId", "uploads", "versionId", "versioning", "versions", "website",
"x-image-process",
         "x-image-save-bucket", "x-image-save-object", "x-obs-security-token",
"object-lock", "retention"));

   private String ak;

   private String sk;

   private boolean isBucketNameValid(String bucketName) {
      if (bucketName == null || bucketName.length() > 63 || bucketName.length() <
3) {
         return false;
      }

      if (!Pattern.matches("^[a-z0-9][a-z0-9.-]+$", bucketName)) {
         return false;
      }

      if (Pattern.matches("(\\d{1,3}\\.){3}\\d{1,3}", bucketName)) {
         return false;
      }

      String[] fragments = bucketName.split("\\.");
      for (int i = 0; i < fragments.length; i++) {
         if (Pattern.matches("^-.*", fragments[i]) || Pattern.matches(".*-$",
fragments[i])
               || Pattern.matches("^$", fragments[i])) {
            return false;
         }
      }
```

```
        return true;
    }

    public String encodeUrlString(String path) throws UnsupportedEncodingException {
        return URLEncoder.encode(path, DEFAULT_ENCODING)
                .replaceAll("\\+", "%20")
                .replaceAll("\\*", "%2A")
                .replaceAll("%7E", "~");
    }

    public String encodeObjectName(String objectName) throws
UnsupportedEncodingException {
        StringBuilder result = new StringBuilder();
        String[] tokens = objectName.split("/");
        for (int i = 0; i < tokens.length; i++) {
            result.append(this.encodeUrlString(tokens[i]));
            if (i < tokens.length - 1) {
                result.append("/");
            }
        }
        return result.toString();
    }

    private String join(List<?> items, String delimiter) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < items.size(); i++) {
            String item = items.get(i).toString();
            sb.append(item);
            if (i < items.size() - 1) {
                sb.append(delimiter);
            }
        }
        return sb.toString();
    }

    private boolean isValid(String input) {
        return input != null && !input.equals("");
    }

    public String hmacSha1(String input) throws NoSuchAlgorithmException,
InvalidKeyException, UnsupportedEncodingException {
        SecretKeySpec signingKey = new
SecretKeySpec(this.sk.getBytes(DEFAULT_ENCODING), "HmacSHA1");
        Mac mac = Mac.getInstance("HmacSHA1");
        mac.init(signingKey);
        return
Base64.getEncoder().encodeToString(mac.doFinal(input.getBytes(DEFAULT_ENCODING)));
    }

    private String stringToSign(String httpMethod, Map<String, String[]> headers,
Map<String, String> queries,
                            String bucketName, String objectName, long expires)
throws Exception {
        String contentMd5 = "";
```

```
        String contentType = "";
        TreeMap<String, String> canonicalizedHeaders = new TreeMap<String, String>();
        String key;
        List<String> temp = new ArrayList<String>();
        for (Map.Entry<String, String[]> entry : headers.entrySet()) {
            key = entry.getKey();
            if (key == null || entry.getValue() == null || entry.getValue().length ==
0) {
                continue;
            }
            key = key.trim().toLowerCase(Locale.ENGLISH);
            if (key.equals("content-md5")) {
                contentMd5 = entry.getValue()[0];
                continue;
            }
            if (key.equals("content-type")) {
                contentType = entry.getValue()[0];
                continue;
            }
            if (key.startsWith(OBS_PREFIX)) {
                for (String value : entry.getValue()) {
                    if (value != null) {
                        temp.add(value.trim());
                    }
                }
                canonicalizedHeaders.put(key, this.join(temp, ","));
                temp.clear();
            }
        }
        // handle method/content-md5/content-type
        StringBuilder stringToSign = new StringBuilder();
        stringToSign.append(httpMethod).append(SIGN_SEP)
                .append(contentMd5).append(SIGN_SEP)
                .append(contentType).append(SIGN_SEP)
                .append(expires).append(SIGN_SEP);


        // handle canonicalizedHeaders
        for (Map.Entry<String, String> entry : canonicalizedHeaders.entrySet()) {

stringToSign.append(entry.getKey()).append(":").append(entry.getValue()).append(SIG
N_SEP);
        }


        // handle CanonicalizedResource
        stringToSign.append("/");
        if (this.isValid(bucketName)) {
            stringToSign.append(bucketName).append("/");
            if (this.isValid(objectName)) {
                stringToSign.append(this.encodeObjectName(objectName));
            }
        }

        TreeMap<String, String> canonicalizedResource = new TreeMap<String,
```

```
String>();
      for (Map.Entry<String, String> entry : queries.entrySet()) {
         key = entry.getKey();
         if (key == null) {
            continue;
         }

         if (SUB_RESOURCES.contains(key)) {
            canonicalizedResource.put(key, entry.getValue());
         }
      }

      if (canonicalizedResource.size() > 0) {
         stringToSign.append("?");
         for (Map.Entry<String, String> entry : canonicalizedResource.entrySet())
{
            stringToSign.append(entry.getKey());
            if (this.isValid(entry.getValue())) {
               stringToSign.append("=").append(entry.getValue());
            }
            stringToSign.append("&");
         }
         stringToSign.deleteCharAt(stringToSign.length() - 1);
      }
      //    System.out.println(String.format("StringToSign:%s%s", SIGN_SEP,
stringToSign.toString()));

      return stringToSign.toString();
   }

   public String querySignature(String httpMethod, Map<String, String[]> headers,
Map<String, String> queries,
                            String bucketName, String objectName, long expires)
throws Exception {
       if (!isBucketNameValid(bucketName)) {
           throw new IllegalArgumentException("the bucketName is illegal");
       }
       //1. stringToSign
       String stringToSign = this.stringToSign(httpMethod, headers, queries,
bucketName, objectName, expires);

       //2. signature
       return this.encodeUrlString(this.hmacSha1(stringToSign));
    }

   public String getURL(String endpoint, Map<String, String> queries,
                        String bucketName, String objectName, String signature, long
expires) throws UnsupportedEncodingException {
      StringBuilder URL = new StringBuilder();

URL.append("https://").append(bucketName).append(".").append(endpoint).append("/").
            append(this.encodeObjectName(objectName)).append("?");
      String key;
      for (Map.Entry<String, String> entry : queries.entrySet()) {
         key = entry.getKey();
```

```
            if (key == null) {
                continue;
            }
            if (SUB_RESOURCES.contains(key)) {
                String value = entry.getValue();
                URL.append(key);
                if (value != null) {
                    URL.append("=").append(value).append("&");
                } else {
                    URL.append("&");
                }
            }
        }

URL.append("AccessKeyId=").append(this.ak).append("&Expires=").append(expires).
                append("&Signature=").append(signature);
        return URL.toString();
    }

    public static void main(String[] args) throws Exception {
        SignDemo demo = new SignDemo();

        /* 认证用的 ak 和 sk 硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
        本示例以 ak 和 sk 保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量 YOUR_AK 和 YOUR_SK。*/
     demo.ak = System.getenv("YOUR_AK");
     demo.sk = System.getenv("YOUR_SK");
        String endpoint = "<your-endpoint>";

        String bucketName = "bucket-test";
        String objectName = "hello.jpg";

        // 如果直接使用 URL 在浏览器地址栏中访问，无法带上头域，此处 headers 加入头域会导致签名不匹配，使用 headers 需要客户端处理
        Map<String, String[]> headers = new HashMap<String, String[]>();
        Map<String, String> queries = new HashMap<String, String>();

        // 请求消息参数 Expires，设置 24 小时后失效
        long expires = (System.currentTimeMillis() + 86400000L) / 1000;
        String signature = demo.querySignature("GET", headers, queries, bucketName, objectName, expires);
        System.out.println(signature);
        String URL = demo.getURL(endpoint, queries, bucketName, objectName, signature, expires);
        System.out.println(URL);
    }
}
```

3.2.4 基于浏览器上传的表单中携带签名

OBS 服务支持基于浏览器的 POST 上传对象请求，此类请求的签名信息通过表单的方式上传。POST 上传对象：首先，创建一个安全策略，指定请求中需要满足的条件，比

如：桶名、对象名前缀；然后，创建一个基于此策略的签名，需要签名的请求表单中必须包含有效的 Signature 和 policy；最后，创建一个表单将对象上传到桶中。

签名的计算过程如下：

1. 对 policy 内容进行 UTF-8 编码。
2. 对第一步的结果进行 Base64 编码。
3. 使用 SK 对第二步的结果进行 HMAC-SHA1 签名计算。
4. 对第三步的结果进行 Base64 编码，得到签名。

```
StringToSign = Base64( UTF-8-Encoding-Of( policy ) )
Signature = Base64( HMAC-SHA1( Your_SK, StringToSign ) )
```

Policy 的内容如下：

```
{ "expiration": "2017-12-31T12:00:00.000Z",
  "conditions": [
    {"x-obs-acl": "public-read" },
    {"x-obs-security-token": "YwkaRTbdY8g7q...." },
    {"bucket": "book" },
    ["starts-with", "$key", "user/"]
  ]
}
```

Policy 策略中包含有效时间 Expiration 和条件元素 Conditions。

Expiration

描述本次签名的有效时间 ISO 8601 UTC，如实例中"expiration": "2017-12-31T12:00:00.000Z"表示请求在 2017 年 12 月 31 日 12 点之后无效。该字段是 policy 中必选字段。合法格式仅有"yyyy-MM-dd'T'HH:mm:ss'Z'"和"yyyy-MM-dd'T'HH:mm:ss.SSS'Z'"。

Conditions

Conditions 是一个用于验证本次请求合法的一种机制，可以使用这些条件限制请求中必须包含的内容。实例中的条件要求请求的桶名必须是 book，对象名必须以 user/为前缀，对象的 acl 必须是公共可读。除了 AccessKeyId、Signature、file、policy、token、field names 以及前缀为 x-ignore-外的表单中的所有项，都需要包含在 policy 中。下表是 conditions 中应该包含的项：

表 3-16 policy 中应该包含的条件元素

元素名称	描述
x-obs-acl	请求中的 ACL。 支持精确匹配和 starts-with 条件匹配。
content-length-range	设置上传对象的最大最小长度，支持 range 匹配。
Cache-Control, Content-Type, Content-	REST 请求特定头域。

元素名称	描述
Disposition, Content-Encoding, Expires	支持精确匹配和 starts-with 条件匹配。
key	上传对象的名字。 支持精确匹配和 starts-with 条件匹配。
bucket	请求桶名。 支持精确匹配。
success_action_redirect	上传对象成功后重定向的 URL 地址。具体描述请参见 5.4.2 POST 上传。 支持精确匹配和 starts-with 条件匹配。
success_action_status	如果未指定 success_action_redirect，则成功上传时返回给客户端的状态码。具体描述请参见 5.4.2 POST 上传。 支持精确匹配。
x-obs-meta-*	用户自定义元数据。 元素中的关键字不允许含有非 ASCII 码或不可识别字符，如果一定要使用非 ASCII 码或不可识别字符，需要客户端自行做编解码处理，可以采用 URL 编码或者 Base64 编码，服务端不会做解码处理。 支持精确匹配和 starts-with 条件匹配。
x-obs-*	其他以 x-obs-为前缀的头域。 支持精确匹配和 starts-with 条件匹配。
x-obs-security-token	请求消息头中字段名。 临时 AK/SK 和 securitytoken 鉴权必加字段名。

Policy 条件匹配的方式如下：

表 3-17 policy 条件匹配方式

条件	描述
Exact Matches	默认是完全匹配，post 表单中该项的值必须和 policy 的 conditions 中设置的值完全一样。例如：上传对象的同时设置对象 ACL 为 public-read，表单中 x-obs-acl 元素的值为 public-read，policy 中的 conditions 可以设置为 {"x-obs-acl": "public-read" }或者["eq", "$x-obs-acl", "public-read"]，这两者是等效的。
Starts With	如果使用该条件，则 post 表单中对应元素的值必须是固定字符串开始。例如：上传对象名以 user/为前缀，表单中 key 元素的值可以是 user/test1、user/test2，policy 的 conditions 中该条件如下： ["starts-with", "$key", "user/"]

条件	描述
Matching Any Content	post 表单中对应元素的值可以是任意值。例如：请求成功后重定向的地址可以是任意地址，表单中 success_action_redirect 元素的值可以是任意值，policy 的 conditions 中该条件如下： ["starts-with", "$success_action_redirect", ""]
Specifying Ranges	post 表单中 file 元素文件的内容长度可以是一个指定的范围，只用于限制对象大小。例如上传对象大小为 1-10MB，表单中 file 元素的内容长度可以是 1048576-10485760，policy 的 conditions 中该条件如下，注意值没有双引号： ["content-length-range", 1048576, 10485760]

说明

policy 使用 json 格式，conditions 可以支持 { } 和 [ ] 两种方式，{ }中包含表单元素的 key 和 value 两项，以冒号分隔；[ ]中包含条件类型、key、value 三项，以逗号分隔，元素 key 之前使用$字符表示变量。

Policy 中必须转义的字符如下：

表 3-18 policy 中必须转义的字符

转义后的字符	真实字符
\\	反斜杠(\)
\$	美元符号($)
\b	退格
\f	换页
\n	换行
\r	回车
\t	水平制表
\v	垂直制表
\u*xxxx*	所有 Unicode 字符

请求和 Policy 示例

下面的几张表提供了一些请求和 Policy 的例子。

示例 1：在 examplebucket 桶中上传 testfile.txt 对象，并且设置对象 ACL 为公共可读。

请求	*policy*

请求	*policy*

POST / HTTP/1.1
Host: examplebucket.obs.*region*.example.com
Content-Type: multipart/form-data; boundary=7e32233530b26
Content-Length: 1250
--7e32233530b26
Content-Disposition: form-data; name="key"
testfile.txt
--7e32233530b26
Content-Disposition: form-data; name="x-obs-acl"
public-read
--7e32233530b26
Content-Disposition: form-data; name="content-type"
text/plain
--7e32233530b26
Content-Disposition: form-data; name="AccessKeyId"
UDSIAMSTUBTEST000002
--7e32233530b26
Content-Disposition: form-data; name="policy"
ewogICJleHBpcmF0aW9uIjogIjIwMTktMDctMDFUMTI6MDA6MDAuMDAwWiIsCiAgImNvbmRpdGlvbnMiOiBbCiAgICB7ImJ1Y2tldCI6ICJleGFtcGxlYnVja2V0IiB9LAogICAgWyJlcSIsICIka2V5IiwgInRlc3RmaWxlLnR4dCJdLAogICAgeyJ4LW9icy1hY2wiOiAicHVibGljLXJlYWQiIH0sCiAgICBbImVxIiwgIiRDb250ZW50LVR5cGUiLCAidGV4dC9wbGFpbiJdLAogICAgWyJjb250ZW50LWxlbmd0aC1yYW5nZSIsIDYsIDEwXQogIF0KfQo=
--7e32233530b26
Content-Disposition: form-data; name="Signature"
xxl7bZs/5FgtBUggOdQ88DPZUo0=
--7e32233530b26
Content-Disposition: form-data; name="file"; filename="E:\TEST_FILE\TEST.txt"
	{
	"expiration": "2019-07-01T12:00:00.000Z",
	"conditions": [
	{"bucket": "examplebucket" },
	["eq", "$key", "testfile.txt"],
	{"x-obs-acl": "public-read" },
	["eq", "$Content-Type", "text/plain"]
	]
	}

请求	*policy*
Content-Type: text/plain 123456 --7e32233530b26 Content-Disposition: form-data; name="submit" Upload --7e32233530b26--	

示例 2：在 examplebucket 桶中上传 file/obj1 对象，并且设置对象的四个自定义元数据。

请求	*policy*

POST / HTTP/1.1
Host: examplebucket.obs.*region*.example.com
Content-Type: multipart/form-data; boundary=7e329d630b26
Content-Length: 1597
--7e3542930b26
Content-Disposition: form-data; name="key"
file/obj1
--7e3542930b26
Content-Disposition: form-data; name="AccessKeyId"
UDSIAMSTUBTEST000002
--7e3542930b26
Content-Disposition: form-data; name="policy"
ewogICJleHBpcmF0aW9uIjogIjIwMTktMDctMDFUMTI6MDA6MDAuMDAwWiIsCiAgImNvbmRpdGlvbnMiOiBbCiAgICB7ImJ1Y2tldCI6ICJleGFtcGxlYnVja2V0IiB9LAogICAgWyJzdGFydHMtd2l0aCIsICIka2V5IiwgImZpbGUvIl0sCiAgICB7Ingtb2JzLW1ldGEtdGVzdDEiOiJ2YWx1ZTEifSwKICAgIFsiZXEiLCAiJHgtb2JzLW1ldGEtdGVzdDIiLCAidmFsdWUyIl0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiR4LW9icy1tZXRhLXRlc3QzIiwgImRvYyJdLAogICAgWyJzdGFydHMtd2l0aCIsICIkeC1vYnMtbWV0YS10ZXN0NCIsICIiXQogIF0KfQ==
--7e3542930b26
Content-Disposition: form-data; name="signature"
	{
	"expiration": "2019-07-01T12:00:00.000Z",
	"conditions": [
	{"bucket": "examplebucket" },
	["starts-with", "$key", "file/"],
	{"x-obs-meta-test1":"value1"},
	["eq", "$x-obs-meta-test2", "value2"],
	["starts-with", "$x-obs-meta-test3", "doc"],
	["starts-with", "$x-obs-meta-test4", ""]
	]
	}

请求	policy
HTId8hcaisn6FfdWKqSJP9RN4Oo= --7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test1" value1 --7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test2" value2 --7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test3" doc123 --7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test4" my --7e3542930b26 Content-Disposition: form-data; name="file"; filename="E:\TEST_FILE\TEST.txt" Content-Type: text/plain 123456 --7e3542930b26 Content-Disposition: form-data; name="submit" Upload --7e3542930b26--	

3.3 返回结果

请求发送以后，您会收到响应，包含状态码、响应消息头和消息体。

状态码

状态码是一组从 2xx（成功）到 4xx 或 5xx（错误）的数字代码，状态码表示了请求响应的状态，完整的状态码列表请参见 8.1 状态码。

响应消息头

对应请求消息头，响应同样也有消息头，如“Content-Length”。

详细的公共响应消息头字段请参见表 3-19。

表 3-19 公共响应消息头

消息头名称	描述
Content-Length	响应消息体的字节长度。 类型：String 默认值：无。
Connection	指明与服务器的连接是长连接还是短连接。 类型：String 有效值：keep-alive | close。 默认值：无。
Date	OBS 系统响应的时间。 类型：String 默认值：无。
ETag	ETag 是对象内容的唯一标识，可以通过该值识别对象内容是否有变化。比如上传对象时 ETag 为 A，下载对象时 ETag 为 B，则说明对象内容发生了变化。当使用 PUT 上传时，ETag 是对象的 MD5 哈希值。如果通过多段上传对象，则无论加密方法如何，MD5 会拆分 ETag，此类情况 ETag 就不是对象的 MD5 哈希值。ETag 只反映变化的内容，而不是其元数据。上传的对象或拷贝操作创建的对象，通过 MD5 加密后都有唯一的 ETag。 类型：String
x-obs-id-2	帮助定位问题的特殊符号。 类型：String 默认值：无。
x-reserved-indicator	帮助定位问题的特殊符号。 类型：String 默认值：无。
x-obs-request-id	由 OBS 创建来唯一确定本次请求的值，可以通过该值来定位问题。 类型：String 默认值：无。

响应消息体（可选）

响应消息体通常以结构化格式（如 JSON 或 XML）返回，与响应消息头中 Content-type 对应，传递除响应消息头之外的内容。

4 快速入门

4.1 创建桶

操作场景

桶是 OBS 中存储对象的容器。您需要先创建一个桶，然后才能在 OBS 中存储数据。

下面介绍如何调用 5.1.2 创建桶 API 在指定的区域创建一个桶，API 的调用方法请参见 3 如何调用 API。

前提条件

- 已获取 AK 和 SK，获取方法参见 8.2 获取访问密钥（AK/SK）。
- 您需要规划桶所在的区域信息，并根据区域确定调用 API 的 Endpoint，您可以向企业管理员获取区域和终端节点信息。

区域一旦确定，创建完成后无法修改。

在 a1 区域创建一个名为 bucket001 的桶

示例中使用通用的 Apache Http Client。

```
package com.obsclient;

import java.io.*;

import org.apache.http.Header;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpPut;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;

public class TestMain {
    /* 认证用的 ak 和 sk 硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
```

```
   本示例以 ak 和 sk 保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量 YOUR_AK 和
YOUR_SK。*/
   public static String accessKey = System.getenv("YOUR_AK"); //取值为获取的 AK
   public static String securityKey = System.getenv("YOUR_SK");  //取值为获取的 SK
   public static String region = "a1"; // 取值为规划桶所在的区域
   public static String createBucketTemplate =
         "<CreateBucketConfiguration " +
         "xmlns=\"http://obs.a1.example.com/doc/2015-06-30/\">\n" +
         "<Location>" + region + "</Location>\n" +
         "</CreateBucketConfiguration>";

   public static void main(String[] str) {

       createBucket();

   }

   private static void createBucket() {
      CloseableHttpClient httpClient = HttpClients.createDefault();
      String requesttime = DateUtils.formatDate(System.currentTimeMillis());
      String contentType = "application/xml";
      HttpPut httpPut = new HttpPut("http://bucket001.obs.a1.example.com");
      httpPut.addHeader("Date", requesttime);
      httpPut.addHeader("Content-Type", contentType);

      /** 根据请求计算签名**/
      String contentMD5 = "";
      String canonicalizedHeaders = "";
      String canonicalizedResource = "/bucket001/";
      // Content-MD5 、Content-Type 没有直接换行， data 格式为 RFC 1123，和请求中的时间一致
      String canonicalString = "PUT" + "\n" + contentMD5 + "\n" + contentType +
"\n" + requesttime + "\n" + canonicalizedHeaders + canonicalizedResource;
      System.out.println("StringToSign:[" + canonicalString + "]");
      String signature = null;
      CloseableHttpResponse httpResponse = null;
      try {
         signature = Signature.signWithHmacSha1(securityKey, canonicalString);

         // 增加签名头域 Authorization: OBS AccessKeyID:signature
         httpPut.addHeader("Authorization", "OBS " + accessKey + ":" + signature);

         // 增加 body 体
         httpPut.setEntity(new StringEntity(createBucketTemplate));

         httpResponse = httpClient.execute(httpPut);

         // 打印发送请求信息和收到的响应消息
         System.out.println("Request Message:");
         System.out.println(httpPut.getRequestLine());
         for (Header header : httpPut.getAllHeaders()) {
            System.out.println(header.getName() + ":" + header.getValue());
         }

         System.out.println("Response Message:");
         System.out.println(httpResponse.getStatusLine());
```

```
            for (Header header : httpResponse.getAllHeaders()) {
               System.out.println(header.getName() + ":" + header.getValue());
            }
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                  httpResponse.getEntity().getContent()));

            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = reader.readLine()) != null) {
               response.append(inputLine);
            }
            reader.close();

            // print result
            System.out.println(response.toString());
         } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
         } catch (IOException e) {
            e.printStackTrace();
         } finally {
            try {
               httpClient.close();
            } catch (IOException e) {
               e.printStackTrace();
            }
         }

      }

}
```

其中 Date 头域 DateUtils 的格式为：

```
package com.obsclient;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Locale;
import java.util.TimeZone;

public class DateUtils {

   public static String formatDate(long time)
   {
      DateFormat serverDateFormat = new SimpleDateFormat("EEE, dd MMM yyyy
HH:mm:ss z", Locale.ENGLISH);
      serverDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
      return serverDateFormat.format(time);
   }
}
```

签名字符串 Signature 的计算方法为：

```
package com.obsclient;

import javax.crypto.Mac;
```

```
import javax.crypto.spec.SecretKeySpec;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;
import java.security.InvalidKeyException;
import java.util.Base64;

public class Signature {
   public static String signWithHmacSha1(String sk, String canonicalString) throws
UnsupportedEncodingException {

      try {
         SecretKeySpec signingKey = new SecretKeySpec(sk.getBytes("UTF-8"),
"HmacSHA1");
         Mac mac = Mac.getInstance("HmacSHA1");
         mac.init(signingKey);
         return
Base64.getEncoder().encodeToString(mac.doFinal(canonicalString.getBytes("UTF-8")));
      } catch (NoSuchAlgorithmException | InvalidKeyException |
UnsupportedEncodingException e) {
         e.printStackTrace();
      }
      return null;
   }
}
```

4.2 获取桶列表

操作场景

如果用户想要查看自己创建的所有桶信息，可以使用获取桶列表接口查看。

下面介绍如何调用 5.1.1 获取桶列表 API，API 的调用方法请参见 3 如何调用 API。

前提条件

- 已获取 AK 和 SK，获取方法参见 8.2 获取访问密钥（AK/SK）。
- 您需要明确需要列举的桶所在的区域信息，并根据区域确定调用 API 的 Endpoint，您可以向企业管理员获取区域和终端节点信息。

获取 *a1* 区域的桶列表

示例中使用通用的 Apache Http Client。

```
package com.obsclient;

import java.io.*;
import java.util.ArrayList;
import java.util.List;

import org.apache.http.Header;
import org.apache.http.HttpEntity;
```

```
import org.apache.http.NameValuePair;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.methods.HttpPut;
import org.apache.http.entity.InputStreamEntity;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.message.BasicNameValuePair;


public class TestMain {

   /* 认证用的 ak 和 sk 硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
   本示例以 ak 和 sk 保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量 YOUR_AK 和 YOUR_SK。*/
   public static String accessKey = System.getenv("YOUR_AK"); //取值为获取的 AK
   public static String securityKey = System.getenv("YOUR_SK");  //取值为获取的 SK

   public static void main(String[] str) {

      listAllMyBuckets();

   }


   private static void listAllMyBuckets() {
      CloseableHttpClient httpClient = HttpClients.createDefault();
      String requesttime = DateUtils.formatDate(System.currentTimeMillis());
      HttpGet httpGet = new HttpGet("http://obs.a1.example.com");
      httpGet.addHeader("Date", requesttime);

      /** 根据请求计算签名**/
      String contentMD5 = "";
      String contentType = "";
      String canonicalizedHeaders = "";
      String canonicalizedResource = "/";
      // Content-MD5 、Content-Type 没有直接换行， data 格式为 RFC 1123，和请求中的时间一致
      String canonicalString = "GET" + "\n" + contentMD5 + "\n" + contentType + "\n" + requesttime + "\n" + canonicalizedHeaders + canonicalizedResource;
      System.out.println("StringToSign:[" + canonicalString + "]");
      String signature = null;
      try {
         signature = Signature.signWithHmacSha1(securityKey, canonicalString);

         // 增加签名头域 Authorization: OBS AccessKeyID:signature
         httpGet.addHeader("Authorization", "OBS " + accessKey + ":" + signature);
         CloseableHttpResponse httpResponse = httpClient.execute(httpGet);

         // 打印发送请求信息和收到的响应消息
         System.out.println("Request Message:");
         System.out.println(httpGet.getRequestLine());
```

```
            for (Header header : httpGet.getAllHeaders()) {
               System.out.println(header.getName() + ":" + header.getValue());
            }

            System.out.println("Response Message:");
            System.out.println(httpResponse.getStatusLine());
            for (Header header : httpResponse.getAllHeaders()) {
               System.out.println(header.getName() + ":" + header.getValue());
            }
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                  httpResponse.getEntity().getContent()));

            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = reader.readLine()) != null) {
               response.append(inputLine);
            }
            reader.close();
            // print result
            System.out.println(response.toString());
         } catch (UnsupportedEncodingException e) {
            e.printStackTrace();

         } catch (IOException e) {
            e.printStackTrace();
         } finally {
            try {
               httpClient.close();
            } catch (IOException e) {
               e.printStackTrace();
            }
         }

      }

}
```

其中 Date 头域 DateUtils 的格式为：

```
package com.obsclient;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Locale;
import java.util.TimeZone;

public class DateUtils {

   public static String formatDate(long time)
   {
      DateFormat serverDateFormat = new SimpleDateFormat("EEE, dd MMM yyyy
HH:mm:ss z", Locale.ENGLISH);
      serverDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
      return serverDateFormat.format(time);
```

```
    }
}
```

签名字符串 Signature 的计算方法为：

```
package com.obsclient;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;
import java.security.InvalidKeyException;
import java.util.Base64;

public class Signature {
   public static String signWithHmacSha1(String sk, String canonicalString) throws
UnsupportedEncodingException {

      try {
         SecretKeySpec signingKey = new SecretKeySpec(sk.getBytes("UTF-8"),
"HmacSHA1");
         Mac mac = Mac.getInstance("HmacSHA1");
         mac.init(signingKey);
         return
Base64.getEncoder().encodeToString(mac.doFinal(canonicalString.getBytes("UTF-8")));
      } catch (NoSuchAlgorithmException | InvalidKeyException |
UnsupportedEncodingException e) {
         e.printStackTrace();
      }
      return null;
   }
}
```

4.3 上传对象

操作场景

您可以根据需要，将任何类型的文件上传到 OBS 桶中进行存储。

下面介绍如何调用 5.4.1 PUT 上传 API 在指定的桶中上传对象，API 的调用方法请参见 3 如何调用 API。

前提条件

- 已获取 AK 和 SK，获取方法参见 8.2 获取访问密钥（AK/SK）。
- 已创建了至少一个可用的桶。
- 已准备好了待上传的文件，并清楚文件所在的本地完整路径。
- 您需要知道待上传桶所在的区域信息，并根据区域确定调用 API 的 Endpoint，您可以向企业管理员获取区域和终端节点信息。

向 a1 区域的桶 bucket001 中上传对象，名称为 objecttest1

示例中使用通用的 Apache Http Client。

说明

本示例代码中的“contentType”参数为空，那么计算签名时，签名头域中的该参数也应为空，否则会报签名不匹配错误。

```
package com.obsclient;

import java.io.*;
import java.util.ArrayList;
import java.util.List;

import org.apache.http.Header;
import org.apache.http.HttpEntity;
import org.apache.http.NameValuePair;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.methods.HttpPut;
import org.apache.http.entity.InputStreamEntity;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.message.BasicNameValuePair;


public class TestMain {

    /* 认证用的 ak 和 sk 硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
    本示例以 ak 和 sk 保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量 YOUR_AK 和 YOUR_SK。*/
    public static String accessKey = System.getenv("YOUR_AK"); //取值为获取的 AK
    public static String securityKey = System.getenv("YOUR_SK");  //取值为获取的 SK

    public static void main(String[] str) {

        putObjectToBucket();

    }


    private static void putObjectToBucket() {

        InputStream inputStream = null;
        CloseableHttpClient httpClient = HttpClients.createDefault();
        CloseableHttpResponse httpResponse = null;
        String requestTime = DateUtils.formatDate(System.currentTimeMillis());
        HttpPut httpPut = new HttpPut("http://bucket001.obs.a1.example.com/objecttest1");
        httpPut.addHeader("Date", requestTime);
```

```
        /** 根据请求计算签名 **/
        String contentMD5 = "";
        String contentType = "";
        String canonicalizedHeaders = "";
        String canonicalizedResource = "/bucket001/objecttest1";
        // Content-MD5 、Content-Type 没有直接换行， data 格式为 RFC 1123，和请求中的时间一致
        String canonicalString = "PUT" + "\n" + contentMD5 + "\n" + contentType +
"\n" + requestTime + "\n" + canonicalizedHeaders + canonicalizedResource;
        System.out.println("StringToSign:[" + canonicalString + "]");
        String signature = null;
        try {
            signature = Signature.signWithHmacSha1(securityKey, canonicalString);
            // 上传的文件目录
            inputStream = new FileInputStream("D:\\OBSobject\\text01.txt");
            InputStreamEntity entity = new InputStreamEntity(inputStream);
            httpPut.setEntity(entity);

            // 增加签名头域 Authorization: OBS AccessKeyID:signature
            httpPut.addHeader("Authorization", "OBS " + accessKey + ":" + signature);
            httpResponse = httpClient.execute(httpPut);

            // 打印发送请求信息和收到的响应消息
            System.out.println("Request Message:");
            System.out.println(httpPut.getRequestLine());
            for (Header header : httpPut.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }

            System.out.println("Response Message:");
            System.out.println(httpResponse.getStatusLine());
            for (Header header : httpResponse.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    httpResponse.getEntity().getContent()));

            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = reader.readLine()) != null) {
                response.append(inputLine);
            }
            reader.close();

            // print result
            System.out.println(response.toString());


        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();

        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
```

```
            httpClient.close();
         } catch (IOException e) {
            e.printStackTrace();
         }
      }
   }

}
```

其中 Date 头域 DateUtils 的格式为：

```
package com.obsclient;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Locale;
import java.util.TimeZone;

public class DateUtils {

   public static String formatDate(long time)
   {
      DateFormat serverDateFormat = new SimpleDateFormat("EEE, dd MMM yyyy
HH:mm:ss z", Locale.ENGLISH);
      serverDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
      return serverDateFormat.format(time);
   }
}
```

签名字符串 Signature 的计算方法为：

```
package com.obsclient;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;
import java.security.InvalidKeyException;
import java.util.Base64;

public class Signature {
   public static String signWithHmacSha1(String sk, String canonicalString) throws
UnsupportedEncodingException {

      try {
         SecretKeySpec signingKey = new SecretKeySpec(sk.getBytes("UTF-8"),
"HmacSHA1");
         Mac mac = Mac.getInstance("HmacSHA1");
         mac.init(signingKey);
         return
Base64.getEncoder().encodeToString(mac.doFinal(canonicalString.getBytes("UTF-8")));
      } catch (NoSuchAlgorithmException | InvalidKeyException |
UnsupportedEncodingException e) {
         e.printStackTrace();
      }
      return null;
```

```
    }
}
```

5 API

5.1 桶的基础操作

5.1.1 获取桶列表

功能介绍

OBS 用户可以通过请求查询自己创建的所有区域的桶列表。

接口约束

- 终端节点（Endpoint）不会限制查询结果，无论哪一个区域的 Endpoint，查询结果都是所有区域的桶列表。
- 创建桶时，请勿并发列举桶。

请求消息样式

```
GET / HTTP/1.1
Host: obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不带请求参数。

请求消息头

该请求消息头使用公共消息字段，具体请参见表 3-3。

该操作消息头与普通请求一样，请参见表 5-1，但可以带附加消息头，附加请求消息头如下所示。

表 5-1 附加请求消息头

参数	是否必选	参数类型	描述
x-obs-bucket-type	否	String	**参数解释：** 此消息头明确获取桶列表的内容，即列表中包含什么类型的桶。 示例：x-obs-bucket-type: POSIX **约束限制：** 不带此消息头则获取所有桶和并行文件系统列表。 **取值范围：** • OBJECT：获取所有对象桶列表。 • POSIX：获取所有并行文件系统列表。 **默认取值：** 不带请求头场景：默认取值为空，表示 OBJECT 和 POSIX 同时携带。

请求消息元素

该请求消息中不带请求元素。

响应消息样式

```
GET HTTP/1.1 status_code
Content-Type: type
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListAllMyBucketsResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Owner>
      <ID>id</ID>
   </Owner>
   <Buckets>
      <Bucket>
         <Name>bucketName</Name>
         <CreationDate>date</CreationDate>
         <Location>region</Location>
      </Bucket>
      ...
   </Buckets>
</ListAllMyBucketsResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中，会以 XML 形式将用户拥有的桶列出来，元素的具体含义如表 5-2 所示。

表 5-2 响应消息元素

参数	参数类型	描述
ListAllMyBucketsResult	XML	**参数解释：** 用户的桶列表，是表 5-3 和表 5-4 的父节点。 **取值范围：** 不涉及
表 5-3	XML	**参数解释：** 桶拥有者信息。 **取值范围：** 请详见表 5-3。
表 5-4	XML	**参数解释：** 本次列举返回的桶列表。 **取值范围：** 请详见表 5-4。

表 5-3 Owner 参数说明

参数	参数类型	描述
ID	String	**参数解释：** 桶拥有者的 DomainID（账号 ID）。 **取值范围：** 请详见 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。

表 5-4 Buckets 参数说明

参数	参数类型	描述
表 5-5	XML	**参数解释：** 具体的桶信息，包含桶名、桶创建时间等参数。 **取值范围：** 请详见表 5-5。

表 5-5 Bucket 参数说明

参数	参数类型	描述

参数	参数类型	描述
Name	String	**参数解释：** 桶名。 **取值范围：** 长度为 3～63 的字符串。
CreationDate	String	**参数解释：** 桶的创建时间（UTC 时间）。日期格式为 ISO8601 的格式，例如：2025-06-28T08:57:41.047Z。 **取值范围：** 长度为 24 的字符串。
Location	String	**参数解释：** 桶所在的区域。

错误响应消息

该请求无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /
 HTTP/1.1
User-Agent: curl/7.29.0
Host: obs.region.example.com
Accept: */*
Date: Mon, 25 Jun 2018 05:37:12 +0000
Authorization: OBS GKDF4C7Q6SI0IPGTXTJN:9HXkVQIiQKw33UEmyBI4rWrzmic=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435722C11379647A8A00A
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSGGDRUM62QZi3hGP8Fz3gOloYCfZ39U
Content-Type: application/xml
Date: Mon, 25 Jun 2018 05:37:12 GMT
Content-Length: 460

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListAllMyBucketsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>783fc6652cf246c096ea836694f71855</ID>
  </Owner>
  <Buckets>
    <Bucket>
      <Name>examplebucket01</Name>
      <CreationDate>2018-06-21T09:15:01.032Z</CreationDate>
      <Location>region</Location>
```

```
    </Bucket>
    <Bucket>
      <Name>examplebucket02</Name>
      <CreationDate>2018-06-22T03:56:33.700Z</CreationDate>
      <Location>region</Location>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

5.1.2 创建桶

功能介绍

创建桶是指按照用户指定的桶名创建一个新桶的操作。

说明

- 默认情况下，一个用户可以拥有的桶的数量不能超过 100 个。
- 因为 OBS 桶是全局资源，所以您自有的桶不能重名，您与其他用户的桶也不能重名。删桶后桶名会释放，释放完成后才能创建同名桶。调用删桶 API 一般耗时约 30 分钟释放，如遇特殊情况（例如欠费、销户触发删桶，删桶前需要先进行资源清理）耗时会更长，OBS 无法保证此过程的时效性。因此创建同名桶失败，可能是由于桶名未释放完成。另外，如果桶名释放后，有其他用户在您之前抢先使用了桶名，创建同名桶也会失败。
- OBS 支持在创建桶时指定桶的 AZ 类型，您可以开启或关闭多 AZ。关闭多 AZ 时，桶内数据默认存储在单个 AZ 内；开启多 AZ 时，桶内数据冗余存储在多个 AZ 内，可靠性更高。旧桶 AZ 类型默认为单 AZ。
- OBS 支持在创建桶时打开桶级 WORM 开关，打开后可以为桶内的对象设置 WORM，详见 5.2.37 配置桶级默认 WORM 策略。在打开 WORM 开关的时候会默认为桶开启多版本，且多版本状态无法关闭；您无法为一个桶同时开启 WORM 开关和并行文件系统。

新创建桶的桶名在 OBS 中必须是唯一的。如果是同一个用户重复创建同一区域的同名桶时返回成功。除此以外的其他场景重复创建同名桶返回桶已存在。用户可以在请求消息头中加入 x-obs-acl 等参数，设置要创建桶的权限控制策略。

存储类型

允许用户创建不同默认存储类型的桶。发送创建桶请求时携带头域“x-obs-storage-class”来指定桶的默认存储类型。桶内对象的存储类型与桶默认存储类型保持一致。存储类型有：STANDARD（标准存储）、WARM（低频访问存储）、COLD（归档存储）。如果没有携带此头域 ，则创建的桶为标准存储类型。

当往桶内上传对象时，如果没有指定对象的存储类别（参考 5.4.1 PUT 上传），则该对象的存储类型取桶的默认存储类型。

- OBS 标准存储拥有低访问时延和较高的吞吐量，因而适用于有大量热点文件需要频繁访问数据的业务场景，例如：大数据、移动应用、热点视频、社交图片等场景。
- OBS 低频访问存储适用于不频繁访问（少于每月一次访问）但在需要时也要求快速访问数据的业务场景，例如：文件同步/共享、企业备份等场景。与标准存储相比，低频访问存储有相同的数据持久性、吞吐量以及访问时延，且成本较低，但是可用性略低于标准存储。

- OBS 归档存储适用于很少访问（平均一年访问一次）数据的业务场景，例如：数据归档、长期备份等场景。归档存储安全、持久且成本极低，可以用来替代磁带库。为了保持成本低廉，数据恢复时间可能长达数分钟到数小时不等。

请求消息样式

```
PUT / HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
x-obs-az-redundancy: 3az

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>location</Location>
</CreateBucketConfiguration>
```

请求消息参数

该请求消息中不带请求参数。

请求消息头

该操作消息头与普通请求一样，请参见表 3-3，但可以带附加消息头，附加请求消息头如下所示。注意，**桶名在表 3-3 的 Host 写明**。

表 5-6 附加请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-acl	String	否	**参数解释：** 创建桶时，可以加上此消息头设置桶的权限控制策略，使用的策略为预定义的 ACL 策略。 **取值范围：** • private：桶或对象的所有者拥有完全控制的权限，其他任何人都没有访问权限。 • public-read：设置在桶上时，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据。 • public-read-write：设在桶上，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据、上传对象、删除对象、初始化段任务、上传段、合并段、拷贝段、取消多段上传任务。 • public-read-delivered：设在桶上，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据，可以获取该桶内对象的内容和元数据。 • public-read-write-delivered：设在桶上，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据、上传对象、删除对象、初始化段任务、上

消息头名称	消息头类型	是否必选	描述
			传段、合并段、拷贝段、取消多段上传任务，可以获取该桶内对象的内容和元数据。 • bucket-owner-full-control：设在对象上，桶和对象的所有者拥有对象的完全控制权限，其他任何人都没有访问权限。 默认情况下，上传对象至其他用户的桶中，桶拥有者没有对象的控制权限。对象拥有者为桶拥有者添加此权限控制策略后，桶所有者可以完全控制对象。 例如，用户 A 上传对象 x 至用户 B 的桶中，系统默认用户 B 没有对象 x 的控制权。当用户 A 为对象 x 设置 bucket-owner-full-control 策略后，用户 B 就拥有了对象 x 的控制权。 **默认取值：** private
x-obs-storage-class	String	否	**参数解释：** 创建桶时，可以加上此消息头设置桶的默认存储类型。 **取值范围：** • STANDARD（标准存储） • WARM（低频访问存储） • COLD（归档存储） **默认取值：** STANDARD
x-obs-grant-read	String	否	**参数解释：** 授权 READ 权限给指定 domain 下的所有用户。允许列举桶内对象、列举桶中多段任务、列举桶中多版本对象、获取桶元数据。 示例：x-obs-grant-read:id=租户 id **约束限制：** 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-write	String	否	**参数解释：** 授权 WRITE 权限给指定 domain 下的所有用户，允

消息头名称	消息头类型	是否必选	描述
			许创建、删除、覆盖桶内所有对象，允许初始化段、上传段、拷贝段、合并段、取消多段上传任务。 示例：x-obs-grant-write:id=租户 id **约束限制：** 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-read-acp	String	否	**参数解释：** 授权 READ_ACP 权限给指定 domain 下的所有用户，允许读桶的 ACL 信息。 示例：x-obs-grant-read-acp:id=租户 id **约束限制：** 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-write-acp	String	否	**参数解释：** 授权 WRITE_ACP 权限给指定 domain 下的所有用户，允许修改桶的 ACL 信息。 示例：x-obs-grant-write-acp:id=租户 id **约束限制：** 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-full-control	String	否	**参数解释：** 授权 FULL_CONTROL 权限给指定 domain 下的所有用户。 示例：x-obs-grant-full-control:id=租户 id

消息头名称	消息头类型	是否必选	描述
			约束限制： 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-read-delivered	String	否	**参数解释：** 授权 READ 权限给指定 domain 下的所有用户，并且在默认情况下，该 READ 权限将传递给桶内所有对象。 示例：x-obs-grant-read-delivered:id=租户 id **约束限制：** 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-full-control-delivered	String	否	**参数解释：** 授权 FULL_CONTROL 权限给指定 domain 下的所有用户，并且在默认情况下，该 FULL_CONTROL 权限将传递给桶内所有对象。 示例：x-obs-grant-full-control-delivered:id=租户 id **约束限制：** 无 **取值范围：** id=租户 id，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-az-redundancy	String	否	**参数解释：** 创建桶时带上此消息头设置桶的数据冗余策略。 **约束限制：** 用户携带该头域指定新创的桶的数据冗余策略，存在一种情况是当该区域如果不支持多 AZ 存储，则该桶的存储类型仍为单 AZ。

消息头名称	消息头类型	是否必选	描述
			取值范围： • 3az：多 AZ **默认取值：** 不指定头域，默认为单 AZ。
x-obs-fs-file-interface	String	否	**参数解释：** 创建桶时可以带上此消息头以创建并行文件系统。 示例：x-obs-fs-file-interface:Enabled **取值范围：** Enabled **默认取值：** 指定头域时必须为 Enabled，无默认取值。
x-obs-epid	String	否	**参数解释：** 企业项目 ID。开通企业项目的用户可以从企业项目服务获取，格式为 uuid，默认项目传“0”或者不带该头域，未开通企业项目的用户可以不带该头域。 示例：x-obs-epid:9892d768-2d13-450f-aac7-ed0e44c2585f **约束限制：** 无 **取值范围：** 填写正确的企业项目 ID 或者为 ID 为 0。 **默认取值：** 0
x-obs-bucket-object-lock-enabled	String	否	**参数解释：** 创建桶时可以带上此消息头来开启 WORM 开关。 示例：x-obs-bucket-object-lock-enabled:true **约束限制：** 只支持对象桶。 **取值范围：** • true：开启 WORM 开关 **默认取值：** 指定头域为 true 或 True 时，开启桶的对象锁策略；如果不指定头域时，不开启对象锁策略。

请求消息元素

该操作可以带附加请求消息元素，附加请求消息元素的具体描述如表 5-7 所示。

表 5-7 附加请求消息元素

元素名称	元素类型	是否必选	描述
Location	String	否	**参数解释：** 指定 Bucket 在哪个区域被创建。 • 使用默认区域的终端节点创建桶时 – 不携带 Location，桶将默认创建在默认区域 – 在 Location 中指定其它区域，桶将创建在指定区域 • 使用非默认区域的终端节点创建桶时，必须携带 Location，并且 Location 只能指定为该终端节点对应的区域。 **取值范围：** 请向企业管理员获取区域和终端节点信息。

响应消息样式

```
HTTP/1.1 status_code
Location: location
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：创建桶

```
PUT / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
```

```
Content-Length: 157

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>region</Location>
</CreateBucketConfiguration>
```

响应示例：创建桶

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

请求示例：创建指定 ACL 和存储类型的桶

```
PUT / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
x-obs-acl:public-read
x-obs-storage-class:STANDARD
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 157

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>region</Location>
</CreateBucketConfiguration>
```

响应示例：创建指定 ACL 和存储类型的桶

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

请求示例：创建桶时选择 AZ

```
PUT / HTTP/1.1
Host: examplebucket.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
x-obs-az-redundancy: 3az
<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
 <Location>region</Location>
</CreateBucketConfiguration>
```

响应示例：创建桶时选择 AZ

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

请求示例：创建并行文件系统

```
PUT / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 157
x-obs-fs-file-interface: Enabled

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Location>region</Location>
</CreateBucketConfiguration>
```

响应示例：创建并行文件系统

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

请求示例：创建桶时打开 WORM 开关

```
PUT / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
x-obs-bucket-object-lock-enabled:true
Content-Length: 0
```

响应示例：创建桶时打开 WORM 开关

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 00000184C11AC7A6809F881341842C02
x-reserved-indicator: Unauthorized
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
```

```
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

5.1.3 列举桶内对象

功能介绍

对桶拥有读权限的用户可以执行获取桶内对象列表的操作。

如果用户在请求中只指定了桶名，则返回信息中会包含桶内部分或所有对象的描述信息（一次最多返回 1000 个对象信息）；如果用户还指定了 prefix、marker、max-keys、delimiter 参数中的一个或多个，则返回的对象列表将按照如表 5-8 所示规定的语义返回指定的对象。

用户也可以请求参数中添加 versions 参数来执行列举桶内多版本对象的操作。

请求消息样式

```
GET / HTTP/1.1

Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息样式（多版本）

```
GET /?versions HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求可以通过带参数，列举出桶内的一部分对象，参数的具体含义如表 5-8 所示。

表 5-8 请求消息参数

参数名称	参数类型	是否必选	描述
prefix	String	否	**参数解释：** 列举桶内对象列表时，指定一个前缀，限定返回的对象名必须带有 prefix 前缀。 **约束限制：** 满足对象名的格式。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无

参数名称	参数类型	是否必选	描述
marker	String	否	**参数解释：** 列举桶内对象列表时，指定一个标识符，作为列举时的起始位置，从该标识符以后按对象名的字典顺序返回对象列表。 **约束限制：** 该字段仅用于非多版本列举。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
max-keys	Integer	否	**参数解释：** 列举对象的最大数目，返回的对象列表将是按照字典顺序的最多前 max_keys 个对象。 **约束限制：** 无 **取值范围：** 1~1000，当超出范围时，按照默认的 1000 进行处理。 **默认取值：** 1000
delimiter	String	否	**参数解释：** delimiter 将对象名进行分组的分隔符。表 5-9 所有具有相同前缀的对象名（最多 1,000 个）会被分组在一起。仅当您指定了分隔符（delimiter）时，响应中才会包含 CommonPrefixes。 当桶中有如下对象时 123/, a1/, abc/。当请求条件为：prefix=, delimiter=/ 结果：commonPrefixes=[123/, a1/, abc/] 说明：没有指定 prefix，从对象名的首字符到第一次出现 delimiter 间具有相同字符串的对象名会被分成一组，形成一条 CommonPrefixes。 当桶中有如下对象时 123/1.txt。当请求条件为：prefix=123/, delimiter=/ 结果：commonPrefixes=[] 说明：如果指定了 prefix，从 prefix 到第一次出现 delimiter 间具有相同字符串的对象名会被分成一组，形成一条 CommonPrefixes。 当桶中有如下对象时 123/a/1.txt。当请求条件

参数名称	参数类型	是否必选	描述
			为： prefix=123/, delimiter=/ 结果：commonPrefixes=[123/a/] **对于并行文件系统，不携带此参数时默认列举是递归列举此目录下所有内容，会列举子目录，当子目录较多时会导致请求响应变慢或请求失败。在大数据场景下（目录层级深、目录下文件多）的列举，建议设置[delimiter=/]，只列举当前目录下的内容，不列举子目录，提高列举效率。** **约束限制：** 无 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
key-marker	String	否	**参数解释：** 列举多版本对象的起始位置，返回的对象列表将是对象名按照字典序排序后该参数以后的所有对象。 **约束限制：** 该字段仅用于多版本列举。 **取值范围：** 上次请求返回体的 NextKeyMarker 值。 **默认取值：** 无
version-id-marker	String	否	**参数解释：** 与 key_marker 配合使用，返回的对象列表将是对象名和版本号按照字典序排序后该参数以后的所有对象（单次返回最大为 1000 个）。key_marker 指定对象名，version_id_marker 指定该对象的具体版本号，两者共同确定对象版本。 **约束限制：** • 该字段只适用于多版本列举场景。 • 如果 version_id_marker 不是 key_marker 的一个版本号，则该参数无效。 **取值范围：** 对象的版本号，即上次请求返回体的 nextVersionIdMarker 值。 **默认取值：**

参数名称	参数类型	是否必选	描述
			无

请求消息头

该请求使用公共的请求消息头，具体如表 3-3 所示。

请求消息元素

该请求消息头中不带消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
x-obs-bucket-location: region
Content-Type: application/xml
Content-Length: length
<Response Body>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中，会以 XML 形式将桶中的对象列出来，元素的具体含义如表 5-9 所示。

表 5-9 响应消息元素

参数名称	参数类型	描述
ListBucketResult	XML	**参数解释：** 桶中对象列表。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Contents	XML	**参数解释：** 对象的元数据信息。 父节点：ListBucketResult **约束限制：**

参数名称	参数类型	描述
		无 **取值范围：** 无 **默认取值：** 无
CommonPrefixes	XML	**参数解释：** 请求中带 delimiter 参数时，返回消息带 CommonPrefixes 分组信息。 父节点：ListBucketResult **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Delimiter	String	**参数解释：** 将对象名进行分组的分隔符。如果指定了 prefix，从 prefix 到第一次出现 delimiter 间具有相同字符串的对象名会被分成一组，形成一条 CommonPrefix；如果没有指定 prefix，从对象名的首字符到第一次出现 delimiter 间具有相同字符串的对象名会被分成一组，形成一条 CommonPrefix。 例如，桶中有 3 个对象，分别为 abcd、abcde、bbcde。如果指定 delimiter 为 d，prefix 为 a，abcd、abcde 会被分成一组，形成一条前缀为 abcd 的 commonPrefix；如果只指定 delimiter 为 d，abcd、abcde 会被分成一组，形成一条前缀为 abcd 的 commonPrefix，而 bbcde 会被单独分成一组，形成一条前缀为 bbcd 的 commonPrefix。 父节点：ListBucketResult **约束限制：** 无 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无

参数名称	参数类型	描述
ETag	String	**参数解释：** 对象的 base64 编码的 128 位 MD5 摘要。ETag 是对象内容的唯一标识，可以通过该值识别对象内容是否有变化。比如上传对象时 ETag 为 A，下载对象时 ETag 为 B，则说明对象内容发生了变化。实际的 ETag 是对象的哈希值。ETag 只反映变化的内容，而不是其元数据。上传的对象或拷贝操作创建的对象，通过 MD5 加密后都有唯一的 ETag。 父节点：ListBucketResult.Contents **约束限制** 当对象是服务端加密的对象时，ETag 值不是对象的 MD5 值，而是通过服务端加密计算出的唯一标识。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无
Type	String	**参数解释：** 对象的类型。 父节点：ListBucketResult.Contents **约束限制：** 非 Normal 对象时返回该参数。 **取值范围：** • NORMAL：普通对象 • APPENDABLE：追加写对象 **默认取值：** 无
ID	String	**参数解释：** 对象拥有者的 DomainId。 父节点：ListBucketResult.Contents.Owner **约束限制：** 无 **取值范围：** 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID **默认取值：**

参数名称	参数类型	描述
		无
IsTruncated	Boolean	**参数解释：** 表明本次请求是否返回了全部结果。 父节点：ListBucketResult **约束限制：** 无 **取值范围：** • true：表示本次没有返回全部结果。 • false：表示本次已经返回了全部结果。 **默认取值：** 无
Key	String	**参数解释：** 对象名。对象名是对象在存储桶中的唯一标识。对象名是对象在桶中的完整路径，路径中不包含桶名。 父节点：ListBucketResult.Contents **约束限制：** 无。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
LastModified	Date	**参数解释：** 对象最近一次被修改的时间（UTC 时间）。 父节点：ListBucketResult.Contents **约束限制：** 日期格式为 ISO8601 的格式。 例如：2018-01-01T00:00:00.000Z，表示最后一次修改时间为 2018-01-01T00:00:00.000Z。 **取值范围：** 无 **默认取值：** 无
Marker	String	**参数解释：** 列举桶内对象列表时，指定一个标识符，

参数名称	参数类型	描述
		作为列举时的起始位置，从该标识符以后按对象名的字典顺序返回对象列表。 例如，您拥有以下对象：test/a、test/b、test/c、test/d。如果您将 test/b 指定为标识符，将返回 test/c、test/d 两个对象。 父节点：ListBucketResult **约束限制：** 无 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
NextMarker	String	**参数解释：** 如果本次没有返回全部结果，响应请求中将包含此字段，用于标明本次请求列举到的最后一个对象。后续请求可以指定 Marker 等于该值来列举剩余的对象。 父节点：ListBucketResult **约束限制：** 无 **取值范围：** 字符串，下一次请求起始位置的对象名。 **默认取值：** 无
MaxKeys	String	**参数解释：** 列举对象的最大数目，返回的对象列表将是按照字典顺序的最多前 max_keys 个对象。 父节点：ListBucketResult **约束限制：** 无 **取值范围：** 1~1000，当超出范围时默认为 1000。 **默认取值：** 1000
Name	String	**参数解释：** 桶名。

参数名称	参数类型	描述
		父节点：ListBucketResult **约束限制：** • 桶的名字需全局唯一，不能与已有的任何桶名称重复，包括其他用户创建的桶。 • 桶命名规则如下： – 3～63 个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。 – 禁止使用 IP 地址。 – 禁止以“-”或“.”开头及结尾。 – 禁止两个“.”相邻（如：“my..bucket”）。 – 禁止“.”和“-”相邻（如：“my-.bucket”和“my.-bucket”）。 • 同一用户在同一个区域多次创建同名桶不会报错，创建的桶属性以第一次请求为准。 **默认取值：** 无
Owner	XML	**参数解释：** 用户信息，包含对象拥有者 DomainId 和对象拥有者名称。 父节点：ListBucketResult.Contents **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
DisplayName	String	**参数解释：** 对象拥有者名称。 父节点：ListBucketResult.Contents.Owner **约束限制：** 无 **取值范围：** 无 **默认取值：** 无

参数名称	参数类型	描述
Prefix	String	**参数解释：** 列举桶内对象列表时，指定一个前缀，限定返回的对象名必须带有 prefix 前缀。 例如，您拥有以下对象：logs/day1、logs/day2、logs/day3 和 ExampleObject.jpg。如果您将 logs/指定为前缀，将返回以字符串“logs/”开头的三个对象。如果您指定空的前缀且请求中没有其他过滤条件，将返回桶中的所有对象。 父节点：ListBucketResult **约束限制：** 为桶内对象的前缀。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
Size	String	**参数解释：** 对象的字节数。 父节点：ListBucketResult.Contents **约束限制：** 无 **取值范围：** 0~48.8TB，单位：字节。 **默认取值：** 无
StorageClass	String	**参数解释：** 对象的存储类型。 父节点：ListBucketResult.Contents **约束限制：** 无 **取值范围：** • STANDARD • WARM • COLD **默认取值：** 无

表 5-10 列举多版本对象响应消息元素

参数名称	参数类型	描述
ListVersionsResult	Container	**参数解释：** 保存列举桶中对象列表（含多版本）请求结果的容器。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Name	String	**参数解释：** 桶名。 父节点：ListVersionsResult **约束限制：** • 桶的名字需全局唯一，不能与已有的任何桶名称重复，包括其他用户创建的桶。 • 桶命名规则如下： – 3～63 个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。 – 禁止使用 IP 地址。 – 禁止以“-”或“.”开头及结尾。 – 禁止两个“.”相邻（如：“my..bucket”）。 – 禁止“.”和“-”相邻（如：“my-.bucket”和“my.-bucket”）。 • 同一用户在同一个区域多次创建同名桶不会报错，创建的桶属性以第一次请求为准。 **取值范围：** 无 **默认取值：** 无
Prefix	String	**参数解释：** 列举桶内对象列表时，指定一个前缀，限定返回的对象名必须带有 prefix 前缀。 例如，您拥有以下对象：logs/day1、

参数名称	参数类型	描述
		logs/day2、logs/day3 和 ExampleObject.jpg。如果您将 logs/指定为前缀，将返回以字符串“logs/”开头的三个对象。如果您指定空的前缀且请求中没有其他过滤条件，将返回桶中的所有对象。 父节点：ListVersionsResult **约束限制：** 为桶内对象的前缀。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
KeyMarker	String	**参数解释：** 列举桶内对象列表时，指定一个标识符，作为列举时的起始位置，从该标识符以后按对象名的字典顺序返回对象列表。 例如，您拥有以下对象：test/a、test/b、test/c、test/d。如果您将 test/b 指定为标识符，将返回 test/c、test/d 两个对象。 父节点：ListVersionsResult **约束限制：** 仅用于多版本列举。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
VersionIdMarker	String	**参数解释：** 表示列举多版本对象的起始位置（versionId 标识），与请求中的该参数对应。 **约束限制：** 仅用于多版本列举。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

参数名称	参数类型	描述
NextKeyMarker	String	**参数解释：** 下次列举多版本对象请求的起始位置。如果本次没有返回全部结果，响应请求中将包含该元素，用于标明接下来请求的 KeyMarker 值。 父节点: ListVersionsResult。 **约束限制：** 无 **取值范围：** 字符串，下一次请求起始位置的对象名。 **默认取值：** 无
NextVersionIdMarker	String	**参数解释：** 下次列举多版本对象请求的起始位置（versionId 标识），与 nextKeyMarker 配合使用。如果本次没有返回全部结果，响应请求中将包含该元素，用于标明接下来请求的 VersionIdMarker 值。 父节点: ListVersionsResult。 **约束限制：** 仅用于多版本列举。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无
MaxKeys	String	**参数解释：** 列举对象的最大数目，返回的对象列表将是按照字典顺序的最多前 MaxKeys 个对象。 父节点：ListVersionsResult **约束限制：** 无 **取值范围：** 1~1000，当超出范围时默认为 1000。 **默认取值：** 1000
IsTruncated	Boolean	**参数解释：**

参数名称	参数类型	描述
		表明本次请求是否返回了全部结果。 父节点：ListVersionsResult **约束限制：** 无 **取值范围：** • true：表示本次没有返回全部结果。 • false：表示本次已经返回了全部结果。 **默认取值：** 无
Version	Container	**参数解释：** 保存版本信息的容器。 父节点：ListVersionsResult **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
DeleteMarker	Container	**参数解释：** 保存删除标记的容器。 父节点：ListVersionsResult **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Key	String	**参数解释：** 对版本对象名。对象名是对象在存储桶中的唯一标识。对象名是对象在桶中的完整路径，路径中不包含桶名。 父节点：ListVersionsResult.Version | ListVersionsResult.DeleteMarker **约束限制：** 无。

参数名称	参数类型	描述
		取值范围： 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
VersionId	String	**参数解释：** 对象的版本号。 父节点：ListVersionsResult.Version | ListVersionsResult.DeleteMarker **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无
IsLatest	Boolean	**参数解释：** 标识对象是否是最新的版本。 父节点：ListVersionsResult.Version | ListVersionsResult.DeleteMarker **约束限制：** 无 **取值范围：** • true：代表是最新版本。 • false：代表非最新版本。 **默认取值：** 无
LastModified	Date	**参数解释：** 对象最近一次被修改的时间（UTC 时间）。 父节点：ListVersionsResult.Version | ListVersionsResult.DeleteMarker **约束限制：** 日期格式为 ISO8601 的格式。 例如：2018-01-01T00:00:00.000Z，表示最后一次修改时间为 2018-01-01T00:00:00.000Z。 **取值范围：** 无

参数名称	参数类型	描述
		默认取值： 无
ETag	String	**参数解释：** 对象的 base64 编码的 128 位 MD5 摘要。ETag 是对象内容的唯一标识，可以通过该值识别对象内容是否有变化。比如上传对象时 ETag 为 A，下载对象时 ETag 为 B，则说明对象内容发生了变化。实际的 ETag 是对象的哈希值。ETag 只反映变化的内容，而不是其元数据。上传的对象或拷贝操作创建的对象，通过 MD5 加密后都有唯一的 ETag。 父节点：ListVersionsResult.Version **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无
Type	String	**参数解释：** 对象的类型。 父节点：ListVersionsResult.Version **约束限制：** 非 Normal 对象时返回该参数。 **取值范围：** • NORMAL：普通对象 • APPENDABLE：追加写对象 **默认取值：** 无
Size	String	**参数解释：** 对象的字节数。 父节点：ListVersionsResult.Version **约束限制：** 无 **取值范围：** 0~48.8TB，单位：字节。 **默认取值：**

参数名称	参数类型	描述
		无
Owner	Container	**参数解释：** 用户信息，包含对象拥有者 DomainId 和对象拥有者名称。 父节点：ListVersionsResult.Version | ListVersionsResult.DeleteMarker **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
ID	String	**参数解释：** 对象拥有者的 DomainId。 父节点：ListVersionsResult.Version.Owner | ListVersionsResult.DeleteMarker.Owner **约束限制：** 无 **取值范围：** 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID **默认取值：** 无
DisplayName	String	**参数解释：** 对象拥有者名称。 父节点：ListVersionsResult.Version.Owner | ListVersionsResult.DeleteMarker.Owner **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
StorageClass	String	**参数解释：** 对象的存储类型。

参数名称	参数类型	描述
		父节点：ListVersionsResult.Version **约束限制：** 无 **取值范围：** • STANDARD • WARM • COLD **默认取值：** 无
CommonPrefixes	Container	**参数解释：** 请求中带 delimiter 参数时，返回消息带 CommonPrefixes 分组信息。 父节点：ListVersionsResult。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Prefix	String	**参数解释：** CommonPrefixes 分组信息中，表明不同的 Prefix。 父节点：ListVersionsResult.CommonPrefixes。 **约束限制：** 为桶内对象的前缀。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：列举所有对象

```
GET / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
```

```
Accept: */*
Date: WED, 01 Jul 2015 02:28:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:KiyoYze4pmRNPYfmlXBfRTVxt8c=
```

响应示例：列举所有对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D34E379ABD93320CB9
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSXiN7GPL/yXM6OSBaYCUV1zcY5OelWp
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:23:30 GMT
Content-Length: 586

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Name>examplebucket</Name>
  <Prefix></Prefix>
  <Marker></Marker>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>object001</Key>
    <LastModified>2015-07-01T00:32:16.482Z</LastModified>
    <ETag>"2fa3bcaaec668adc5da177e67a122d7c"</ETag>
    <Size>12041</Size>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      <DisplayName>ObjectOwnerName</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
</ListBucketResult>
```

请求示例：筛选对象

用户有桶名为 examplebucket，桶内共有四个名为 newfile，obj001，obj002，obs001 的对象，如果只需要列出对象名为 obj002 的对象，请求消息格式为：

```
GET /?marker=obj001&prefix=obj HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:28:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:KiyoYze4pmRNPYfmlXBfRTVxt8c=
```

响应示例：筛选对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000016435D758FBA857E0801874
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCShn/xAyk/xHBX6qgGSB36WXrbco0X80
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:29:48 GMT
Content-Length: 707
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketResult xmlns="http://obs.example.com/doc/2015-06-30/">
<Name>examplebucket</Name>
<Prefix>obj</Prefix>
<Marker>obj001</Marker>
<MaxKeys>1000</MaxKeys>
<IsTruncated>false</IsTruncated>
  <Contents>
    <Key>obj002</Key>
    <LastModified>2015-07-01T02:11:19.775Z</LastModified>
    <ETag>"a72e382246ac83e86bd203389849e71d"</ETag>
    <Size>9</Size>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      <DisplayName>ObjectOwnerName</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
</ListBucketResult>
```

请求示例：多版本

```
GET /?versions HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:29:45 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:iZeDESIMxBK2YODk7vIeVpyO8DI=
```

响应示例：多版本

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D758FBA857E0801874
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCShn/xAyk/xHBX6qgGSB36WXrbco0X80
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:29:48 GMT
Content-Length: 707

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListVersionsResult xmlns="http://obs.example.com/doc/2015-06-30/">
 <Name>bucket02</Name>
 <Prefix></Prefix>
 <KeyMarker></KeyMarker>
  <VersionIdMarker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <Version>
    <Key>object001</Key>

<VersionId>0001100000000000013F16000001643A22E476FFFF9046024ECA3655445346485a</Versio
nId>
    <IsLatest>true</IsLatest>
    <LastModified>2015-07-01T00:32:16.482Z</LastModified>
    <ETag>"2fa3bcaaec668adc5da177e67a122d7c"</ETag>
```

```
    <Size>12041</Size>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      <DisplayName>ObjectOwnerName</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Version>
</ListVersionsResult>
```

相关文档

调用列举桶内对象接口操作会产生计费，该操作相关计费说明请参考。

5.1.4 获取桶元数据

功能介绍

对桶拥有读权限的用户可以执行查询桶元数据是否存在的操作。

请求消息样式

```
HEAD / HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不带消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

如果想要获取 CORS 配置信息，则需要使用的消息头如下表 5-11 所示。

表 5-11 获取 CORS 配置的请求消息头

消息头名称	消息头类型	是否必选	描述
Origin	String	是	**参数解释：** 预请求指定的跨域请求 Origin（通常为域名）。 **约束限制：** 允许多条匹配规则，以回车换行为间隔。每个匹配规则允许使用最多一个“*”通配符。 **取值范围：**

消息头名称	消息头类型	是否必选	描述
			符合 http 协议的该头域的值。 **默认取值：** 无
Access-Control-Request-Headers	String	否	**参数解释：** 实际请求可以带的 HTTP 头域。 **约束限制:** 允许的头域可设置多个，多个头域之间换行隔开，每行最多可填写一个*符号，不支持&、:、<、空格以及中文字符。 **取值范围：** 符合 http 协议的该头域的值。 **默认取值：** 无

请求消息元素

该请求消息中不带消息元素。

响应消息样式

```
HTTP/1.1 status_code
x-obs-bucket-location: region
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如下表 5-12 中的消息头。

表 5-12 附加响应消息头

消息头名称	消息头类型	描述
x-obs-bucket-location	String	**参数解释：** 桶的区域位置信息。 **约束限制：** 无 **取值范围：** 无 **默认取值：**

消息头名称	消息头类型	描述
		无
x-obs-storage-class	String	**参数解释：** 桶的默认存储类型。 **约束限制：** 无 **取值范围：** • STANDARD（标准存储） • WARM（低频访问存储） • COLD（归档存储） **默认取值：** 无
x-obs-version	String	**参数解释：** 桶所在的 OBS 服务版本号。 **约束限制：** 无 **取值范围：** • 3.0：最新版本的桶。 • --：表示老版本的桶。 **默认取值：** 无
x-obs-fs-file-interface	String	**参数解释：** 判断是否为并行文件系统。 **约束限制：** 不携带此头域表示不属于并行文件系统。 **取值范围：** 取值包含 Enabled（并行文件系统）。 **默认取值：** 无
x-obs-epid	String	**参数解释：** 当前桶的企业项目 ID，开通企业项目的用户可以从企业项目服务获取。 **约束限制：** 格式为 uuid，未开通企业项目的用户可以不带该头域。 **默认取值：** 无

消息头名称	消息头类型	描述
x-obs-az-redundancy	String	**参数解释：** 桶的数据冗余存储策略属性，即 AZ 类型。 取值为 3az，表示数据冗余存储在同一区域的多个可用区。 不携带此头域表示为单 az 存储，仅使用 1 个可用区存储。 **约束限制：** 不支持多 AZ。如果桶所在区域不支持多 AZ 存储，则该桶的存储类型默认为单 AZ。 **取值范围：** 如果桶配置为多 AZ，则返回值为“3az”。如果桶配置为单 AZ，则返回值为 None。 **默认取值：** 无
Access-Control-Allow-Origin	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 Origin 满足服务端的 CORS 配置，则在响应中包含这个 Origin。 **约束限制：** 无 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-Allow-Headers	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 headers 满足服务端的 CORS 配置，则在响应中包含这个 headers。 **约束限制：** 无 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-Max-Age	Integer	**参数解释：** 当桶设置了 CORS 配置，服务端 CORS 配置中的 MaxAgeSeconds。

消息头名称	消息头类型	描述
		约束限制： 无 **取值范围：** 大于等于 0 的整型数，单位：秒。 **默认取值：** 3000
Access-Control-Allow-Methods	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 Access-Control-Request-Method 满足服务端的 CORS 配置，则在响应中包含这条 rule 中的 Methods。 **约束限制：** 无 **取值范围：** • GET • PUT • HEAD • POST • DELETE **默认取值：** 无
Access-Control-Expose-Headers	String	**参数解释：** 桶 CORS 规则中的 ExposeHeader。ExposeHeader 是指 CORS 规则允许响应中可返回的附加头域，给客户端提供额外的信息。默认情况下浏览器只能访问以下头域：Content-Length、Content-Type，如果需要访问其他头域，需要在附加头域中配置。 **约束限制：** 不支持*、&、:、<、空格以及中文字符。 **取值范围：** 无 **默认取值：** 无

响应消息元素

该请求的响应中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：未携带获取 CORS 配置

```
HEAD / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:30:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:niCQCuGIZpETKIyx1datxHZyYlk=
```

响应示例：未携带获取 CORS 配置

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016439C734E0788404623FA8
Content-Type: application/xml
x-obs-storage-class: STANDARD
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSxwLpq9Hzf3OnaXr+pI/OPLKdrtiQAF
Date: WED, 01 Jul 2015 02:30:25 GMT
x-obs-bucket-location: region
x-obs-version: 3.0
Content-Length: 0
```

请求示例：桶设置了 CORS 后，获取桶元数据和 CORS 配置

```
HEAD / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:30:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:niCQCuGIZpETKIyx1datxHZyYlk=
Origin:www.example.com
Access-Control-Request-Headers:AllowedHeader_1
```

响应示例：桶设置了 CORS 后，获取桶元数据和 CORS 配置

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016439C734E0788404623FA8
Content-Type: application/xml
x-obs-storage-class: STANDARD
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSxwLpq9Hzf3OnaXr+pI/OPLKdrtiQAF
Date: WED, 01 Jul 2015 02:30:25 GMT
x-obs-bucket-location: region
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT
Access-Control-Allow-Headers: AllowedHeader_1
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: ExposeHeader_1
```

```
x-obs-version: 3.0
Content-Length: 0
```

5.1.5 获取桶区域位置

功能介绍

对桶拥有读权限的用户可以执行获取桶区域位置信息的操作。

请求消息样式

```
GET /?location HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不带消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不带消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Location xmlns="http://obs.region.example.com/doc/2015-06-30/">region</Location>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该响应中将桶的区域信息以消息元素的形式返回，元素的具体含义如表 5-13 所示。

表 5-13 响应消息元素

元素名称	描述
Location	桶的区域位置信息。

元素名称	描述
	类型：String

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?location HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:30:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:1DrmbCV+lhz3zV7uywlj7lrh0MY=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D9F27CB2758E9B41A5
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSKWoJmaMyRXqofHgapbETDyI2LM9rUw
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:30:25 GMT
Content-Length: 128

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Location xmlns="http://obs.region.example.com/doc/2015-06-30/">region</Location>
```

5.1.6 删除桶

功能介绍

删除桶操作用于删除用户指定的桶。只有桶的所有者或者拥有桶的删桶 policy 权限的用户可以执行删除桶的操作，要删除的桶必须是空桶。如果桶中有对象或者有多段任务则认为桶不为空，可以使用列举桶内对象和列举出多段上传任务接口来确认桶是否为空。

注：

如果删除桶时，服务端返回 5XX 错误或超时，系统需要时间进行桶信息一致性处理，在此期间桶的信息会不准确，过一段时间再查看桶是否删除成功，查询到桶，需要再次发送删除桶消息。

请求消息样式

```
DELETE / HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共的请求消息头，具体请参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，错误已经包含在表 6-2 中。

请求示例

```
DELETE / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:31:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF260000016435DE6D67C35F9B969C47
x-obs-id-2: 32AAAQAAEAABKAAQAAEAABAAAQAAEAABCTukraCnXLsb7lEw4ZKjzDWWhzXdgme3
Date: WED, 01 Jul 2015 02:31:25 GMT
```

5.2 桶的高级配置

5.2.1 设置桶策略

功能介绍

使用本接口可以修改或者新建指定桶的桶策略。如果桶已经存在一个策略，那么当前请求中的策略将完全覆盖桶中现存的策略。单个桶的桶策略条数（statement）没有限制，但一个桶中所有桶策略的 JSON 描述总大小不能超过 20KB。

要使用该接口，使用者要求必须是桶的所有者，或者是桶所有者的子用户且具有设置桶策略的权限。

请求消息样式

```
PUT /?policy HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: signatureValue
Policy written in JSON
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体请参见表 3-3。

请求消息元素

请求消息体是一个符合 JSON 格式的字符串，包含了桶策略的信息。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例 1

向 OBS 租户授予权限

给租户 ID 为 783fc6652cf246c096ea836694f71855 的租户授权。

如何获取租户 ID 请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:32:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Sid": "Stmt1375240018061",
         "Action": [
            "GetBucketLogging"
         ],
         "Effect": "Allow",
         "Resource": "logging.bucket",
         "Principal": {
            "ID": [
               "domain/783fc6652cf246c096ea836694f71855:user/*"
            ]
         }
      }
   ]
}
```

响应示例 1

```
HTTP/1.1 204 No Content
x-obs-request-id: 7B6DFC9BC71DD58B061285551605709
x-obs-id-2: N0I2REZDOUJDNzFERDU4QjA2MTI4NTU1MTYwNTcwOUFBQUFBQUFBYmJiYmJiYmJD
Date: WED, 01 Jul 2015 02:32:25 GMT
Content-Length: 0
Server: OBS
```

请求示例 2

向 OBS 用户授予权限

用户 ID 为 71f3901173514e6988115ea2c26d1999，用户所属租户 ID 为 783fc6652cf246c096ea836694f71855。

如何获取租户 ID 和用户 ID 请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:33:28 GMT
```

```
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Sid": "Stmt1375240018062",
         "Action": [
            "PutBucketLogging"
         ],
         "Effect": "Allow",
         "Resource": "examplebucket",
         "Principal": {
            "ID": [

"domain/783fc6652cf246c096ea836694f71855:user/71f3901173514e6988115ea2c26d1999"
            ]
         }
      }
   ]
}
```

响应示例 2

```
HTTP/1.1 204 No Content
x-obs-request-id: 7B6DFC9BC71DD58B061285551605709
x-obs-id-2: N0I2REZDOUJDNzFERDU4QjA2MTI4NTU1MTYwNTcwOUFBQUFBQUFBYmJiYmJiYmJD
Date: WED, 01 Jul 2015 02:33:28 GMT
Content-Length: 0
Server: OBS
```

请求示例 3

拒绝除了某个指定 OBS 用户的其他用户执行所有操作

用户 ID 为 71f3901173514e6988115ea2c26d1999，用户所属租户 ID 为 783fc6652cf246c096ea836694f71855。

如何获取租户 ID 和用户 ID 请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:34:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Effect": "Deny",
         "Action": ["*"],
         "Resource": [
            "examplebucket/*",
            "examplebucket"
         ],
```

```
         "NotPrincipal": {
            "ID": [

"domain/783fc6652cf246c096ea836694f71855:user/71f3901173514e6988115ea2c26d1999",
               "domain/783fc6652cf246c096ea836694f71855:root"
            ]
         }
      }
   ]
}
```

响应示例 3

```
HTTP/1.1 204 No Content
x-obs-request-id: A603000001604A7DFE4A4AF31E301891
x-obs-id-2: BKOvGmTlt6sda5X4G89PuMO4fabObGYmnpRGkaMba1LqPt0fCACEuCMllAObRK1n
Date: WED, 01 Jul 2015 02:34:34 GMT
Content-Length: 0
Server: OBS
```

请求示例 5

向指定委托授予权限

租户的账号 ID 为 783fc6652cf246c096ea836694f71855 的租户，该租户有一个名为 exampleAgency 的委托，以下为向委托授予名为 logging.bucket 桶的日志查看权限。

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:32:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Sid": "Stmt1375240018061",
         "Action": [
            "GetBucketLogging"
         ],
         "Effect": "Allow",
         "Resource": "logging.bucket",
         "Principal": {
            "ID": [
               "domain/783fc6652cf246c096ea836694f71855:agency/exampleAgency"
            ]
         }
      }
   ]
}
```

响应示例 5

```
HTTP/1.1 204 No Content
x-obs-request-id: A603000001604A7DFE4A4AF31E301891
```

```
x-obs-id-2: BKOvGmTlt6sda5X4G89PuMO4fabObGYmnpRGkaMba1LqPt0fCACEuCMllAObRK1n
Date: WED, 01 Jul 2015 02:34:34 GMT
Content-Length: 0
Server: OBS
```

5.2.2 获取桶策略

功能介绍

该接口的实现使用 policy 子资源来将指定桶的策略返回给客户端。

要使用该接口，使用者要求必须是桶的所有者，或者是桶所有者的子用户且具有获取桶策略的权限。

以下两种场景无法使用此接口获取桶策略，系统将返回“404 NoSuchBucketPolicy”的错误：

- 指定桶的策略不存在
- 指定桶的标准桶策略为私有且未设置高级桶策略

请求消息样式

```
GET /?policy HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: application/xml
Date: date
Policy Content
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

响应消息体是一个 JSON 格式的桶策略字符串。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:35:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

响应示例

```
HTTP/1.1 200 OK
x-obs-request-id: A603000001604A7DFE4A4AF31E301891
x-obs-id-2: BKOvGmTlt6sda5X4G89PuMO4fabObGYmnpRGkaMba1LqPt0fCACEuCMllAObRK1n
Date: WED, 01 Jul 2015 02:35:46 GMT
Content-Length: 509
Server: OBS

{
   "Statement":[
      {
         "Sid":"Stmt1375240018061",
         "Effect":"Allow",
         "Principal":{
            "ID":[

"domain/domainiddomainiddomainiddo006666:user/useriduseriduseriduseridus004001",
                "domain/domainiddomainiddomainiddo006667:user/*"
            ]
         },
         "Action":[
            "*"
         ],
         "Resource":[
            "examplebucket"
         ]
      }
   ]
}
```

5.2.3 删除桶策略

功能介绍

该接口的实现是通过使用 policy 子资源来删除一个指定桶上的策略。

要使用该接口，使用者要求必须是桶的所有者，或者是桶所有者的子用户且具有删除桶策略的权限。

无论桶的策略本身是否存在，删除成功后系统都直接返回“204 No Content”的结果。

请求消息样式

```
DELETE /?policy HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: text/xml
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:36:06 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

响应示例

```
HTTP/1.1 204 No Content
x-obs-request-id: 9006000001643AAAF70BF6152D71BE8A
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSB4oWmNX3gVGGLr1cRPWjOhffEbq1XV
```

```
Date: WED, 01 Jul 2015 02:36:06 GMT
Server: OBS
```

5.2.4 设置桶 ACL

功能介绍

OBS 支持对桶操作进行权限控制。默认情况下，只有桶的创建者才有该桶的读写权限。用户也可以设置其他的访问策略，比如对一个桶可以设置公共访问策略，允许所有人对其都有读权限。

OBS 用户在创建桶时可以设置权限控制策略，也可以通过 ACL 操作 API 接口对已存在的桶更改或者获取 ACL(access control list) 。一个桶的 ACL 最多支持 100 条 Grant 授权。PUT 接口为幂等的覆盖写语意，新设置的桶 ACL 将覆盖原有的桶 ACL，如果需要修改或者删除某条 ACL 重新 PUT 一个新的桶 ACL 即可。

请求消息样式

```
PUT /?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-Type: application/xml
Content-Length: length

<AccessControlPolicy>
   <Owner>
      <ID>ID</ID>
   </Owner>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>domainId</ID>
         </Grantee>
         <Permission>permission</Permission>
         <Delivered>false</Delivered>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

请求消息参数

该操作请求不带消息参数。

请求消息头

使用者可以使用头域设置的方式来更改桶的 ACL，每一种头域设置的 ACL 都有一套自己预先定义好的被授权用户以及相应权限，通过头域设置的方式授予访问权限，使用者必须添加以下的头域并且指定取值。

表 5-14 头域方式设置桶 ACL

名称	类型	是否必选	描述
x-obs-acl	String	否	**参数解释：** 通过 canned ACL 的方式来设置桶的 ACL。 **约束限制：** 无 **取值范围：** • private • public-read • public-read-write • public-read-delivered • public-read-write-delivered **默认取值：** private

请求消息元素

更改桶的 ACL 请求需要在消息元素中带上 ACL 信息，元素的具体含义如表 3-3 所示。

表 5-15 附加请求消息元素

元素名称	元素类型	是否必选	描述
Owner	XML	是	**参数解释：** 桶的所有者信息，是桶拥有者账号 ID 的父节点。 **约束限制：** 无
ID	String	是	**参数解释：** 桶拥有者账号 ID。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
AccessControlList	XML	是	**参数解释：** 访问控制列表，是 Grant 的父节点。 **约束限制：** 无

元素名称	元素类型	是否必选	描述
Grant	XML	否	**参数解释：** 用于标记用户及用户的权限，是 Grantee、Permission 和 Delivered 的父节点。 **约束限制：** 单个桶的 ACL，Grant 元素不能超过 100 个。
Grantee	XML	否	**参数解释：** 记录用户信息，是被授权账号 ID 的父节点。 **约束限制：** 无
ID	String	是	**参数解释：** 被授权用户的账号 ID。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Canned	String	否	**参数解释：** 向所有人授予权限。 **约束限制：** 无 **取值范围：** Everyone：所有人 **默认取值：** 无
Permission	String	是	**参数解释：** 授予的权限。 **约束限制：** 无 **取值范围：** • READ：允许被授权者获取桶内对象列表和桶元数据。 • READ_ACP：允许被授权者读取桶

元素名称	元素类型	是否必选	描述
			ACL 属性。 • WRITE：允许被授权者向桶中上传对象。对于桶中现有对象，允许授权者删除和覆盖这些对象。 • WRITE_ACP：允许被授权者更新桶 ACL 属性。 • FULL_CONTROL：允许被授予者拥有 READ、WRITE、READ_ACP 和 WRITE_ACP 权限。 **默认取值：** 无
Delivered	Boolean	否	**参数解释：** 桶的 ACL 是否向桶内对象传递。作用于桶内所有对象。 **约束限制：** 无 **取值范围：** • true：向桶内对象传递 • false：不向桶内对象传递 **默认取值：** false

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?acl HTTP/1.1
User-Agent: curl/7.29.0
```

```
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:37:22 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:iqSPeUBl66PwXDApxjRKk6hlcN4=
Content-Length: 727

<AccessControlPolicy xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
      <Delivered>false</Delivered>
    </Grant>
    <Grant>
      <Grantee>
        <Canned>Everyone</Canned>
      </Grantee>
      <Permission>READ_ACP</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164361F2954B4D063164704
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT78HTIBuhe0FbtSptrb/akwELtwyPKs
Date: WED, 01 Jul 2015 02:37:22 GMT
Content-Length: 0
```

5.2.5 获取桶 ACL

功能介绍

用户执行获取桶 ACL 的操作，返回信息包含指定桶的权限控制列表信息。用户必须拥有对指定桶 READ_ACP 的权限或 FULL_CONTROL 权限，才能执行获取桶 ACL 的操作。

请求消息样式

```
GET /?acl HTTP/1.1
Host: bucketname.obs.region.example.com
```

```
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Owner>
      <ID>id</ID>
   </Owner>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>id</ID>
         </Grantee>
         <Permission>permission</Permission>
         <Delivered>false</Delivered>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应中以消息元素的形式返回桶的 ACL 信息，元素的具体意义如表 5-16 所示。

表 5-16 响应消息元素

元素	元素说明
Owner	桶的所有者信息。

元素	元素说明
	类型：XML
ID	用户所属租户的租户 Id。 类型：String
AccessControlList	访问控制列表，记录了对该桶有访问权限的用户列表和这些用户具有的权限。 类型：XML
Grant	用于标记用户及用户的权限。 类型：XML
Grantee	记录用户信息。 类型：XML
Canned	向所有人授予权限。 类型：String，其值只能是 Everyone。
Delivered	桶的 ACL 是否向桶内对象传递。 类型：Boolean
Permission	指定的用户对该桶所具有的操作权限。 类型：String

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:39:28 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:X7HtzGsIEkzJbd8vo1DRu30vVrs=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436B69D82F14E93528658
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSjTh8661+HF5y8uAnTOBIpNO133hji+
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:39:28 GMT
Content-Length: 784

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
```

```
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
      <Delivered>false</Delivered>
    </Grant>
    <Grant>
      <Grantee>
        <Canned>Everyone</Canned>
      </Grantee>
      <Permission>READ_ACP</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

5.2.6 设置桶日志管理配置

功能介绍

创建桶时，默认是不生成桶的日志的，如果需要生成桶的日志，该桶需要打开日志配置管理的开关。桶日志功能开启后，桶的每次操作将会产生一条日志，并将多条日志打包成一个日志文件。日志文件存放位置需要在开启桶日志功能时指定，可以存放到开启日志功能的桶中，也可以存放到其他您有权限的桶中，但需要和开启日志功能的桶在同一个 region 中。

由于日志文件是 OBS 产生，并且由 OBS 上传到存放日志的桶中，因此 OBS 需要获得委托授权，用于上传生成的日志文件，所以在配置桶日志管理前，需要先到统一身份认证服务生成一个对 OBS 服务的委托，并将委托名作为参数配置到桶上，并且在 xml 文件中<LoggingEnabled>标签下配置相应的日志管理功能。在为委托配置权限时只需设置目标桶的上传对象权限。

委托权限示例

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Action": [
            "obs:object:PutObject"
         ],
         "Resource": [
            "OBS:*:*:object:mybucketlogs/*"
         ],
```

```
            "Effect": "Allow"
        }
    ]
}
```

关闭桶日志功能的方法是上传一个带有空的 BucketLoggingStatus 标签的 logging 文件。

默认存储类别为低频访问存储或归档存储的桶不能作为存放日志文件的桶。日志文件存放到桶中后，这些日志文件会占用空间，并按照用户存放数据同样的计费策略进行计费。

⚠ 注意

目标桶开启 KMS 加密时，需要在委托中授予 KMS 访问权限。

请求消息样式

```
PUT /?logging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: signatureValue
<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus>
  <Agency>agency-name</Agency>
  <LoggingEnabled>
    <TargetBucket>mybucketlogs</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>

    <TargetGrants>
      <Grant>
        <Grantee>
          <ID>domainID</ID>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体请参见表 3-3。

请求消息元素

表 5-17 请求消息元素表

名字	描述	是否必选

名字	描述	是否必选
BucketLoggingStatus	日志状态信息的容器。 类型：Container	是
Agency	目标桶 Owner 通过统一身份认证服务创建的对 OBS 服务的委托的名称。 类型：String	设置 logging 时必选。关闭 logging 时勿选。
LoggingEnabled	该元素起到对日志配置管理的使能作用（呈现此元素则打开日志配置，否则关闭配置）。在此元素下，可加入具体的日志配置信息。 类型：Container	设置 logging 时必选。关闭 logging 时勿选。
Grant	是被授权者及其权限的容器。用于描述谁有什么权限来访问产生的日志文件。 类型：Container	否
Grantee	作为被授权 logging 权限用户的容器。 类型：Container	否
ID	被授权者的租户 ID，全局唯一标识。 类型：String	否
Permission	产生的日志文件对被授权者的具体权限。 类型：String 权限有效值：FULL_CONTROL | READ | WRITE	否
TargetBucket	在生成日志时，配置日志桶的所有者可以指定一个桶用于存放产生的日志文件。需要保证配置日志文件的桶 owner 对存放日志文件的桶有 FULL_CONTROL 权限。支持多个桶生成的日志放在同一个目标桶中，如果这样做，就需要指定不同	设置 logging 时必选。关闭 logging 时勿选。

名字	描述	是否必选
	的 TargetPrefix 以达到为来自不同源桶的日志分类的目的。 类型：String	
TargetPrefix	通过该元素指定一个前缀，所有生成的日志对象的对象名都以此元素的内容为前缀。 类型：String	设置 logging 时必选。关闭 logging 时勿选。
TargetGrants	授权信息的容器。 类型：Container	否

存储访问日志的 *object* 命名规则

```
<TargetPrefix>YYYY-mm-DD-HH-MM-SS-<UniqueString>
```

- <TargetPrefix>为用户指定的目标前缀。
- YYYY-mm-DD-HH-MM-SS 为日志生成的日期与时间，各字段依次表示年、月、日、时、分、秒。
- <UniqueString>为 OBS 自动生成的字符串。

一个实际用于存储 OBS 访问日志的 object 名称示例如下：

```
bucket-log2015-06-29-12-22-07-N7MXLAF1BDG7MPDV
```

- "bucket-log"为用户指定的目标前缀。
- "2015-06-29-12-22-07"为日志生成的日期与时间。
- "N7MXLAF1BDG7MPDV"为 OBS 自动生成的字符串。

桶访问日志格式

以下所示为在目标桶生成的桶访问日志文件记录：

```
787f2f92b20943998a4fe2ab75eb09b8 bucket [13/Aug/2015:01:43:42 +0000] xx.xx.xx.xx
787f2f92b20943998a4fe2ab75eb09b8 281599BACAD9376ECE141B842B94535B
REST.GET.BUCKET.LOCATION - "GET /bucket?location HTTP/1.1" 200 - 211 - 6 6 "-"
"HttpClient" - -
```

每个桶访问日志都包含如下信息：

表 5-18 Bucket Logging 格式

名称	示例	含义
BucketOwner	787f2f92b20943998a4fe2ab75eb09b8	桶的 ownerId
Bucket	bucket	桶名

名称	示例	含义
Time	[13/Aug/2015:14:43:42 +0000]	请求时间戳。格式为：[dd/MMM/yyyy:HH:mm:ss Z]，即 [日/月/年:小时:分钟:秒 时区]
Remote IP	xx.xx.xx.xx	请求 IP
Requester	787f2f92b20943998a4fe2ab75eb09b8	请求者 ID • 当使用账号或 IAM 用户发起请求时，此 ID 为请求者所属账号的账号 ID。 • 当使用匿名用户发起请求时，取值为 Anonymous。
RequestID	281599BACAD9376ECE141B842B94535B	请求 ID
Operation	REST.GET.BUCKET.LOCATION	操作名称
Key	-	对象名
Request-URI	GET /bucket?location HTTP/1.1	请求 URI
HTTPStatus	200	返回码。
ErrorCode	-	错误码
BytesSent	211	HTTP 响应的字节大小
ObjectSize	-	对象大小
TotalTime	6	服务端处理时间 单位：ms
Turn-AroundTime	6	总请求时间 单位：ms
Referer	-	请求的 referrer 头域
User-Agent	HttpClient	请求的 user-agent 头域
VersionID	-	请求中带的 versionId
STSLogUrn	-	联邦认证及委托授权信息
StorageClass	STANDARD_IA	当前的对象存储类型。 • STANDARD：标准存储 • STANDARD_IA：即

名称	示例	含义
		“WARM”，低频访问存储 • COLD：归档存储
TargetStorageClass	GLACIER	通过转换后的对象存储类型
DentryName	12456%2Ffile.txt	• 对于并行文件系统，是文件/目录的内部标识，由父目录 inode 编号与文件/目录名称组成，此字段最终呈现为经过 URL 编码后的格式。 • 对于对象桶，该字段为"-"。
IAMUserId	8f3b8c53d29244a780084f2b8c106c32	IAM 用户 ID。 当使用匿名用户发起请求，记为 Anonymous。
AccessKeyID	UDSIAMSTUBTEST002852	请求者的 AccessKey ID。取值-表示匿名请求。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?logging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:40:06 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mCOjER/L4ZZUY9qr6AOnkEiwvVk=
Content-Length: 528
```

```
<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus>
  <Agency>agencyGrantPutLogging</Agency>
  <LoggingEnabled>
    <TargetBucket>log-bucket</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>

    <TargetGrants>
      <Grant>
        <Grantee>
          <ID>783fc6652cf246c096ea836694f71855</ID>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643663CE53B6AF31C619FD
x-obs-id-2: 32AAAQAAEAABSAAkpAIAABAAAQAAEAABCT9CjuOx8cETSRbqkm35s1dL/tLhRNdZ
Date: WED, 01 Jul 2015 02:40:06 GMT
Content-Length: 0
```

5.2.7 获取桶日志管理配置

功能介绍

该接口的目的是查询当前桶的日志管理配置情况。其实现是通过使用 http 的 get 方法再加入 logging 子资源来返回当前桶的日志配置情况。

要使用该接口，使用者必须是桶的所有者或者是被桶策略授权 GetBucketLogging 权限的用户。

请求消息样式

```
GET /?logging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Agency>agency-name</Agency>
<LoggingEnabled>
   <TargetBucket>bucketName</TargetBucket>
   <TargetPrefix>prefix</TargetPrefix>

      <TargetGrants>
         <Grant>
            <Grantee>
               <ID>id</ID>
            </Grantee>
            <Permission>permission</Permission>
         </Grant>
      </TargetGrants>
   </LoggingEnabled>
</BucketLoggingStatus>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应中以消息元素的形式返回桶的日志信息，元素的具体意义如表 5-19 所示。

表 5-19 响应消息元素

名字	描述
BucketLoggingStatus	logging 状态信息的容器。 类型：Container
Agency	产生 logging 日志桶 Owner 创建委托 OBS 上传 logging 日志的委托名。 类型：String
LoggingEnabled	用于 logging 信息的容器。并且该元素起到对 logging 配置管理的使能作用（呈现此元素则打开 logging 配置，否则关闭）。 类型：Container

名字	描述
Grant	是被授权者及其权限的容器。 类型：Container
Grantee	作为被授权 logging 权限用户的容器。 类型：Container
ID	被授权用户的 Domain Id，全局唯一标识。 类型：String
Permission	对于一个桶的 logging 权限来说，owner 在创建桶时将自动获得对源桶的 FULL_CONTROL 权限。不同的权限决定了对不同日志的访问限制。 类型：String 权限有效值：FULL_CONTROL | READ | WRITE
TargetBucket	在生成日志时，源桶的 owner 可以指定一个目标桶，将生成的所有日志放到该桶中。在 OBS 系统中，支持多个源桶生成的日志放在同一个目标桶中，如果这样做，就需要指定不同的 TargetPrefix 以达到为来自不同源桶的日志分类的目的。 类型：String
TargetPrefix	通过该元素可以指定一个前缀给一类日志生成的对象。 类型：String
TargetSorting	配置桶日志归类时指定，按照指定的类型进行日志归类。 类型：String 有效值：DAY|HOUR 否
TargetGrants	授权信息的容器。 类型：Container

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?logging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:42:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:hUk+jTnR07hcKwJh4ousF2E1U3E=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436B8EEE7FBA2AA3335E3
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCShuQJoWFpS77C8bOv1mqURv0UY+0ejx
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:42:46 GMT
Content-Length: 429

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<BucketLoggingStatus xmlns="http://obs.example.com/doc/2015-06-30/">
  <Agency>agency-name</Agency>
  <LoggingEnabled>
    <TargetBucket>log-bucket</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>

    <TargetGrants>
      <Grant>
        <Grantee>
          <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

5.2.8 设置桶的生命周期配置

功能介绍

OBS 系统支持指定规则来实现定时删除或迁移桶中对象，这就是生命周期配置。典型的应用场景如：

- 周期性上传的日志文件，可能只需要保留一个星期或一个月，到期后要删除它们。
- 某些文档在一段时间内经常访问，但是超过一定时间后就可能不会再访问了。这种文档您可能会先选择归档，然后在一定时间后删除。

本接口实现为桶创建或更新生命周期配置信息。

说明

- 对象生命周期到期以后，对象将会永久删除，无法恢复。

要正确执行此操作，需要确保执行者有 PutLifecycleConfiguration 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

生命周期配置实现了定时删除对象和定时迁移对象的功能，所以如果想要阻止用户删除或迁移对象，以下几项操作的权限都应该被禁止：

- DeleteObject
- DeleteObjectVersion
- PutLifecycleConfiguration

如果想要阻止用户管理桶的生命周期配置，应该禁止 PutLifecycleConfiguration 权限。

请求消息样式

```
PUT /?lifecycle HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
Content-MD5: MD5
<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration>
   <Rule>
      <ID>id</ID>
      <Prefix>prefix</Prefix>
      <Status>status</Status>
      <Expiration>
         <Days>days</Days>
      </Expiration>
      <NoncurrentVersionExpiration>
         <NoncurrentDays>days</NoncurrentDays>
      </NoncurrentVersionExpiration>
      <Transition>
       <Days>30</Days>
        <StorageClass>WARM</StorageClass>
      </Transition>
      <Transition>
       <Days>60</Days>
       <StorageClass>COLD</StorageClass>
      </Transition>
      <NoncurrentVersionTransition>
       <NoncurrentDays>30</NoncurrentDays>
       <StorageClass>WARM</StorageClass>
      </NoncurrentVersionTransition>
      <NoncurrentVersionTransition>
       <NoncurrentDays>60</NoncurrentDays>
       <StorageClass>COLD</StorageClass>
      </NoncurrentVersionTransition>
      <AbortIncompleteMultipartUpload>
         <DaysAfterInitiation>10</DaysAfterInitiation>
      </AbortIncompleteMultipartUpload>
   </Rule>
</LifecycleConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用的消息头如下表 5-20 所示。

表 5-20 请求消息头

参数	是否必选	参数类型	描述
Content-MD5	是	String	**参数解释：** 按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128bit MD5 值经过 base64 编码后得到的字符串。 示例：n58IG6hfM7vqI4K0vnWpog== **取值范围：** 不涉及 **默认取值：** 不涉及

请求消息元素

在此请求中，需要在请求的消息体中配置桶的生命周期配置信息。配置信息以 XML 格式上传，具体的配置元素如表 5-21 描述。

- 如果桶的多版本是 Enabled 或者 Suspended，那么可以设置 NoncurrentVersionTransition 或 NoncurrentVersionExpiration 来控制对象的历史版本的生命周期。一个历史版本的生命周期，取决于它成为历史版本的时刻（即被新版本覆盖的那个时刻）和 NoncurrentDays。对于删除来说，例如 NoncurrentDays 配置为 1 的话，表示当一个版本成为历史版本之后，再过 1 天才能删除。对象 A 的版本 V1 创建于 1 号，5 号的时候又上传新的版本 V2，此时 V1 成为历史版本，那么再过 1 天，7 号的 0 点，V1 就过期了。如果该版本不满足删除，迁移配置 NoncurrentDays 为 1，StorageClass 为 WARM 的话，表示当一个版本成为历史版本之后，再过 1 天转为低频访问存储对象。对象 A 的版本 V1 创建于 1 号，5 号的时候又上传新的版本 V2，此时 V1 成为历史版本，那么再过 1 天，7 号的 0 点，V1 就会迁移成低频访问存储对象了。（备注：对象过期后被删除或对象迁移的时间可能会有一定的延迟，一般不超过 48 小时。）
- 如果桶的多版本是 Enabled 或者 Suspended，且最新版本对象满足 Expiration 规则时的处理：
  - 桶当前的多版本状态为 Enabled：

    如果对象的最新版本不是 deletemarker，则该对象会产生一个新的 deletemarker；

    如果最新版本是 deletemarker，且该对象只有这一个版本，则这个版本会被删除；

    如果最新版本是 deletemarker，且对象还有其他版本，则该对象的所有版本维持不变，没有新增和删除，也不会被修改（即无任何变化）。
  - 桶当前的多版本状态为 Suspended：

    如果对象的最新版本不是 deletemarker，且版本不是 null 版本，则会产生一个新的 null 版本的 deletemarker；

    如果对象的最新版本不是 deletemarker，且版本是 null 版本，则这个 null 版本会被新产生的 null 版本的 deletemarker 覆盖；

如果最新版本是 deletemarker，且该对象只有这一个版本，则这个版本会被删除；

如果最新版本是 deletemarker，且对象还有其他版本，则该对象的所有版本维持不变，没有新增和删除，也不会被修改（即无任何变化）。

- 如果桶的多版本是 Enabled 或者 Suspended，且最新版本对象满足 Transition 规则时的处理：
  - 如果对象的最新版本是 deletemarker，则这个版本不被迁移；
  - 如果最新版本不是 deletemarker，且该对象满足迁移条件，则这个版本会被迁移。

表 5-21 生命周期配置元素

参数名称	是否必选	参数类型	描述
LifecycleConfiguration	是	XML	**参数解释：** 生命周期配置规则。可以配置多条规则，LifecycleConfiguration 是表 5-22 的父节点。 **约束限制：** 整个配置消息体总大小不超过 20KB。 **取值范围：** 不涉及 **默认取值：** 不涉及
Rule	是	XML	**参数解释：** 具体某一条生命周期配置。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-22。 **默认取值：** 不涉及

表 5-22 Rule 参数说明

参数	是否必选	参数类型	描述
ID	否	String	**参数解释：** 一条规则的标识。 **约束限制：** 只能由大写或小写的英文字母、数字、英文句号（.）、下划线（_）和连字符

参数	是否必选	参数类型	描述
			（-）组成。 **取值范围：** 长度为 0~255 的字符串。 **默认取值：** 不涉及
Prefix	是	String	**参数解释：** 对象名前缀，用来标识哪些对象可以匹配到当前规则。 **约束限制：** • 当按前缀配置时，如果指定的前缀名与某条已配置的生命周期规则指定的前缀名存在包含关系，OBS 会将两条规则视为同一条，而禁止您配置本条规则。例如，系统中已存在指定前缀名为“abc”的规则，则不允许再配置指定前缀以“abc”字段开头的规则。 • 如果已存在按前缀配置的生命周期规则，则不允许再新增配置到整个桶的规则。 **取值范围：** 长度为 0~1024 的字符串。 **默认取值：** 不涉及
Transition	如果没有 NoncurrentVersionTransition, Expiration, NoncurrentVersionExpiration 则必选	XML	**参数解释：** 生命周期配置中表示转换时间和转换后对象存储级别的元素。 **约束限制：** 仅针对对象的最新版本。 **取值范围：** 请详见表 5-23。 **默认取值：** 不涉及
Expiration	如果没有 Transition, NoncurrentVersionTransition, NoncurrentVersionExpirat	XML	**参数解释：** 在生命周期配置中表示过期时间。 **约束限制：** 仅针对对象的最新版本。 **取值范围：**

参数	是否必选	参数类型	描述
	ion 则必选		请详见表 5-24。 **默认取值：** 不涉及
NoncurrentVersionTransition	如果没有 Transition, Expiration, NoncurrentVersionExpiration 则必选	XML	**参数解释：** 生命周期配置中表示转换时间和转换后对象存储级别的元素。 **约束限制：** 仅针对对象的历史版本。 **取值范围：** 请详见表 5-25。 **默认取值：** 不涉及
NoncurrentVersionExpiration	否	XML	**参数解释：** 在生命周期配置中表示历史版本的过期时间。您可以将该动作设置在已启用多版本（或暂停）的桶，让系统删除对象的满足特定生命周期的历史版本。 **约束限制：** 仅针对历史版本。 **取值范围：** 请详见表 5-26 **默认取值：** 不涉及
Status	是	String	**参数解释：** 标识当前规则是否启用。 **约束限制：** 不涉及 **取值范围：** • Enabled：启用规则。 • Disabled：停用规则。 **默认取值：** 不涉及

表 5-23 Transition 参数说明

参数	是否必选	参数类型	描述

参数	是否必选	参数类型	描述
Date	如果没有 Days 元素，则必选	String	**参数解释：** 指定 OBS 对该日期之前的对象执行生命周期规则。 **约束限制：** 日期格式必须为 ISO8601 的格式，并且为 UTC 的零点。 例如：2018-01-01T00:00:00.000Z，表示将最后修改时间早于 2018-01-01T00:00:00.000Z 的对象删除或转换成其他存储类型，等于或晚于这个时间的对象不会被删除或转储。 **取值范围：** 日期格式的字符串。 **默认取值：** 不涉及
Days	如果没有 Date 元素，则必选	Integer	**参数解释：** 指定生命周期规则在对象最后更新过后多少天生效。单位：天。 **约束限制：** 仅针对对象的最新版本。 **取值范围：** 取值为 1~2147483647 的数字。 **默认取值：** 不涉及
StorageClass	如果有 Transition 或 NoncurrentVersionTransition 元素，则必选	String	**参数解释：** 表示对象将被修改成的目标存储类别。 **约束限制：** 不涉及 **取值范围：** • WARM：低频访问存储 • COLD：归档存储 **默认取值：** 不涉及

表 5-24 Expiration 参数说明

参数	是否必选	参数类型	描述
Date	如果没有 Days 元素，则必选	String	**参数解释：** 指定 OBS 对该日期之前的对象执行生命周期规则。 **约束限制：** 日期格式必须为 ISO8601 的格式，并且为 UTC 的零点。 例如：2018-01-01T00:00:00.000Z，表示将最后修改时间早于 2018-01-01T00:00:00.000Z 的对象删除或转换成其他存储类型，等于或晚于这个时间的对象不会被删除或转储。 **取值范围：** 日期格式的字符串。 **默认取值：** 不涉及
Days	如果没有 Date 元素，则必选	Integer	**参数解释：** 指定生命周期规则在对象最后更新过后多少天生效。单位：天。 **约束限制：** 仅针对对象的最新版本。 **取值范围：** 取值为 1~2147483647 的数字。 **默认取值：** 不涉及

表 5-25 NoncurrentVersionTransition 参数说明

参数	是否必选	参数类型	描述
Noncurrent Days	如果有 NoncurrentVersionExpiration 或 NoncurrentVersionTransition 元素，则必选	Integer	**参数解释：** 表示对象在成为历史版本之后第几天时规则生效。单位：天。 **约束限制：** 仅针对历史版本。 **取值范围：** • 仅设置一种转换：最少设置 1 天。 • 设置多种转换：后者时间要比前者时间最少长 1 天。

参数	是否必选	参数类型	描述
			• 设置多种转换：针对同一个对象的历史版本转深度归档存储时间需要晚于转归档存储时间，转归档存储时间需要晚于转低频访问存储时间。 **默认取值：** 不涉及
StorageClass	如果有 Transition 或 NoncurrentVersionTransition 元素，则必选	String	**参数解释：** 表示对象将被修改成的目标存储类别。 **约束限制：** 不涉及 **取值范围：** • WARM：低频访问存储 • COLD：归档存储 **默认取值：** 不涉及

表 5-26 NoncurrentVersionExpiration 参数说明

参数	是否必选	参数类型	描述
NoncurrentDays	如果有 NoncurrentVersionExpiration 或 NoncurrentVersionTransition 元素，则必选	Integer	**参数解释：** 表示对象在成为历史版本之后第几天时规则生效。单位：天。 **约束限制：** 仅针对历史版本。 **取值范围：** 取值为 1~2147483647 的数字。 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?lifecycle HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:05:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:DpSAlmLX/BTdjxU5HOEwflhM0WI=
Content-MD5: ujCZn5p3fmczNiQQxdsGaQ==
Content-Length: 919

<?xml version="1.0" encoding="utf-8"?>
<LifecycleConfiguration>
  <Rule>
    <ID>lifecycle-rule-id</ID>
    <Prefix>test/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>70</Days>
    </Expiration>
    <NoncurrentVersionExpiration>
      <NoncurrentDays>70</NoncurrentDays>
    </NoncurrentVersionExpiration>
    <Transition>
      <Days>30</Days>
      <StorageClass>WARM</StorageClass>
    </Transition>
    <Transition>
      <Days>60</Days>
      <StorageClass>COLD</StorageClass>
    </Transition>
    <NoncurrentVersionTransition>
      <NoncurrentDays>30</NoncurrentDays>
      <StorageClass>WARM</StorageClass>
    </NoncurrentVersionTransition>
    <NoncurrentVersionTransition>
      <NoncurrentDays>60</NoncurrentDays>
      <StorageClass>COLD</StorageClass>
    </NoncurrentVersionTransition>
    <AbortIncompleteMultipartUpload>
       <DaysAfterInitiation>10</DaysAfterInitiation>
    </AbortIncompleteMultipartUpload>
  </Rule>
</LifecycleConfiguration>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643670AC06E7B9A7767921
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSvK6z8HV6nrJh49gsB5vqzpgtohkiFm
Date: WED, 01 Jul 2015 03:05:34 GMT
Content-Length: 0
```

5.2.9 获取桶的生命周期配置

功能介绍

获取该桶设置的生命周期配置信息。

要正确执行此操作，需要确保执行者有 GetLifecycleConfiguration 执行权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?lifecycle HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<LifecycleConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Rule>
      <ID>id</ID>
      <Prefix>prefix</Prefix>
      <Status>status</Status>
      <Expiration>
```

```
            <Date>date</Date>
        </Expiration>
        <NoncurrentVersionExpiration>
            <NoncurrentDays>days</NoncurrentDays>
        </NoncurrentVersionExpiration>
        <Transition>
         <Date>date</Date>
         <StorageClass>WARM</StorageClass>
        </Transition>
        <Transition>
         <Date>date</Date>
         <StorageClass>COLD</StorageClass>
        </Transition>
        <NoncurrentVersionTransition>
         <NoncurrentDays>30</NoncurrentDays>
         <StorageClass>WARM</StorageClass>
        </NoncurrentVersionTransition>
        <NoncurrentVersionTransition>
         <NoncurrentDays>60</NoncurrentDays>
         <StorageClass>COLD</StorageClass>
        </NoncurrentVersionTransition>
        <AbortIncompleteMultipartUpload>
            <DaysAfterInitiation>10</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如下表 5-27 描述。

表 5-27 生命周期配置元素

参数	参数类型	描述
LifecycleConfiguration	XML	**参数解释：** 生命周期配置规则的 Container。可以配置多条规则，LifecycleConfiguration 是表 5-28 的父节点。 **取值范围：** 不涉及
Rule	Container	**参数解释：** 具体某一条生命周期配置的 Container。 **取值范围：** 请详见表 5-28。

表 5-28 Rule 参数说明

参数	参数类型	描述
Transition	XML	**参数解释：** 生命周期配置中表示转换时间和转换后对象存储级别的元素（仅针对对象的最新版本）。 **取值范围：** 请详见表 5-29。 **默认取值：** 不涉及
Expiration	XML	**参数解释：** 生命周期配置中表示过期时间的 Container。 **取值范围：** 请详见表 5-30。
NoncurrentVersionTransition	XML	**参数解释：** 生命周期配置中表示对象的历史版本转换时间和转换后对象存储级别的元素。 **取值范围：** 请详见表 5-31。
NoncurrentVersionExpiration	XML	**参数解释：** 生命周期配置中表示历史版本过期时间的 Container。您可以将该动作设置在已启用多版本（或暂停）的桶，让系统删除对象的满足特定生命周期的历史版本。 **取值范围：** 请详见表 5-32。
ID	String	**参数解释：** 一条规则的标识。 **取值范围：** 长度为 0~255 的字符串。
Prefix	String	**参数解释：** 对象名前缀，用来标识哪些对象可以匹配到当前规则。 **取值范围：** 长度为 0~1024 的字符串。
Status	String	**参数解释：** 标识当前规则是否启用。 **取值范围：**

参数	参数类型	描述
		• Enabled：启用规则。 • Disabled：停用规则。

表 5-29 Transition 参数说明

参数	参数类型	描述
Date	String	**参数解释：** 指定查询某一条规则。 **取值范围：** 日期格式必须为 ISO8601 的格式，并且为 UTC 的零点。 例如：2018-01-01T00:00:00.000Z，表示将最后修改时间早于 2018-01-01T00:00:00.000Z 的对象删除或转换成其他存储类型，等于或晚于这个时间的对象不会被删除或转储。
Days	Integer	**参数解释：** 指定在对象最后修改时间的多少天后执行生命周期规则（仅针对对象的最新版本）。 **取值范围：** 不涉及
StorageClass	String	**参数解释：** 表示对象将被修改成的目标存储类别。 **取值范围：** • WARM：低频访问存储 • COLD：归档存储

表 5-30 Expiration 参数说明

参数	参数类型	描述
Date	String	**参数解释：** 指定查询某一条规则。 **取值范围：** 日期格式必须为 ISO8601 的格式，并且为 UTC 的零点。 例如：2018-01-01T00:00:00.000Z，表示将最后修改时间早于 2018-01-01T00:00:00.000Z 的对象删除或转换成其他存储类型，等于或晚于这个时间的对象不会被删除或转储。
Days	Integer	**参数解释：** 指定在对象最后修改时间的多少天后执行生命

参数	参数类型	描述
		周期规则（仅针对对象的最新版本）。 **取值范围：** 不涉及

表 5-31 NoncurrentVersionTransition 参数说明

参数	参数类型	描述
NoncurrentDays	Integer	**参数解释：** 表示对象在成为历史版本之后第几天时规则生效。单位：天。 **取值范围：** • 仅设置一种转换：最少设置 1 天。 • 设置多种转换：后者时间要比前者时间最少长 1 天。 • 设置多种转换：针对同一个对象的历史版本转深度归档存储时间需要晚于转归档存储时间，转归档存储时间需要晚于转低频访问存储时间。
StorageClass	String	**参数解释：** 表示对象将被修改成的目标存储类别。 **取值范围：** • WARM：低频访问存储 • COLD：归档存储

表 5-32 NoncurrentVersionExpiration 参数说明

参数	参数类型	描述
NoncurrentDays	Integer	**参数解释：** 表示对象在成为历史版本之后第几天时规则生效。单位：天。 **取值范围：** • 仅设置一种转换：最少设置 1 天。 • 设置多种转换：后者时间要比前者时间最少长 1 天。 • 设置多种转换：针对同一个对象的历史版本转深度归档存储时间需要晚于转归档存储时间，转归档存储时间需要晚于转低频访问存储时间。

错误响应消息

此请求可能的特殊错误如下表 5-33 描述。

表 5-33 特殊错误

错误码	描述	HTTP 状态码
NoSuchLifecycleConfiguration	桶的生命周期配置不存在	404 Not Found

其余错误已经包含在表 6-2 中。

请求示例

```
GET /?lifecycle HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:06:56 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:/Nof9FCNANfzIXDS0NDp1IfDu8I=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436BA5684FF5A10370EDB
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSEMKZSIeboCA1eAukgYOOAd7oX3ZONn
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:06:56 GMT
Content-Length: 919

<?xml version="1.0" encoding="utf-8"?>
<LifecycleConfiguration>
  <Rule>
    <ID>delete-2-days</ID>
    <Status>Enabled</Status>
    <Expiration>
      <Days>2</Days>
    </Expiration>
    <NoncurrentVersionExpiration>
      <NoncurrentDays>5</NoncurrentDays>
    </NoncurrentVersionExpiration>
    <Transition>
      <Days>30</Days>
      <StorageClass>WARM</StorageClass>
    </Transition>
    <Transition>
      <Days>60</Days>
      <StorageClass>COLD</StorageClass>
    </Transition>
    <NoncurrentVersionTransition>
      <NoncurrentDays>30</NoncurrentDays>
      <StorageClass>WARM</StorageClass>
    </NoncurrentVersionTransition>
```

```
    <NoncurrentVersionTransition>
      <NoncurrentDays>60</NoncurrentDays>
      <StorageClass>COLD</StorageClass>
    </NoncurrentVersionTransition>
    <AbortIncompleteMultipartUpload>
       <DaysAfterInitiation>10</DaysAfterInitiation>
    </AbortIncompleteMultipartUpload>
  </Rule>
</LifecycleConfiguration>
```

5.2.10 删除桶的生命周期配置

功能介绍

删除指定桶的生命周期配置信息。删除后桶中的对象不会过期，OBS 不会自动删除桶中对象。

要正确执行此操作，需要确保执行者有 PutLifecycleConfiguration 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
DELETE /?lifecycle HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: Authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: text/xml
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /?lifecycle HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:12:22 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:5DGAS7SBbMC1YTC4tNXY57Zl2Fo=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF260000016436C2550A1EEA97614A98
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSB7A0KZEBOCutgcfZvaGVthTGOJSuyk
Date: WED, 01 Jul 2015 03:12:22 GMT
```

5.2.11 设置桶的多版本状态

功能介绍

多版本功能可在用户意外覆盖或删除对象的情况下提供一种恢复手段。用户可以使用多版本功能来保存、检索和还原对象的各个版本，这样用户能够从意外操作或应用程序故障中轻松恢复数据。多版本功能还可用于数据保留和存档。

默认情况下，桶没有设置多版本功能。

当开启 WORM 开关后，桶默认开启了多版本功能，并且无法暂停。

本接口设置桶的多版本状态，用来开启或暂停桶的多版本功能。

设置桶的多版本状态为 Enabled，开启桶的多版本功能：

- 上传对象时，系统为每一个对象创建一个唯一版本号，上传同名的对象将不再覆盖旧的对象，而是创建新的不同版本号的同名对象
- 可以指定版本号下载对象，不指定版本号默认下载最新对象；
- 删除对象时可以指定版本号删除，不带版本号删除对象仅产生一个带唯一版本号的删除标记，并不删除对象；
- 列出桶内对象列表时默认列出最新对象列表，可以指定列出桶内所有版本对象列表；
- 除了删除标记外，每个版本的对象存储均需计费（不包括对象元数据）。

设置桶的多版本状态为 Suspended，暂停桶的多版本功能：

- 旧的版本数据继续保留 ；
- 上传对象时创建对象的版本号为 null，上传同名的对象将覆盖原有同名的版本号为 null 的对象；
- 可以指定版本号下载对象，不指定版本号默认下载最新对象；
- 删除对象时可以指定版本号删除，不带版本号删除对象将产生一个版本号为 null 的删除标记，并删除版本号为 null 的对象；
- 除了删除标记外，每个版本的对象存储均需计费（不包括对象元数据）。

只有桶的所有者可以设置桶的多版本状态。

请求消息样式

```
PUT /?versioning HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-Length: length

<VersioningConfiguration>
   <Status>status</Status>
</VersioningConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

在此请求中，需要在请求的消息体中配置桶的多版本状态，配置信息以 XML 格式上传。具体的配置元素如表 5-34 描述。

表 5-34 桶的多版本状态配置元素

参数名称	参数类型	是否必选	描述
VersioningConfiguration	XML	是	**参数解释：** 多版本配置的根节点，是 Status 的父节点。 **约束限制：** 不涉及
Status	String	是	**参数解释：** 标识桶的多版本状态。 **约束限制：** 不涉及

参数名称	参数类型	是否必选	描述
			取值范围： • Enabled：开启多版本控制 • Suspended：暂停多版本控制 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Date: date

Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?versioning HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:14:18 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:sc2PM13Wlfcoc/YZLK0MwsI2Zpo=
Content-Length: 89

<VersioningConfiguration>
   <Status>Enabled</Status>
</VersioningConfiguration>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643672B973EEBC5FBBF909
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSH6rPRHjQCa62fcNpCCPs7+1Aq/hKzE
Date: Date: WED, 01 Jul 2015 03:14:18 GMT
Content-Length: 0
```

5.2.12 获取桶的多版本状态

功能介绍

桶的所有者可以获取指定桶的多版本状态。

如果从未设置桶的多版本状态，则此操作不会返回桶的多版本状态。

请求消息样式

```
GET /?versioning HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length

<VersioningConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Status>status</Status>
</VersioningConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应中以消息元素的形式返回桶的多版本状态，元素的具体意义如表 5-35 所示。

表 5-35 响应消息元素

参数名称	参数类型	描述

参数名称	参数类型	描述
VersioningConfiguration	XML	**参数解释：** 多版本配置的根节点，是 Status 的父节点。 **约束限制：** 不涉及
Status	String	**参数解释：** 标识桶的多版本状态。 **约束限制：** 不涉及 **取值范围：** • Enabled：开启多版本控制 • Suspended：暂停多版本控制 **默认取值：** 不涉及

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?versioning HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:15:20 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:4N5qQIoluLO9xMY0m+8lIn/UWXM=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436BBA4930622B4FC9F17
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQIrNJ5/Ag6EPN8DAwWlPWgBc/xfBnx
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:15:20 GMT
Content-Length: 180

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VersioningConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

5.2.13 设置桶默认存储类型

功能介绍

本接口实现为创建桶或更新桶的默认存储类型配置信息。

要正确执行此操作，需要确保执行者有 PutBucketStoragePolicy 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

设置了桶的默认存储类型之后，如果上传对象、复制对象和初始化多段上传任务时未指定对象的存储类型，则该对象的存储类别默认取桶的存储类别。

请求消息样式

```
PUT /?storageClass HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Type: type
Content-Length: length
Authorization: authorization

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<StorageClass xmlns="http://obs.example.com/doc/2015-06-
30/">STANDARD</StorageClass>
```

请求消息参数

该请求在请求消息中没有带有参数。

请求消息头

该请求没有特殊的请求消息头，公共部分参见表 3-3。

请求消息元素

该操作需要附加请求消息元素来指定桶的默认存储类型，具体见表 5-36。

表 5-36 附加请求消息元素

参数	是否必选	参数类型	描述
StorageClass	是	String	**参数解释：** 指定桶默认存储类别。 **约束限制：** 不涉及 **取值范围：** • STANDARD：标准存储 • WARM：低频访问存储 • COLD：归档存储

参数	是否必选	参数类型	描述
			默认取值： 未配置桶的默认存储类别时，默认取值为STANDARD（标准存储）。

响应消息样式

```
HTTP/1.1 status_code
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，错误已经包含在表 6-2 中。

请求示例

```
PUT /?storageClass HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:18:19 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:Tf6XbndPx/yNgfAVQ6KIXr7tMj4=
Content-Length: 87

<StorageClass xmlns="http://obs.example.com/doc/2015-06-
30/">STANDARD</StorageClass>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164368E704B571F328A8797
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSIsw3tPtUn6damTI5acQmQAcEfmTwl3
Date: WED, 01 Jul 2015 03:18:19 GMT
Content-Length: 0
```

5.2.14 获取桶默认存储类型

功能介绍

获取该桶设置的默认存储类型信息。

要正确执行此操作，需要确保执行者有 GetBucketStoragePolicy 执行权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?storageClass HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请示消息中不带消息参数。

请求消息头

该请求使用公共的请求消息头，具体参见表 3-3。

请求消息元素

该请求消息不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length

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

<StorageClass xmlns="http://obs.example.com/doc/2015-06-
30/">STANDARD</StorageClass>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该响应以消息元素的形式返回桶的存储级别信息，元素的具体意义如表 5-37 所示。

表 5-37 响应消息元素

参数	参数类型	描述
StorageClass	String	**参数解释：** 表示桶的默认存储类型。 **取值范围：** • STANDARD：标准存储

参数	参数类型	描述
		• WARM：低频访问存储 • COLD：归档存储

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?storageClass HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:20:28 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:0zVTSdKG6OFCIH2dKvmsVGYCQyw=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436BE45820FDF3A65B42C
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSCju1CZy3ZfRVW5hiNd024lRFdUoqWy
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:20:28 GMT
Content-Length: 142

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

<StorageClass xmlns="http://obs.example.com/doc/2015-06-
30/">STANDARD</StorageClass>
```

5.2.15 设置桶的跨区域复制配置

功能介绍

跨区域复制是指跨不同区域中的桶自动、异步地复制对象。通过激活跨区域复制，OBS 可将新创建的对象及修改的对象从一个源桶复制到不同区域中的目标桶。

说明

配置跨区域复制需要选择 IAM 委托，配置方法请参见《对象存储用户指南》的“相关操作>创建 IAM 委托”章节中的“创建用于跨区域复制的委托”内容。

设置桶的跨区域复制，需要满足以下两个要求：

1. 要求源桶和目标桶多版本状态保持一致，否则不能设置 replication。如何设置桶的多版本，请参见 5.2.11 设置桶的多版本状态。
2. 源桶的拥有者和代理人（OBS）必须要有目标桶的写权限（目标桶需要配置 BucketPolicy），同时代理人（OBS）还要有源桶的读权限。这需要通过“BucketPolicy”来实现这个权限委托。

如何设置桶策略，请参见 5.2.1 设置桶策略。设置桶策略后，代理人（OBS）就有权限可以读取源桶的对象，也有权限将对象复制到目标桶中。

请求消息样式

```
PUT /?replication HTTP/1.1
Host: bucketname.obs.region.example.com
x-obs-date: date
Content-MD5: MD5
Authorization: authorization string
Content-Length: contentlength

<ReplicationConfiguration>
   <Agency>testAcy</Agency>
   <Rule>
      <ID>rule1</ID>
      <Prefix></Prefix>
      <Status>rule-status</Status>
      <Destination>
         <Bucket>targetbucketname</Bucket>
         <StorageClass>STANDARD</StorageClass>
      </Destination>
      <HistoricalObjectReplication>Enabled</HistoricalObjectReplication>
   </Rule>
</ReplicationConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用的消息头如下所示。

表 5-38 设置桶的复制配置请求消息头

参数	是否必选	参数类型	描述
Content-MD5	是	string	**参数解释：** 按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128bit MD5 值经过 base64 编码后得到的字符串。如何计算 Content-MD5 可参考 Java 中 Content-MD5 的计算方法示例。 **取值范围：** 不涉及 **默认取值：** 不涉及

请求消息元素

在此请求中，需要在请求的消息体中配置桶的复制配置，通知的配置信息（请求 body）以 XML 格式上传。具体的配置元素如下描述。

表 5-39 设置桶的复制配置元素

参数	是否必选	参数类型	描述
ReplicationConfiguration	是	Container	**参数解释：** 跨区域复制规则的容器。ReplicationConfiguration 是表 5-41 和 Agency 的父节点。 **约束限制：** • 单桶可以配置 1~100 条跨区域复制规则。 • 单次配置跨区域复制规则的请求 body 大小不能超过 50K。 **取值范围：** 不涉及 **默认取值：** 不涉及
Rule	是	Container	**参数解释：** 一条特定跨区域复制规则信息的容器。 **约束限制：** 一个桶上可以配置 1~100 条跨区域复制规则。 **取值范围：** 请详见表 5-41。 **默认取值：** 不涉及
Agency	是	String	**参数解释：** 用户创建用于跨区域复制的委托的名称。 **约束限制：** • 委托必须为“对象存储服务 OBS”的云服务委托，委托需要具有 Tenant Administrator 权限。 • 如果勾选了“复制使用 KMS 加密的对象”，委托还需要具有“KMS Administrator 或 Tenant

参数	是否必选	参数类型	描述
			Administrator”权限，委托的授权范围选择“指定区域项目资源”并选择源桶和目标桶所在区域。 **取值范围：** 长度为 0~64 的字符串。 **默认取值：** 不涉及

表 5-40 委托需要的 OBS 权限

操作	描述
obs:object:GetObject	获取对象内容、获取对象元数据。
obs:object:DeleteObjectVersion	删除对象、批量删除对象。
obs:object:PutObjectVersionAcl	设置对象 ACL。
obs:object:AbortMultipartUpload	取消多段上传任务。
obs:object:PutObjectAcl	设置对象 ACL。
obs:object:DeleteObject	删除对象、批量删除对象。
obs:bucket:HeadBucket	获取桶元数据。
obs:object:PutObject	PUT 上传、POST 上传、复制对象、追加写对象、初始化上传段任务、上传段、合并段
obs:object:GetObjectVersionAcl	获取对象 ACL。
obs:bucket:GetBucketVersioning	获取桶的多版本状态。
obs:bucket:ListBucketMultipartUploads	列举多段上传任务。
obs:object:ListMultipartUploadParts	列举已上传的段。
obs:object:ModifyObjectMetaData	修改对象元数据。
obs:bucket:ListBucketVersions	列举桶内多版本对象。
obs:bucket:ListBucket	列举桶内对象。
obs:object:GetObjectVersion	获取对象内容、获取对象元数据。
obs:object:GetObjectAcl	获取对象 ACL。
obs:bucket:GetReplicationConfiguration	获取桶的跨区域复制配置。

表 5-41 Rule 参数说明

参数	是否必选	参数类型	描述
ID	否	String	**参数解释：** 跨区域复制规则的规则 ID。 **约束限制：** 规则 ID 不能重复。 **取值范围：** 长度为 0~255 的字符串。 **默认取值：** 不涉及
Status	是	String	**参数解释：** 是否启用此跨区域复制规则。 **约束限制：** 不涉及 **取值范围：** • Enabled：启用规则 • Disabled：未启用规则 **默认取值：** 不涉及
Prefix	是	String	**参数解释：** 对象名的前缀。适配于一个或者多个对象。 **约束限制：** • 如果前缀配置为空，则跨区域复制规则将作用于整个桶。 • 规则之间前缀的字符不支持重叠。例如用户配置了两条规则，规则 1 的前缀为 object，规则 2 的前缀是 obj，则这两条规则就有个重叠的前缀 obj。 **取值范围：** 经过 UTF-8 编码的长度大于 0 且不超过 1024 的字符序列。 **默认取值：** 不涉及
Destination	是	Container	**参数解释：** 目标桶信息的容器。 **约束限制：**

参数	是否必选	参数类型	描述
			不涉及 **取值范围：** 请详见表 5-42。 **默认取值：** 不涉及
HistoricalObjectReplication	否	String	**参数解释：** 是否复制符合跨区域复制规则的历史对象。 **约束限制：** 不涉及 **取值范围：** • Enabled：复制符合跨区域复制规则的历史对象 • Disabled：不复制符合跨区域复制规则的历史对象 **默认取值：** 如果不设置，则默认为 Disabled。

表 5-42 Destination 参数说明

参数	是否必选	参数类型	描述
Bucket	是	String	**参数解释：** 跨区域复制目标桶的桶名，该桶用于存储被跨区域复制规则标识的对象副本。 **约束限制：** 如果在跨区域复制配置中有多条规则，这些规则必须都要标识同一个桶作为目标桶。 **取值范围：** 长度为 3~63 的字符串。 **默认取值：** 不涉及
StorageClass	否	String	**参数解释：** 表示复制到目标桶的对象将被修改成的目标存储类别。 **约束限制：** 不涉及 **取值范围：**

参数	是否必选	参数类型	描述
			• STANDARD：标准存储 • WARM：低频访问存储 • COLD：归档存储 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Server: OBS
Date:date
Content-Length: contentlength
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

在此请求的响应中不会返回特殊错误。

请求示例

```
PUT /?replication HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:39:15 +0000
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:CdeqU0Vg9xNdJMZ0PGPgh5EnkO0=
Content-MD5: l/Z8mfSX+VyV8k5EhIQz5Q==
Content-Length: 330

<ReplicationConfiguration>
  <Agency>testAcy</Agency>
  <Rule>
     <ID>Rule-1</ID>
     <Status>Enabled</Status>
     <Prefix></Prefix>
     <Destination>
        <Bucket>dstbucket</Bucket>
        <StorageClass>STANDARD</StorageClass>
     </Destination>
     <HistoricalObjectReplication>Enabled</HistoricalObjectReplication>
   </Rule>
</ReplicationConfiguration>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: B595000001644178399 32E5A2295674C
x-obs-id-2: 32AAAQAAEAABKAAQAAEAABAAAQAAEAABCStv51t2NMMx+Ou+ow7IWV4Sxo231fKe
Date: Wed, 27 Jun 2018 13:39:15 GMT
Content-Length: 0
```

5.2.16 获取桶的跨区域复制配置

功能介绍

获取指定桶的复制配置信息。执行该配置操作前需要确保执行者拥有 GetReplicationConfiguration 权限。

请求消息样式

```
GET /?replication HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization:authorization string
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Server:OBS
Content-Length: contentlength

<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://obs.example.com/doc/2006-03-01/">
  <Agency>testAcy</Agency>
  <Rule>
     <ID>rule1</ID>
     <Status>Enabled</Status>
     <Prefix></Prefix>
     <Destination>
        <Bucket>exampletargetbucket</Bucket>
        <StorageClass>WARM</StorageClass>
     </Destination>
```

```
    <HistoricalObjectReplication>Enabled</HistoricalObjectReplication>
  </Rule>
</ReplicationConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如下描述。

表 5-43 获取桶的复制配置元素

参数	参数类型	描述
ReplicationConfiguration	Container	**参数解释：** 跨区域复制规则的容器。ReplicationConfiguration 是表 5-41 和 Agency 的父节点。 **取值范围：** 不涉及
Rule	Container	**参数解释：** 一条特定跨区域复制规则信息的容器。 **取值范围：** 请详见表 5-44。
Agency	String	**参数解释：** 用户创建的委托名字。 **取值范围：** 长度为 0~64 的字符串。

表 5-44 Rule 参数说明

参数	参数类型	描述
ID	String	**参数解释：** 跨区域复制规则的规则 ID。 **取值范围：** 长度为 0~255 的字符串。
Status	String	**参数解释：** 是否启用跨区域复制规则。 **取值范围：** • Enabled：启用规则

参数	参数类型	描述
		• Disabled：未启用规则
Prefix	String	**参数解释：** 对象名的前缀。 **取值范围：** 经过 UTF-8 编码的长度大于 0 且不超过 1024 的字符序列，规则之间前缀的字符不支持重叠。例如用户配置了两条规则，规则 1 的前缀为 object，规则 2 的前缀是 obj，则这两条规则就有个重叠的前缀 obj。
Destination	Container	**参数解释：** 目标桶信息的容器。 **取值范围：** 请详见表 5-45。
HistoricalObjectReplication	String	**参数解释：** 是否复制符合跨区域复制规则的历史对象。 **取值范围：** • Enabled：复制符合跨区域复制规则的历史对象 • Disabled：不复制符合跨区域复制规则的历史对象

表 5-45 Destination 参数说明

参数	参数类型	描述
Bucket	String	**参数解释：** 存储被跨区域复制规则标识的对象副本的桶名称。 **取值范围：** 长度为 3~63 的字符串。
StorageClass	String	**参数解释：** 表示复制到目标桶的对象将被修改成的目标存储类别。 **取值范围：** • STANDARD：标准存储 • WARM：低频访问存储 • COLD：归档存储

错误响应消息

在此请求的响应中错误响应消息如下描述。

表 5-46 桶的错误响应元素

错误码	描述	HTTP 响应码	SOAP 错误码前缀
NoSuchReplicationConfiguration	跨 region 复制配置不存在	404 not found	Client

请求示例

```
GET /?replication HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:42:40 +0000
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jGHviInfRyOkT/EpySpua1hlBuY=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: B595000001644417B57D02F7EF8823152
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSHu6lz4vgk5G3E32OFcIPEZZgdOEYE/
Content-Type: application/xml
Date: Wed, 27 Jun 2018 13:42:39 GMT
Content-Length: 337

<?xml version="1.0" encoding="utf-8"?>
<ReplicationConfiguration xmlns="http://obs.example.com/doc/2006-03-01/">
  <Rule>
    <ID>Rule-1</ID>
    <Status>Enabled</Status>
    <Prefix></Prefix>
    <Destination>
      <Bucket>dstbucket</Bucket>
      <StorageClass>STANDARD</StorageClass>
    </Destination>
    <HistoricalObjectReplication>Enabled</HistoricalObjectReplication>
  </Rule>
  <Agency>testAcy</Agency>
</ReplicationConfiguration>
```

5.2.17 删除桶的跨区域复制配置

功能介绍

删除桶的复制配置。执行该配置操作前需要确保执行者拥有 DeleteReplicationConfiguration 权限。

请求消息样式

```
DELETE /?replication HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求中不使用消息元素。

响应消息样式

```
HTTP/1.1 204 No Content
Server: OBS
Date: date
Connection: keep-alive
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

在此请求的响应中不会返回特殊错误。

请求示例

```
DELETE /?replication HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:45:50 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:3ycNYD0CfMf0gOmmXzdGJ58KjHU=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 900B000001643FE6BBCC9C9F54FA7A7E
```

```
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCS8Exs52zCf9duxPLnBircmGa/JOCjec
Date: Wed, 27 Jun 2018 13:45:50 GMT
```

5.2.18 设置桶标签

功能介绍

OBS 使用 PUT 操作为一个已经存在的桶添加标签。

为桶添加标签后，该桶上所有请求产生的计费话单里都会带上这些标签，从而可以针对话单报表做分类筛选，进行更详细的成本分析。例如：某个应用程序在运行过程会往桶里上传数据，我们可以用应用名称作为标签，设置到被使用的桶上。在分析话单时，就可以通过应用名的标签来分析此应用的成本。

要正确执行此操作，需要确保执行者有 PutBucketTagging 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

接口约束

- 每个桶最多能设置 10 个标签。

请求消息样式

```
PUT /?tagging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
Content-MD5: md5
<Tagging>
   <TagSet>
      <Tag>
         <Key>Tag Name</Key>
         <Value>Tag Value</Value>
      </Tag>
   </TagSet>
</Tagging>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用的消息头如下表 5-47 所示。

表 5-47 请求消息头

消息头名称	参数类型	是否必选	描述
Content-MD5	String	是	**参数解释：** 按照 RFC 1864 标准计算出消息体的 MD5 摘

消息头名称	参数类型	是否必选	描述
			要字符串，即消息体 128-bit MD5 值经过 base64 编码后得到的字符串。 示例：n58IG6hfM7vqI4K0vnWpog== **约束限制：** 无 **取值范围：** 无 **默认取值：** 无

请求消息元素

在此请求中，需要在请求的消息体中配置桶的标签。标签的配置信息以 XML 格式上传。具体的配置元素如表 5-48。

表 5-48 桶的标签配置元素

消息头名称	参数类型	是否必选	描述
Tagging	XML	是	**参数解释：** TagSet 和 Tag 的父元素。 **约束限制：** 不涉及
TagSet	XML	是	**参数解释：** Tag 的父元素。父元素：Tagging **约束限制：** 每个桶最多能设置 10 个标签，即 TagSet 下最多有 10 个 Tag 节点。
Tag	XML	是	**参数解释：** Tag 的信息元素。父元素：TagSet **约束限制：** 每个桶最多能设置 10 个标签，即 TagSet 下最多有 10 个 Tag 节点。
Key	String	是	**参数解释：** 标签的名字。父元素：Tag **约束限制：** • 标签的键名（Key）的最大长度为 36 个字符。

消息头名称	参数类型	是否必选	描述
			• 标签的键名（Key）和键值（Value）不能包含字符“,”、“*”、“|”、“/”、“<”、“>”、“=”、“\”以及 ASCII 码 0x00--0x1F 的控制字符，在发送到服务器之前，必须将键名（Key）和键值（Value）进行 UrlEncode 编码。 **取值范围：** 长度大于 1 小于 36 的字符串 **默认取值：** 无
Value	String	是	**参数解释：** 标签的值。父元素：Tag **约束限制：** • 标签的键值（Value）的最大长度为 43 个字符。 • 标签的键名（Key）和键值（Value）不能包含字符“,”、“*”、“|”、“/”、“<”、“>”、“=”、“\”以及 ASCII 码 0x00--0x1F 的控制字符，在发送到服务器之前，必须将键名（Key）和键值（Value）进行 UrlEncode 编码。 **取值范围：** 长度大于等于 0 小于 43 的字符串。 **默认取值：** 无

响应消息样式

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Length: length
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

除了公共的错误码外，此接口还会返回一些其他的错误码。下表中列出本接口的一些常见错误，以及可能原因。如表 5-49。

表 5-49 配置桶标签错误码列表

错误码	描述	HTTP 状态码
InvalidTagError	配置桶标签时，提供了无效的 Tag。	400 Bad Request
MalformedXMLError	配置桶标签时，提供的 xml 格式错误	400 Bad Request

请求示例

例如要为桶名为 examplebucket 的桶打上键名（Key）为 TagKey(Name1)，键值（Value）为 TagValue(Value1)的标签，则发送的请求为：

```
PUT /?tagging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:22:50 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:Pf1ZyGvVYg2BzOjokZ/BAeR1mEQ=
Content-MD5: MnAEvkfQIGnBpchOE2U6Og==
Content-Length: 182

<Tagging xmlns="http://obs.example.com/doc/2015-06-30/">
  <TagSet>
    <Tag>
      <Key>TagKey%28Name1%29</Key>
      <Value>TagValue%28Value1%29</Value>
    </Tag>
  </TagSet>
</Tagging>
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF26000001643FEBA09B1ED46932CD07
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSEZp87iEirC6DggPB5cN49pSvHBWClg
Date: Wed, 27 Jun 2018 13:22:50 GMT
```

5.2.19 获取桶标签

功能介绍

OBS 使用 GET 操作来获取指定桶的标签。

要正确执行此操作，需要确保执行者有 GetBucketTagging 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?tagging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
Content-Length: length
Date: date
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Tagging xmlns="http://obs.example.com/doc/2015-06-30/">
   <TagSet>
      <Tag>
         <Key>key</Key>
         <Value>value</Value>
      </Tag>
   </TagSet>
</Tagging>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如下表 5-50。

表 5-50 桶的标签配置元素

消息头名称	参数类型	描述

消息头名称	参数类型	描述
Tagging	XML	**参数解释：** TagSet 和 Tag 的父元素。 **约束限制：** 不涉及
TagSet	XML	**参数解释：** Tag 的父元素。父元素：Tagging **约束限制：** 每个桶最多能设置 10 个标签，即 TagSet 下最多有 10 个 Tag 节点。
Tag	XML	**参数解释：** Tag 的信息元素。父元素：TagSet **约束限制：** 每个桶最多能设置 10 个标签，即 TagSet 下最多有 10 个 Tag 节点。
Key	String	**参数解释：** 标签的名字。父元素：Tag **约束限制：** • 标签的键名（Key）的最大长度为 36 个字符。 • 标签的键名（Key）和键值（Value）不能包含字符“,”、“*”、“|”、“/”、“<”、“>”、“=”、“\”以及 ASCII 码 0x00--0x1F 的控制字符，在发送到服务器之前，必须将键名（Key）和键值（Value）进行 UrlEncode 编码。 **取值范围：** 长度大于 1 小于 36 的字符串 **默认取值：** 无
Value	String	**参数解释：** 标签的值。父元素：Tag **约束限制：** • 标签的键值（Value）的最大长度为 43 个字符。 • 标签的键名（Key）和键值（Value）不能包含字符“,”、“*”、“|”、“/”、“<”、“>”、“=”、“\”以及 ASCII 码 0x00--0x1F 的控制字符，在发送到服务器之前，必须将键名（Key）和键值（Value）进行 UrlEncode 编码。

消息头名称	参数类型	描述
		取值范围： 长度大于等于 0 小于 43 的字符串。 **默认取值：** 无

错误响应消息

除了公共的错误码外，此接口还会返回一些其他的错误码。下表中列出本接口的一些常见错误，以及可能原因。如表 5-51。

表 5-51 配置桶标签的错误码列表

错误码	描述	*HTTP* 状态码
NoSuchTagSet	指定的桶没有设置标签	404 Not Found

请求示例

```
GET /?tagging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:25:44 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:H1INcyc5i0XlHqYTfuzkPxLZUPM=
```

响应示例

```
HTTP/1.1 200 OK
x-obs-request-id: 0002B7532E0000015BEB35330C5884X1
x-obs-id-2: s12w20LYNQqSb7moq4ibgJwmQRSmVQV+rFBqplOGYkXUpXeS/nOmbkyD+E35K79j
Content-Type: application/xml
Date: Wed, 27 Jun 2018 13:25:44 GMT
Content-Length: 441

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Tagging xmlns="http://obs.example.com/doc/2015-06-30/">
  <TagSet>
    <Tag>
      <Key>TagName1</Key>
      <Value>TagSetValue1</Value>
    </Tag>
  </TagSet>
</Tagging>
```

5.2.20 删除桶标签

功能介绍

OBS 使用 DELETE 操作来删除指定桶的标签。

要正确执行此操作，需要确保执行者有 DeleteBucketTagging 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
DELETE /?tagging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Length: length
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2

请求示例

```
DELETE /?tagging HTTP/1.1
User-Agent: curl/7.19.7
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:46:58 GMT
Authorization: authorization string
```

响应示例

```
HTTP/1.1 204 No Content
x-obs-request-id: 0002B7532E0000015BEB2C212E53A17L
x-obs-id-2: CqT+86nnOkB+Cv9KZoVgZ28pSgMF+uGQBUC68flvkQeq6CxoCz65wWFMNBpXvea4
Content-Length: 0
Date: Wed, 27 Jun 2018 13:46:58 GMT
```

5.2.21 设置桶配额

功能介绍

桶空间配额值必须为非负整数，单位为 Byte（字节），能设的最大值为 2^{63}-1。桶的默认配额为 0，表示没有限制桶配额。

说明

1. 桶配额设置后，如果想取消配额限制，可以把配额设置为 0。
2. 由于桶配额的校验依赖于桶存量，而桶存量是后台计算，因此桶配额可能不会及时生效，存在滞后性。可能会出现桶存量超出配额或者删除数据后存量未能及时回落的情况。
3. 桶存量查询接口请参见 5.2.23 获取桶存量信息。
4. 桶存量超出配额后再上传对象，会返回 HTTP 状态码 403 Forbidden，错误码 InsufficientStorageSpace。请扩大配额，或取消配额限制（设置为 0），或删除不需要的对象。

请求消息样式

```
PUT /?quota HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Length: length
Authorization: authorization

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Quota xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <StorageQuota>value</StorageQuota>
</Quota>
```

请求消息参数

该请求在请求消息中没有带有参数。

请求消息头

该请求没有特殊的请求消息头，公共部分参见表 3-3。

请求消息元素

该操作需要附加请求消息元素来指定桶的空间配额，具体见表 5-52。

表 5-52 附加请求消息元素

元素名称	描述	是否必选
StorageQuota	指定桶空间配额值单位为字节。 类型：Integer	是

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?quota HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:24:37 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:k/rbwnYaqYf0Ae6F0M3OJQ0dmI8=
Content-Length: 106

<Quota xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <StorageQuota>10240000</StorageQuota>
</Quota>
```

响应示例

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435E09A2BCA388688AA08
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSHbmBecv7ohDSvqaRObpxzgzJ9+l8xT
Date: WED, 01 Jul 2015 03:24:37 GMT
Content-Length: 0
```

5.2.22 获取桶配额

功能介绍

桶的拥有者可以执行获取桶配额信息的操作。桶的拥有者状态为欠费冻结时不可以查询桶配额信息。桶空间配额值的单位为 Byte（字节），0 代表不设上限。

请求消息样式

```
GET /?quota HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请示消息中不带消息参数。

请求消息头

该请求使用公共的请求消息头，具体参见表 3-3。

请求消息元素

该请求消息不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: application/xml
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Quota xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <StorageQuota>quota</StorageQuota>
</Quota>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该响应以消息元素的形式返回桶的配额信息，元素的具体意义如表 5-53 所示。

表 5-53 响应消息元素

元素名称	描述
Quota	桶的配额，包含配额量元素。

元素名称	描述
	类型：XML
StorageQuota	桶的配额量。单位字节。 类型：String

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?quota HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:27:45 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:8m4bW1gFCNeXQlfu45uO2gpo7l8=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436B55D8DED9AE26C4D18
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSs2Q5vz5AfpAJ/CMNgCfo2hmDowp7M9
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:27:45 GMT
Content-Length: 150

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Quota xmlns="http://obs.example.com/doc/2015-06-30/">
  <StorageQuota>0</StorageQuota>
</Quota>
```

5.2.23 获取桶存量信息

功能介绍

查询桶对象个数及对象占用空间，对象占用空间大小值为非负整数，单位为 Byte（字节）。

说明

由于 OBS 桶存量是后台统计，因此存量会有一定的时延，不能实时更新，因此不建议对存量做实时校验。

请求消息样式

```
GET /?storageinfo HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求不使用请求消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用请求消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<GetBucketStorageInfoResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Size>size</Size>
<ObjectNumber>number</ObjectNumber>
</GetBucketStorageInfoResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该响应中将桶存量信息以消息元素的形式返回，元素的具体含义如表 5-54 所示。

表 5-54 响应消息元素

元素名称	参数类型	描述
GetBucketStorageInfoResult	XML	**参数解释：** 保存桶存量请求结果，包含存量大小和对象个数。 取值范围： xxx。
Size	Long	**参数解释：** 返回存量大小。 取值范围： xxx。
ObjectNumber	Integer	**参数解释：** 返回对象个数。

元素名称	参数类型	描述
		取值范围： xxx。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?storageinfo HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:31:18 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:bLcdeJGYWw/eEEjMhPZx2MK5R9U=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435DD2958BFDCDB86B55E
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSitZctaPYVnat49fVMd1O+OWIP1yrg3
Content-Type: application/xml
WED, 01 Jul 2015 03:31:18 GMT
Content-Length: 206

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<GetBucketStorageInfoResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Size>25490</Size>
  <ObjectNumber>24</ObjectNumber>
</GetBucketStorageInfoResult>
```

5.2.24 设置桶清单

功能介绍

OBS 使用 PUT 操作为一个桶配置清单规则，每个桶最多可以配置 10 条清单规则。

要使用此操作，需确保执行者有 PutBucketInventoryConfiguration 操作的权限 。桶拥有者默认具有此权限，并且可以将此权限授予其他人。

请求消息样式

```
PUT /?inventory&id=configuration-id  HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
Content-Length: length
Expect: 100-continue
```

```
<InventoryConfiguration>
   <Id>configuration-id</Id>
   <IsEnabled>true</IsEnabled>
   <Filter>
        <Prefix>inventoryTestPrefix</Prefix>
   </Filter>
   <Destination>
        <Format>CSV</Format>
        <Bucket>destbucket</Bucket>
        <Prefix>dest-prefix</Prefix>
   </Destination>
   <Schedule>
         <Frequency>Daily</Frequency>
   </Schedule>
   <IncludedObjectVersions>All</IncludedObjectVersions>
   <OptionalFields>
         <Field>Size</Field>
         <Field>LastModifiedDate</Field>
         <Field>ETag</Field>
         <Field>StorageClass</Field>
         <Field>IsMultipartUploaded</Field>
         <Field>ReplicationStatus</Field>
         <Field>EncryptionStatus</Field>
   </OptionalFields>
</InventoryConfiguration>
```

请求消息参数

表 5-55 请求消息参数

参数名称	参数类型	是否必选	描述
id	String	是	**参数解释：** 清单配置的 id，必须和消息体中的清单配置 id 一致。 规格：最长 64 字节 有效字符："a-z"、"A-Z"、"0-9"、"-"、"_"和"."

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

在此请求中，需要在请求的消息体中配置桶的清单。清单的配置信息以 XML 格式上传。具体的配置元素如表 5-56。

表 5-56 桶的清单配置元素

名称	描述	是否必选
InventoryConfiguration	清单配置。 类型：Container 父节点：无 子节点：Id、IsEnabled、Filter、Destination、Schedule、IncludedObjectVersions 以及 OptionalFields	是
Id	清单配置的 id，必须和请求参数中的清单配置 id 一致。 类型：String 规格：最长 64 字节 默认值：无 有效字符："a-z"、"A-Z"、"0-9"、"-"、"_"和"." 父节点：InventoryConfiguration	是
IsEnabled	规则是否启用，如果设置为 true，则生成清单，反之不生成。 类型：Boolean 有效值：true、false 父节点：InventoryConfiguration	是
Filter	清单过滤器配置，清单只包含符合过滤器规则的对象（只支持按对象名前缀进行过滤），如果没有配置过滤器，则包含所有对象。 类型：Container 父节点：InventoryConfiguration 子节点：Prefix	否
Prefix	前缀过滤条件，清单文件中只生成以此前缀开头的对象列表。 类型：String 父节点：Filter	否
Schedule	清单文件的生成周期。 类型：Container 父节点：InventoryConfiguration 子节点：Frequency	是
Frequency	清单文件的生成周期，只支持按天和按周生成清单，第一次配置完桶清单，任务会在一个小时内启动，之后每隔一个周期启动一次。 类型：String	是

名称	描述	是否必选
	父节点：Schedule 有效值：Daily、Weekly	
Destination	清单的目标配置。 类型：Container 父节点：InventoryConfiguration	是
Format	生成的清单文件的格式，现只支持 CSV 格式。 类型：String 父节点：Destination 有效值：CSV	是
Bucket	存放清单文件的目标桶的桶名。 类型：String 父节点：Destination	是
Prefix	生成的清单文件对象名会以此前缀开头，如果不配置前缀，则生成的清单文件对象名默认以 BucketInventory 开头。 类型：String 父节点：Destination	否
IncludedObjectVersions	清单文件中包含对象的多版本配置。 类型：String 父节点：InventoryConfiguration 有效值： • All：清单会包含对象所有的版本，清单中**会增加**版本相关的字段：VersionId、IsLatest、和 DeleteMarker。 • Current：清单文件中只会列出当前版本信息，**不会**出现与多版本相关的 VersionId、IsLatest、DeleteMarker 字段。	是
OptionalFields	在此选项中可以添加一些额外的对象元数据字段，生成的清单文件中会包含 OptionalFields 中配置的字段。 类型：Container 父节点：InventoryConfiguration 子节点：Field	否
Field	可选字段类型，OptionalFields 可以包含多个 Field 元素。 类型：String	否

名称	描述	是否必选
	父节点：OptionalFields 有效值：Size、LastModifiedDate、StorageClass、ETag、IsMultipartUploaded、ReplicationStatus、EncryptionStatus。	

响应消息样式

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

除了公共的错误码外，此接口还会返回一些其他的错误码。下面列出本接口的一些常见错误，以及可能原因，如表 5-57。

表 5-57 设置桶清单错误码列表

错误码	描述	HTTP 状态码
MalformedXML	清单的 XML 配置格式错误。	400 Bad Request
InvalidArgument	无效参数。	400 Bad Request
InventoryCountOverLimit	配置清单数量超过最大限制。	400 Bad Request
PrefixExistInclusionRelationship	清单配置中的前缀存在包含关系。	400 Bad Request

请求示例

```
PUT /?inventory&id=test_id HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 08:17:10 +0000
Authorization: OBS UDSIAMSTUBTEST000001:/e2fqSfzLDb+0M36D4Op/s5KKr0=
Content-Length: 600
```

```
Expect: 100-continue

<InventoryConfiguration>
   <Id>test_id</Id>
   <IsEnabled>true</IsEnabled>
   <Filter>
        <Prefix>inventoryTestPrefix</Prefix>
   </Filter>
   <Destination>
        <Format>CSV</Format>
        <Bucket>destbucket</Bucket>
        <Prefix>dest-prefix</Prefix>
   </Destination>
   <Schedule>
         <Frequency>Daily</Frequency>
   </Schedule>
   <IncludedObjectVersions>All</IncludedObjectVersions>
   <OptionalFields>
         <Field>Size</Field>
         <Field>LastModifiedDate</Field>
         <Field>ETag</Field>
         <Field>StorageClass</Field>
         <Field>IsMultipartUploaded</Field>
         <Field>ReplicationStatus</Field>
         <Field>EncryptionStatus</Field>
   </OptionalFields>
</InventoryConfiguration>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001682C8545B0680893425D60AB83
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSIGTuRtBfo7lpHSt0ZknhdDHmllwd/p
Date: Tue, 08 Jan 2019 08:12:38 GMT
Content-Length: 0
```

5.2.25 获取桶清单

功能介绍

OBS 使用 GET 操作来获取指定桶的某个清单配置。

要正确执行此操作，需要确保执行者有 GetBucketInventoryConfiguration 权限。桶拥有者默认具有此权限，并且可以将此权限授予其他人。

请求消息样式

```
GET /?inventory&id=configuration-id HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

请求消息参数

表 5-58 请求消息参数

参数	描述	是否必选
id	需要获取的清单配置的 id。 类型：String 规格：最长 64 字节 默认值：无 有效字符："a-z"、"A-Z"、"0-9"、"-"、"_"和"."	是

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Id>configuration-id</Id>
  <IsEnabled>true</IsEnabled>
  <Destination>
    <Format>CSV</Format>
    <Bucket>destbucket</Bucket>
    <Prefix>prefix</Prefix>
  </Destination>
  <Schedule>
    <Frequency>Daily</Frequency>
  </Schedule>
  <IncludedObjectVersions>Current</IncludedObjectVersions>
  <OptionalFields>
    <Field>Size</Field>
    <Field>LastModifiedDate</Field>
    <Field>ETag</Field>
    <Field>StorageClass</Field>
    <Field>IsMultipartUploaded</Field>
    <Field>ReplicationStatus</Field>
    <Field>EncryptionStatus</Field>
```

```
  </OptionalFields>
</InventoryConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如表 5-59。

表 5-59 桶清单响应消息元素

名称	描述
InventoryConfiguration	清单配置。 类型：Container 父节点：无 子节点：Id、IsEnabled、Filter、Destination、Schedule、IncludedObjectVersions 以及 OptionalFields
Id	清单配置的 id，必须和请求参数中的清单配置 id 一致。 类型：String 规格：最长 64 字节 默认值：无 有效字符："a-z"、"A-Z"、"0-9"、"-"、"_"和"." 父节点：InventoryConfiguration
IsEnabled	规则是否启用，如果设置为 true，则生成清单，反之不生成。 类型：Boolean 有效值：true、false 父节点：InventoryConfiguration
Filter	清单过滤器配置，清单只包含符合过滤器规则的对象（只支持按对象名前缀进行过滤），如果没有配置过滤器，则包含所有对象。 类型：Container 父节点：InventoryConfiguration 子节点：Prefix
Prefix	前缀过滤条件，清单文件中只生成以此前缀开头的对象列表。 类型：String 父节点：Filter
Schedule	清单文件的生成周期。

名称	描述
	类型：Container 父节点：InventoryConfiguration 子节点：Frequency
Frequency	清单文件的生成周期，只支持按天和按周生成清单，第一次配置完桶清单，任务会在一个小时内启动，之后每隔一个周期启动一次。 类型：String 父节点：Schedule 有效值：Daily、Weekly
Destination	清单的目标配置。 类型：Container 父节点：InventoryConfiguration
Format	生成的清单文件的格式，现只支持 CSV 格式。 类型：String 父节点：Destination 有效值：CSV
Bucket	存放清单文件的目标桶的桶名。 类型：String 父节点：Destination
Prefix	生成的清单文件对象名会以此前缀开头，如果不配置前缀，则生成的清单文件对象名默认以 BucketInventory 开头。 类型：String 父节点：Destination
IncludedObjectVersions	清单文件中包含对象的多版本配置。 • 如果设置为 All，清单会包含对象所有的版本，清单中会增加版本相关的字段：VersionId、IsLatest、和 DeleteMarker。 • 如果设置为 Current，则清单文件中只会列出当前版本信息，不会出现版本相关字段。 类型：String 父节点：InventoryConfiguration 有效值：All、Current
OptionalFields	在此选项中可以添加一些额外的对象元数据字段，生成的清单文件中会包含 OptionalFields 中配置的字段。 类型：Container

名称	描述
	父节点：InventoryConfiguration 子节点：Field
Field	可选字段类型，OptionalFields 可以包含多个 Field 元素。 类型：String 父节点：OptionalFields 有效值：Size、LastModifiedDate、StorageClass、ETag、IsMultipartUploaded、ReplicationStatus、EncryptionStatus。

错误响应消息

除了公共的错误码外，此接口还会返回一些其他的错误码。下表中列出本接口的一些常见错误，以及可能原因。如表 5-60。

表 5-60 获取桶清单的错误码列表

错误码	描述	HTTP 状态码
NoSuchInventoryConfiguration	没有指定 Id 对应的清单配置。	404 Not Found

请求示例

```
GET /?inventory&id=id1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 09:32:24 +0000
Authorization: OBS UDSIAMSTUBTEST000001:ySWncC9M08jNsyXdJLSMJkpi7XM=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001682CB4C2EE6808A0D8DF9F3D00
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSBjn5O7Jv9CqvUMO0BenehRdil1n8rR
Content-Type: application/xml
Date: Tue, 08 Jan 2019 09:04:30 GMT
Content-Length: 626

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Id>id1</Id>
  <IsEnabled>true</IsEnabled>
  <Destination>
    <Format>CSV</Format>
    <Bucket>bucket</Bucket>
    <Prefix>prefix</Prefix>
  </Destination>
```

```
  <Schedule>
    <Frequency>Daily</Frequency>
  </Schedule>
  <IncludedObjectVersions>Current</IncludedObjectVersions>
  <OptionalFields>
    <Field>Size</Field>
    <Field>LastModifiedDate</Field>
    <Field>ETag</Field>
    <Field>StorageClass</Field>
    <Field>IsMultipartUploaded</Field>
    <Field>ReplicationStatus</Field>
    <Field>EncryptionStatus</Field>
  </OptionalFields>
</InventoryConfiguration>
```

5.2.26 列举桶清单

功能介绍

OBS 使用不带清单 id 的 GET 操作来获取指定桶的所有清单配置，获取到的清单配置一次性返回，不分页。

要正确执行此操作，需要确保执行者有 GetBucketInventoryConfiguration 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?inventory HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

请求消息参数

该请求消息中不使用请求消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
```

```
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListInventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
 <InventoryConfiguration>
  <Id>id</Id>
   <IsEnabled>true</IsEnabled>
   <Destination>
     <Format>CSV</Format>
     <Bucket>bucket</Bucket>
     <Prefix>prefix</Prefix>
   </Destination>
   <Schedule>
     <Frequency>Daily</Frequency>
   </Schedule>
   <IncludedObjectVersions>Current</IncludedObjectVersions>
   <OptionalFields>
     <Field>Size</Field>
     <Field>LastModifiedDate</Field>
     <Field>ETag</Field>
     <Field>StorageClass</Field>
     <Field>IsMultipartUploaded</Field>
     <Field>ReplicationStatus</Field>
     <Field>EncryptionStatus</Field>
   </OptionalFields>
 </InventoryConfiguration>
</ListInventoryConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如表 5-61。

表 5-61 桶的清单配置元素

名称	描述
ListInventoryConfiguration	桶清单配置列表。 类型：Container
InventoryConfiguration	桶清单配置，配置元素见表 5-59。 类型：Container 父节点：ListInventoryConfiguration

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?inventory HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 09:32:24 +0000
Authorization: OBS UDSIAMSTUBTEST000001:ySWncC9M08jNsyXdJLSMJkpi7XM=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001682CB4C2EE6808A0D8DF9F3D00
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSBjn5O7Jv9CqvUMO0BenehRdil1n8rR
Content-Type: application/xml
Date: Tue, 08 Jan 2019 09:04:30 GMT
Content-Length: 626

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListInventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
 <InventoryConfiguration>
  <Id>id1</Id>
   <IsEnabled>true</IsEnabled>
   <Destination>
     <Format>CSV</Format>
     <Bucket>bucket</Bucket>
     <Prefix>prefix</Prefix>
   </Destination>
   <Schedule>
     <Frequency>Daily</Frequency>
   </Schedule>
   <IncludedObjectVersions>Current</IncludedObjectVersions>
   <OptionalFields>
     <Field>Size</Field>
     <Field>LastModifiedDate</Field>
     <Field>ETag</Field>
     <Field>StorageClass</Field>
     <Field>IsMultipartUploaded</Field>
     <Field>ReplicationStatus</Field>
     <Field>EncryptionStatus</Field>
   </OptionalFields>
 </InventoryConfiguration>
</ListInventoryConfiguration>
```

5.2.27 删除桶清单

功能介绍

OBS 使用 DELETE 操作来删除指定桶的清单配置（通过清单 id 来指定清单配置）。

要正确执行此操作，需要确保执行者有 DeleteBucketInventoryConfiguration 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
DELETE /?inventory&id=configuration-id HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

请求消息参数

表 5-62 请求消息参数

参数	描述	是否必选
id	需要删除的清单配置的 id。 类型：String 规格：最长 64 字节 默认值：无 有效字符："a-z"、"A-Z"、"0-9"、"-"、"_"和"."	是

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /test?inventory&id=id1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 13:18:35 +0000
Authorization: OBS UDSIAMSTUBTEST000001:UT9F2YUgaFu9uFGMmxFj2CBgQHs=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 000001682D993B666808E265A3F6361D
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSyB46jGSQsu06m1nyIeKxTuJ+H27ooC
Date: Tue, 08 Jan 2019 13:14:03 GMT
```

5.2.28 设置桶的自定义域名

功能介绍

OBS 使用 PUT 操作为桶设置自定义域名，设置成功之后，用户访问桶的自定义域名就能访问到桶。请确保自定义域名通过 DNS 能够正确解析到 OBS 服务。

请求消息样式

```
PUT /?customdomain=domainname HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
Content-Length: 0
```

请求参数

表 5-63 请求消息参数

参数	是否必选	参数类型	描述
customdomain	是	String	**参数解释：** 桶的自定义域名。 **约束限制：** 一个桶最多可以设置 30 个自定义域名，一个自定义域名只能被一个桶使用。 **取值范围：** 长度为 0~256 的字符串。 **默认取值：** 不涉及

请求消息头

表 5-64 请求消息头

消息头名称	参数类型	是否必选	描述
Content-MD5	String	否	**参数解释：** 设置自定义域名证书时，必须要携带此头域，来校验请求消息体内容和发送时是否一致。 按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128-bit MD5 值经过 base64 编码后得到的字符串。 示例：n58IG6hfM7vqI4K0vnWpog== **约束限制：** 无 **取值范围：** 无 **默认取值：** 无

其他公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: request id
x-obs-id-2:  id
Date: date
Content-Length: 0
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?customdomain=obs.ccc.com HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Jan 2019 08:31:36 +0000
Authorization: OBS UDSIAMSTUBTEST000094:u2kJF4kENs6KlIDcAZpAKSKPtnc=
Content-Length: 0
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001697692CC5380E9D272E6D8F830
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSsfu2GXj9gScHhFnrrTPY2cFOEZuvta
Date: Wed, 13 Mar 2019 10:22:05 GMT
Content-Length: 0
```

5.2.29 获取桶的自定义域名

功能介绍

OBS 使用 GET 操作来获取桶的自定义域名。

请求消息样式

```
GET /?customdomain HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

请求参数

该请求消息中不使用请求消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
```

```
Date: date
Content-Length: 272

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketCustomDomainsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Domains>
    <DomainName>domainname</DomainName>
    <CreateTime>createtime</CreateTime>
    <CertName>exampleCertificateName</CertName>
    <ExpiredTime>2026-03-13T10:22:05.912Z</ExpiredTime>
  </Domains>
</ListBucketCustomDomainsResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该响应以消息元素的形式返回桶的自定义域名，元素的具体意义如表 5-65 所示。

表 5-65 响应消息元素

参数	参数类型	描述
ListBucketCustomDomainsResult	Container	**参数解释：** 自定义域名返回结果容器。ListBucketCustomDomainsResult 是表 5-66 的父节点。 **取值范围：** 不涉及
Domains	Container	**参数解释：** 自定义域名元素。 **取值范围：** 请详见表 5-66。

表 5-66 Domains 参数说明

参数	参数类型	描述
DomainName	String	**参数解释：** 自定义域名。 **取值范围：** 长度为 0~256 的字符串。
CreateTime	String	**参数解释：** 自定义域名的创建时间。

参数	参数类型	描述
		取值范围： • 格式要求为 UTC 时间，并符合 ISO 8601 标准。例如：2018-01-01T00:00:00.000Z，表示创建时间为 2018-01-01T00:00:00.000Z。 • 长度为 24 的字符串。
CertName	String	**参数解释：** 自定义域名绑定的证书名称。 **取值范围：** 长度为 3~63 的字符串。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?customdomain HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Jan 2019 08:31:45 +0000
Authorization: OBS UDSIAMSTUBTEST000094:veTm8B18MPLFqNyGh2wmQqovZ2U=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001697693130C80E9D2D29FA84FC2
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSM80AI9weqGUsIFJScVxSKlG4DmypX9
Content-Type: application/xml
Date: Wed, 13 Mar 2019 10:22:24 GMT
Content-Length: 272

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketCustomDomainsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Domains>
    <DomainName>obs.ccc.com</DomainName>
    <CreateTime>2019-03-13T10:22:05.912Z</CreateTime>
    <CertName>exampleCertificateName</CertName>
  </Domains>
</ListBucketCustomDomainsResult>
```

5.2.30 删除桶的自定义域名

功能介绍

OBS 使用 DELETE 操作来删除桶的自定义域名。

要使用该接口，使用者要求必须是桶的所有者，或者是桶所有者的子用户且具有删除自定义域名的权限。

请求消息样式

```
DELETE /?customdomain=domainname HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

请求参数

表 5-67 请求消息参数

参数	是否必选	参数类型	描述
customdomain	是	String	**参数解释：** 需要删除的自定义域名。 **约束限制：** 不涉及 **取值范围：** 长度为 0~256 的字符串。 **默认取值：** 不涉及

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /?customdomain=obs.ccc.com HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Jan 2019 08:27:50 +0000
Authorization: OBS UDSIAMSTUBTEST000094:ACgHHA1z+dqZhqS7D2SbU8ugluw=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 000001697694073F80E9D3D43BB10B8F
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSyjWyXNRPSnFymJW0AI59GKpW0Qm9UJ
Date: Wed, 13 Mar 2019 10:23:26 GMT
```

5.2.31 设置桶的加密配置

功能介绍

OBS 使用 PUT 操作为桶创建或更新默认服务端加密配置信息。

设置桶加密配置后，在该桶中上传对象时，会采用桶的默认加密配置对数据进行加密。目前支持配置的服务端加密方式有：KMS 托管密钥的服务端加密(SSE-KMS)。有关服务端加密方式的更多信息请参考 5.6 服务端加密章节。

要使用此操作，您必须具有执行 PutEncryptionConfiguration 操作的权限。桶拥有者默认具有此权限，并且可以将此权限授予其他人。

请求消息样式（SSE-KMS AES256）

```
PUT /?encryption  HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
Content-Length: length

<ServerSideEncryptionConfiguration>
```

```
   <Rule>
      <ApplyServerSideEncryptionByDefault>
         <SSEAlgorithm>kms</SSEAlgorithm>
         <KMSMasterKeyID>kmskeyid-value</KMSMasterKeyID>

      </ApplyServerSideEncryptionByDefault>
   </Rule>
</ServerSideEncryptionConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

在此请求中，需要在请求的消息体中配置桶的加密配置。桶加密的配置信息以 XML 格式上传。具体的配置元素如表 5-68。

表 5-68 桶的加密配置元素

参数	是否必选	参数类型	描述
ServerSideEncryptionConfiguration	是	Container	**参数解释：** 桶默认加密配置的根元素。ServerSideEncryptionConfiguration 是表 5-69 的父节点。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及
Rule	是	Container	**参数解释：** 桶默认加密配置的子元素，Rule 是 ApplyServerSideEncryptionByDefault 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-69。 **默认取值：** 不涉及

表 5-69 Rule 参数说明

参数	是否必选	参数类型	描述
ApplyServerSideEncryptionByDefault	是	Container	**参数解释：** 桶默认加密配置的子元素。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-70。 **默认取值：** 不涉及

表 5-70 ApplyServerSideEncryptionByDefault 参数说明

参数	是否必选	参数类型	描述
SSEAlgorithm	是	String	**参数解释：** 桶默认加密配置要使用的服务端加密算法。 **约束限制：** 不涉及 **取值范围：** • kms：使用 SSE-KMS 加密方式，默认使用 AES256 算法。 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 加密算法。 **默认取值：** 不涉及
KMSMasterKeyID	否	String	**参数解释：** SSE-KMS 加密方式下使用的 KMS 主密钥 ID。 **约束限制：** • 如果用户没有提供该头域，那么默认的主密钥将会被使用。 • BucketKeyEnabled 取值为 true 时，KMSMasterKeyID 取值为携带用户在 kms 上创建的主密钥 ID。 **取值范围：** • regionID:domainID(账号ID):key/key_id • key_id

参数	是否必选	参数类型	描述
			其中： • regionID 是使用密钥所属 region 的 ID； • domainID 是使用密钥所属账号的账号 ID，获取方法参见 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID； • key_id 是从数据加密服务创建的密钥 ID。 **默认取值：** 不涉及
ProjectID	否	String	**参数解释：** SSE-KMS 加密方式下 KMS 主密钥所属的项目 ID。 **约束限制：** • 如果 KMS 主密钥所属的项目为非默认项目，则需要通过该参数进行项目 ID 指定。 • 在未设置 KMSMasterKeyID 的情况下，不可设置项目 ID。 • 使用非默认 IAM 项目下的自定义密钥对桶内对象进行 SSE-KMS 加密，只有密钥拥有者可以对加密后的对象进行上传下载类操作，非密钥拥有者不能对加密后的对象进行上传下载类操作。 **取值范围：** 与 KMSMasterKeyID 相匹配的项目 ID，即 ID 为 KMSMasterKeyID 的主密钥所在项目的项目 ID。 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例（SSE-KMS AES256）

```
PUT /?encryption HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date:  Thu, 21 Feb 2019 03:05:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:DpSAlmLX/BTdjxU5HOEwflhM0WI=
Content-Length: 778

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServerSideEncryptionConfiguration xmlns="http://obs.region.example.com/doc/2015-
06-30/">
   <Rule>
      <ApplyServerSideEncryptionByDefault>
         <SSEAlgorithm>kms</SSEAlgorithm>
         <KMSMasterKeyID>4f1cd4de-ab64-4807-920a-47fc42e7f0d0</KMSMasterKeyID>
      </ApplyServerSideEncryptionByDefault>
   </Rule>
</ServerSideEncryptionConfiguration>
```

响应示例（SSE-KMS AES256）

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643670AC06E7B9A7767921
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSvK6z8HV6nrJh49gsB5vqzpgtohkiFm
Date: Thu, 21 Feb 2019 03:05:34 GMT
Content-Length: 0
```

5.2.32 获取桶的加密配置

功能介绍

OBS 使用 GET 操作来获取指定桶的加密配置。

要正确执行此操作，需要确保执行者有 GetEncryptionConfiguration 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?encryption  HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
```

```
Date: date
Authorization: authorization string
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
Content-Length: length
Date: date

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServerSideEncryptionConfiguration xmlns="http://obs.region.example.com/doc/2015-
06-30/">
   <Rule>
      <ApplyServerSideEncryptionByDefault>
         <SSEAlgorithm>kms</SSEAlgorithm>
         <KMSMasterKeyID>kmskeyid-value</KMSMasterKeyID>
         <ProjectID>projectid</ProjectID>
      </ApplyServerSideEncryptionByDefault>
   </Rule>
</ServerSideEncryptionConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如下。

表 5-71 桶的加密配置元素

参数	参数类型	描述
ServerSideEncryption Configuration	Container	**参数解释：** 桶默认加密配置的根元素。ServerSideEncryptionConfiguration 是表 5-69 的父节点。

参数	参数类型	描述
		取值范围： 不涉及
Rule	Container	**参数解释：** 桶默认加密配置的子元素，Rule 是 ApplyServerSideEncryptionByDefault 的父节点。 **取值范围：** 请详见表 5-69。

表 5-72 Rule 参数说明

参数	参数类型	描述
ApplyServerSideEncryptionByDefault	Container	**参数解释：** 桶默认加密配置的子元素。 **取值范围：** 请详见表 5-70。

表 5-73 ApplyServerSideEncryptionByDefault 参数说明

参数	参数类型	描述
SSEAlgorithm	String	**参数解释：** 桶默认加密配置要使用的服务端加密算法。 **取值范围：** • kms：使用 SSE-KMS 加密方式，默认使用 AES256 算法。 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 加密算法。
KMSMasterKeyID	String	**参数解释：** SSE-KMS 加密方式下使用的 KMS 主密钥 ID。 **取值范围：** • regionID:domainID(账号 ID):key/key_id • key_id 其中： • regionID 是使用密钥所属 region 的 ID； • domainID 是使用密钥所属账号的账号 ID，获取方法参见 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID；

参数	参数类型	描述
		• key_id 是从数据加密服务创建的密钥 ID。
ProjectID	String	**参数解释：** SSE-KMS 加密方式下 KMS 主密钥所属的项目 ID。 **取值范围：** 与 KMSMasterKeyID 相匹配的项目 ID，即 ID 为 KMSMasterKeyID 的主密钥所在项目的项目 ID。

错误响应消息

除了公共的错误码外，此接口还会返回一些其他的错误码。下表中列出本接口的一些常见错误，以及可能原因。如表 5-74。

表 5-74 获取桶加密配置的错误码列表

错误码	描述	*HTTP* 状态码
NoSuchEncryptionConfiguration	指定的桶没有设置加密配置。	404 Not Found

请求示例

```
GET /?encryption HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date:  Thu, 21 Feb 2019 03:05:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:DpSAlmLX/BTdjxU5HOEwflhM0WI=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643670AC06E7B9A7767921
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSvK6z8HV6nrJh49gsB5vqzpgtohkiFm
Date: Thu, 21 Feb 2019 03:05:34 GMT
Content-Length: 788

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServerSideEncryptionConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Rule>
      <ApplyServerSideEncryptionByDefault>
         <SSEAlgorithm>kms</SSEAlgorithm>
         <KMSMasterKeyID>4f1cd4de-ab64-4807-920a-47fc42e7f0d0</KMSMasterKeyID>
      </ApplyServerSideEncryptionByDefault>
```

```
   </Rule>
</ServerSideEncryptionConfiguration>
```

5.2.33 删除桶的加密配置

功能介绍

OBS 使用 DELETE 操作来删除指定桶的加密配置。

要正确执行此操作，需要确保执行者有 PutEncryptionConfiguration 权限。缺省情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
DELETE /?encryption HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

此请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2

请求示例

```
DELETE /examplebucket?encryption HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 13:18:35 +0000
Authorization: OBS UDSIAMSTUBTEST000001:UT9F2YUgaFu9uFGMmxFj2CBgQHs=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 000001682D993B666808E265A3F6361D
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSyB46jGSQsu06m1nyIeKxTuJ+H27ooC
Date: Tue, 08 Jan 2019 13:14:03 GMT
```

5.2.34 设置桶归档存储对象直读策略

功能介绍

归档存储对象直读是指用户可以不用恢复归档存储对象，便能直接对其进行操作。

默认情况下，桶没有开启归档存储对象直读功能。

本接口主要用来开启或关闭桶的归档存储对象直读功能。

- 设置桶的归档存储对象直读状态为 Enabled，开启桶的归档存储对象直读功能：
  - 桶内存在归档存储对象时，不管该归档存储对象是否已经恢复，均可以直接操作该归档存储对象。
  - 如果归档存储对象没有恢复，操作该归档存储对象时会增加单独计费。
- 设置桶的归档存储对象直读状态为 Disabled，关闭桶的归档存储对象直读功能：
  - 归档存储对象无法直接操作，需要先恢复。

要正确执行此操作，需要确保执行者有 PutDirectColdAccessConfiguration 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
PUT /?directcoldaccess HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-MD5: MD5
Content-Length: length

<DirectColdAccessConfiguration>
```

```
   <Status>status</Status>
</DirectColdAccessConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用的消息头如表 5-75 所示。

表 5-75 请求消息头

参数名称	是否必选	参数类型	描述
Content-MD5	是	String	**参数解释：** 按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128bit MD5 值经过 base64 编码后得到的字符串。 示例：n58IG6hfM7vqI4K0vnWpog== **取值范围：** 不涉及 **默认取值：** 不涉及

请求消息元素

在此请求中，需要在请求的消息体中配置桶的归档存储对象直读状态，配置信息以 XML 格式上传。具体的配置元素如表 5-76 所示。

表 5-76 桶的归档存储对象直读状态配置元素

参数名称	是否必选	参数类型	描述
DirectColdAccessConfiguration	是	Container	**参数解释：** 归档存储对象直读配置的根节点。DirectColdAccessConfiguration 是 Status 的父节点。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及

参数名称	是否必选	参数类型	描述
Status	是	String	**参数解释：** 标识桶的归档存储对象直读状态。 **约束限制：** 不涉及 **取值范围：** • Enabled：开启桶归档存储对象直读策略。 • Disabled：不开启桶归档存储对象直读策略。 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Date: date

Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?directcoldaccess HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Date: Fri, 26 Apr 2019 07:37:36 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:sc2PM13Wlfcoc/YZLK0MwsI2Zpo=
Content-MD5: h4A//0EKGFKAwJkH231A==
Content-Length: 92

<DirectColdAccessConfiguration>
   <Status>Enabled</Status>
</DirectColdAccessConfiguration>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 0000016A58940244809DEF00122E6802
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCStOlo1yxthHHo2GlS3WGgt1ekAIh3Vy
Date: Fri, 26 Apr 2019 07:37:36 GMT
Content-Length: 0
```

5.2.35 获取桶归档存储对象直读策略

功能介绍

桶的所有者可以获取指定桶的归档存储对象直读状态。

如果从未设置桶的归档存储对象直读状态，或者已经删除桶的归档存储对象直读状态，则此操作不会返回桶的归档存储对象直读状态。

要正确执行此操作，需要确保执行者有 GetDirectColdAccessConfiguration 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?directcoldaccess HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status code
Date: date
Content-Type: type
Content-Length: length

<DirectColdAccessConfiguration xmlns="http://obs.region.example.com/doc/2015-06-
30/">
   <Status>Enabled</Status>
</DirectColdAccessConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应中以消息元素的形式返回桶的归档存储对象直读状态，元素的具体意义如表 5-77 所示。

表 5-77 响应消息元素

参数名称	参数类型	描述
DirectColdAccessConfiguration	Container	**参数解释：** 归档存储对象直读状态信息的元素。 **取值范围：** 不涉及
Status	String	**参数解释：** 标识桶的归档存储对象直读状态。如果从未设置桶的归档存储对象直读状态，或者已经删除桶的归档存储对象直读状态，则不会返回桶的归档存储对象直读状态。 **取值范围：** • Enabled：开启桶归档存储对象直读策略。 • Disabled：未开启桶归档存储对象直读策略。 • 无 Status 参数：未开启桶归档存储对象直读策略。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /?directcoldaccess HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:15:20 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:4N5qQIoluLO9xMY0m+8lIn/UWXM=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 0000016A6C21AD79654C09D9AA45EB5D
```

```
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSmfq4hegf1QZv8/ewfveE4B566v5DZ8
Content-Type: application/xml
Date: Tue, 30 Apr 2019 02:45:07 GMT
Content-Length: 192

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DirectColdAccessConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
  <Status>Enabled</Status>
</DirectColdAccessConfiguration>
```

5.2.36 删除桶归档存储对象直读策略

功能介绍

删除指定桶的归档存储对象直读配置信息。

删除后桶内的归档存储对象不能直接读取。对未恢复或正在恢复的归档存储对象进行操作时，会返回错误 403 Forbidden。

要正确执行此操作，需要确保执行者有 DeleteDirectColdAccessConfiguration 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
DELETE /?directcoldaccess HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: Authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: text/xml
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /?directcoldaccess HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 30 Apr 2019 03:04:48 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:5DGAS7SBbMC1YTC4tNXY57Zl2Fo=
```

5.2.37 配置桶级默认 WORM 策略

功能介绍

本接口用于为指定桶配置默认保护策略和保护期限。

当您在桶内配置了桶级默认 WORM 策略以后，如果您在上传对象时没有指定保护策略和保护期限，则新上传的对象会自动应用桶级默认 WORM 策略。和配置对象级 WORM 保护策略不同的地方在于，对象级 WORM 保护策略需要您提供一个明确的时间，在这个时间之前对象都会受到保护，桶级默认 WORM 策略则要求您提供一个保护期限，实际上对象受到保护的时间点为其上传时间+您指定的保护期限。

要正确执行此操作，需要确保执行者有 "PutBucketObjectLockConfiguration"权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

说明

- 您可以修改甚至清空桶级默认 WORM 策略，但这仅对修改后上传的对象生效，修改前上传的对象的保护状态不受影响。
- 多段上传的对象在合并前不受保护，合并后受桶级默认对象策略保护，您可以在其合并后单独为其配置对象级 WORM 保护策略。

其它约束如下：

- 策略目前仅支持设置为合规模式"COMPLIANCE"
- 支持设置的保留期限为 1 天-100*365 天或 1 年~100 年。

请求消息样式

```
PUT /?object-lock HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-Type: application/xml
Content-Length: length
```

```
<ObjectLockConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <ObjectLockEnabled>Enabled</ObjectLockEnabled>
   <Rule>
      <DefaultRetention>
         <Days>integer</Days>
         <Mode>COMPLIANCE</Mode>
         <Years>integer</Years>
      </DefaultRetention>
   </Rule>
</ObjectLockConfiguration>
```

请求消息参数

该请求消息中不使用消息参数

请求消息头

该请求使用公共消息头，具体参见表 5-78。

请求消息元素

表 5-78 请求消息元素表

参数名称	是否必选	参数类型	描述
ObjectLockConfiguration	是	Container	**参数解释：** 桶级 WORM 配置的容器，ObjectLockConfiguration 是 ObjectLockEnabled 和 Rule 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-79。 **默认取值：** 不涉及

表 5-79 ObjectLockConfiguration 参数解释

参数名称	是否必选	参数类型	描述
ObjectLockEnabled	否	String	**参数解释：** 桶级 WORM 开关状态。 **约束限制：** 不涉及 **取值范围：**

参数名称	是否必选	参数类型	描述
			Enabled：启用桶级 WORM 功能。 **默认取值：** 不涉及
Rule	ObjectLockEnabled 为 Enabled 时必选，不携带则会清空当前配置的桶级默认 WORM 策略	Container	**参数解释：** 桶级 WORM 策略的规则容器，Rule 是 DefaultRetention 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-80。 **默认取值：** 不涉及

表 5-80 Rule 参数说明

参数名称	是否必选	参数类型	描述
DefaultRetention	如果有 Rule 容器则必选	Container	**参数解释：** 桶级 WORM 策略的容器，DefaultRetention 是 Mode、Days、Years 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-81。 **默认取值：** 不涉及

表 5-81 DefaultRetention 参数解释

参数名称	是否必选	参数类型	描述
Mode	如果有 DefaultRetention 容器则必选	String	**参数解释：** 桶的 WORM 保护策略。 **约束限制：** 不涉及 **取值范围：**

参数名称	是否必选	参数类型	描述
			COMPLIANCE：合规模式。 **默认取值：** 不涉及
Days	如果有DefaultRetention容器则和 Years 二选一，必须选择其中一个且不能同时指定	Integer	**参数解释：** 保护天数。单位：天。 **约束限制：** Days 和 Years 只能有一个不为 0，并且在规定取值范围内。 **取值范围：** 1~36500 **默认取值：** 不涉及
Years	如果有DefaultRetention容器则和 Days 二选一，必须选择其中一个且不能同时指定	Integer	**参数解释：** 默认的保护年数，单位：年。 **约束限制：** • 一年实际上视为保护 365 天，不考虑闰年。 • Days 和 Years 只能有一个不为 0，并且在规定取值范围内。 **取值范围：** 1~100 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

此请求可能的特殊错误如下表 5-82 描述。

表 5-82 错误响应消息

错误码	描述	HTTP 状态码
InvalidRequest	目标桶没有开启桶级 WORM 开关	400
MalformedXML	策略配置格式错误	400

其余错误已经包含在表 6-2 中。

请求示例 1

配置桶级默认 WORM 策略为保护两年

```
PUT /?object-lock HTTP/1.1
Host: bucketname.obs.region.example.com
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Type: application/xml
Content-Length: 157
<ObjectLockConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <ObjectLockEnabled>Enabled</ObjectLockEnabled>
   <Rule>
      <DefaultRetention>
         <Mode>COMPLIANCE</Mode>
         <Years>2</Years>
      </DefaultRetention>
   </Rule>
</ObjectLockConfiguration>
```

响应示例 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

请求示例 2

清空当前的桶级默认 WORM 策略配置

```
PUT /?object-lock HTTP/1.1
Host: bucketname.obs.region.example.com
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Type: application/xml
```

```
Content-Length: 157
<ObjectLockConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
</ObjectLockConfiguration>
```

响应示例 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

5.2.38 获取桶级默认 WORM 策略

功能介绍

获取该桶设置的桶级默认 WORM 策略。

要正确执行此操作，需要确保操作者有 GetBucketObjectLockConfiguration 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

说明

如果您打开了桶级 WORM 开关，但从未配置过桶级默认 WORM 策略，您依然可以使用此接口查看开关的打开情况。

请求消息样式

```
GET /?object-lock HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-Type: application/xml
Content-Length: length
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
```

```
Content-Type: application/xml
Content-Length: length

<?xml version="1.0" encoding="UTF-8"?>
<ObjectLockConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <ObjectLockEnabled>Enabled</ObjectLockEnabled>
   <Rule>
      <DefaultRetention>
         <Days>integer</Days>
         <Mode>COMPLIANCE</Mode>
         <Years>integer</Years>
      </DefaultRetention>
   </Rule>
</ObjectLockConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的桶级默认 WORM 策略元素如下表 5-83 描述。

表 5-83 桶级默认 WORM 策略元素

参数名称	参数类型	描述
ObjectLockConfiguration	Container	**参数解释：** 桶级 WORM 配置的容器，ObjectLockConfiguration 是 ObjectLockEnabled 和 Rule 的父节点。 **取值范围：** 请详见表 5-84。

表 5-84 ObjectLockConfiguration 参数说明

参数名称	参数类型	描述
ObjectLockEnabled	String	**参数解释：** 桶级 WORM 开关状态。 **取值范围：** Enabled：启用桶级 WORM 功能。
Rule	Container	**参数解释：** 桶级 WORM 策略的规则容器。如果从未配置过桶级默认 WORM 策略，则返回中不会包含此部分。Rule 是 DefaultRetention 的父节点。 **取值范围：** 请详见表 5-85。

表 5-85 Rule 参数说明

参数名称	参数类型	描述
DefaultRetention	Container	**参数解释：** 桶级默认 WORM 策略的容器，DefaultRetention 是 Mode、Days、Years 的父节点。 **取值范围：** 请详见表 5-86。

表 5-86 DefaultRetention 参数说明

参数名称	参数类型	描述
Mode	String	**参数解释：** 默认的保护策略。 **取值范围：** COMPLIANCE：合规模式。
Days	Integer	**参数解释：** 默认的保护天数。单位：天。 **取值范围：** 1~36500
Years	Integer	**参数解释：** 默认的保护年数，单位：年。 **取值范围：** 1~100

错误响应消息

此请求可能的特殊错误如下表 5-87 描述。

表 5-87 错误响应消息

错误码	描述	HTTP 状态码
InvalidRequest	目标桶没有开启桶级 WORM 开关	400

其余错误已经包含在表 6-2 中。

请求示例 1

打开了桶级 WORM 开关，未配置桶级默认 WORM 策略的情况

```
GET /?object-lock HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 0
```

响应示例 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 157

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ObjectLockConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <ObjectLockEnabled>Enabled</ObjectLockEnabled>
</ObjectLockConfiguration>
```

请求示例 2

打开了桶级 WORM 开关，且配置了桶级默认 WORM 策略的情况：

```
GET /?object-lock HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 0
```

响应示例 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 157

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ObjectLockConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <ObjectLockEnabled>Enabled</ObjectLockEnabled>
  <Rule>
    <DefaultRetention>
      <Mode>COMPLIANCE</Mode>
      <Days>10</Days>
      <Years>0</Years>
    </DefaultRetention>
  </Rule>
</ObjectLockConfiguration>
```

5.3 静态网站托管

5.3.1 设置桶的网站配置

功能介绍

OBS 允许在桶内保存静态的网页资源，如.html 网页文件、flash 文件、音视频文件等，当客户端通过桶的 Website 接入点访问这些对象资源时，浏览器可以直接解析出这些支持的网页资源，呈现给最终用户。典型的应用场景有：

- 重定向所有的请求到另外一个站点。
- 设定特定的重定向规则来重定向特定的请求。

本接口实现为桶创建或更新网站配置信息。

要正确执行此操作，需要确保执行者有 PutBucketWebsite 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

说明

1.尽量避免目标桶名中带有“.”，否则通过 HTTPS 访问时可能出现客户端校验证书出错。

2.设置桶的静态网站托管配置请求消息体的上限是 10KB。

请求消息样式

```
PUT /?website HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
<WebsiteConfiguration>
   <RedirectAllRequestsTo>
      <HostName>hostName</HostName>
   </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

在此请求中，需要在请求的消息体中配置桶的网站配置信息，配置信息以 XML 格式上传。

- 如果重定向所有请求，网站配置元素如表 5-88 描述。

表 5-88 重定向所有请求 Website 配置元素

名称	描述	是否必选
WebsiteConfiguration	网站配置的根节点。 类型：Container 父节点：无	是
RedirectAllRequestsTo	描述所有请求的重定向行为，如果这个节点出现，所有其他的兄弟节点都不能出现。 类型：Container 父节点：WebsiteConfiguration	是
HostName	描述重定向的站点名。 类型：String 父节点：RedirectAllRequestsTo	是
Protocol	描述重定向请求时使用的协议（http，https），默认使用 http 协议。 类型：String 父节点：RedirectAllRequestsTo	否

- 如果想要设定重定向规则，网站配置元素如表 5-89 描述。

表 5-89 设定重定向规则 Website 配置元素

名称	描述	是否必选
WebsiteConfiguration	Website 配置的根节点。 类型：Container 父节点：无	是
IndexDocument	*Suffix 元素*。 类型：Container 父节点：WebsiteConfiguration	是
Suffix	*Suffix* 元素被追加在对文件夹的请求的末尾（例如：*Suffix* 配置的是“index.html”，请求的是“samplebucket/images/”，返回的数据将是“samplebucket”桶内名为“images/index.html”的对象的内容）。*Suffix* 元素不能为空或者包含“/”字符。	是

名称	描述	是否必选
	类型：String 父节点：IndexDocument	
ErrorDocument	*Key* 元素。 类型：Container 父节点：WebsiteConfiguration	否
Key	当 4XX 错误出现时使用的对象的名称。这个元素指定了当错误出现时返回的页面。 类型：String 父节点：ErrorDocument 条件：父节点 ErrorDocument 存在时	否
RoutingRules	*Routing 元素*。 类型：Container 父节点：WebsiteConfiguration	否
RoutingRule	重定向规则的元素。一条重定向规则包含一个 *Condition* 和一个 *Redirect*，当 *Condition* 匹配时，*Redirect 生效*。 类型：Container 父节点：RoutingRules 元素中至少要有一个 *RoutingRule* 元素	是
Condition	描述重定向规则匹配的条件的元素。 类型：Container 父节点：RoutingRule	否
KeyPrefixEquals	描述当重定向生效时对象名的前缀。 例如： • 重定向 ExamplePage.html 对象的请求，*KeyPrefixEquals* 设为 ExamplePage.html。 类型：String 父节点：Condition 条件：父节点 Condition 存在，并且兄弟节点 HttpErrorCodeReturnedEquals 不存在。如果设定了两个条件，只有都匹配时，*Redirect* 才生效。	否
HttpErrorCodeReturnedEquals	描述 *Redirect* 生效时的 HTTP 错误码。当发生错误时，如果错误码等于这个值，那么	否

名称	描述	是否必选
	Redirect 生效。 例如： • 当返回的 http 错误码为 404 时重定向到 NotFound.html，可以将 *Condition* 中的 HttpErrorCodeReturnedEquals 设置为 404，*Redirect* 中的 ReplaceKeyWith 设置为 NotFound.html。 类型：String 父节点：Condition 条件：父节点 Condition 存在，并且兄弟节点 KeyPrefixEquals 不存在。如果设定了多个条件，需要同时匹配所有的条件，*Redirect* 才可生效。	
Redirect	重定向信息的元素。可以重定向到另一个站点、另一个页面或使用另一个协议。当事件或错误发生时，可以指定不同的返回码。 类型：Container 父节点：RoutingRule	是
Protocol	描述重定向请求时使用的协议。 类型：String 父节点：Redirect 可选值：http、https 条件：有其他兄弟节点存在时非必选	否
HostName	描述重定向请求时使用的站点名。 类型：String 父节点：Redirect 条件：有其他兄弟节点存在时非必选	否
ReplaceKeyPrefixWith	描述重定向请求时使用的对象名前缀，请求中的对象名会将 KeyPrefixEquals 的内容替换为 ReplaceKeyPrefixWith 的内容。 例如： 想把所有对 docs（目录下的对象）的请求重定向到 documents（目录下的对象），可以将 Condition 中的 KeyPrefixEquals 设置为 docs，Redirect 中的 ReplaceKeyPrefixWith 设置为 documents。那么对于对象名称为 "docs/a.html"，重定向的结果为	否

名称	描述	是否必选
	"documents/a.html"。 类型：String 父节点：Redirect 条件：有其他兄弟节点存在时非必选，不可与 *ReplaceKeyWith* 同时存在	
ReplaceKeyWith	描述重定向请求时使用的对象名，请求中的整个对象名会被替换为 ReplaceKeyWith 的内容。 例如： 想把所有对"docs"目录下的所有对象的请求重定向到"documents/error.html"，可以将 Condition 中的 KeyPrefixEquals 设置为 docs，Redirect 中的 ReplaceKeyWith 设置为 "documents/error.html"。那么对于对象名称为 "docs/a.html"和"docs/b.html"，重定向的结果都为"documents/error.html"。 类型：String 父节点：Redirect 条件：有其他兄弟节点存在时非必选，不可与 *ReplaceKeyPrefixWith* 同时存在	否
HttpRedirectCode	描述响应中的 HTTP 状态码。 类型：String 父节点：Redirect 条件：有其他兄弟节点存在时非必选	否

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：将该桶的所有请求重定向至其他桶或 URL

```
PUT /?website HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:40:29 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:pUK7Yp0yebnq4P6gqzVjoS7whoM=
Content-Length: 194

<WebsiteConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
  <RedirectAllRequestsTo>
    <HostName>www.example.com</HostName>
  </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

响应示例：将该桶的所有请求重定向至其他桶或 URL

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164360D144670B9D02AABC6
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSItqMZ/AoFUX97l1xx8s67V3cCQtXWk
Date: WED, 01 Jul 2015 03:40:29 GMT
Content-Length: 0
```

5.3.2 获取桶的网站配置

功能介绍

获取该桶设置的网站配置信息。

要正确执行此操作，需要确保执行者有 GetBucketWebsite 执行权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?website HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WebsiteConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <RedirectAllRequestsTo>
      <HostName>hostName</HostName>
   </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素与设置桶的网站配置请求的请求消息元素一致，见请求消息元素。

错误响应消息

此请求可能的特殊错误如下表 5-90 描述。

表 5-90 特殊错误

错误码	描述	HTTP 状态码
NoSuchWebsiteConfiguration	桶的 Website 配置不存在	404 Not Found

其余错误已经包含在表 6-2 中。

请求示例

```
GET /?website HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:41:54 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:Yxt1Ru+feHE0S94R7dcBp+hfLnI=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164363442EC03A8CA3DD7F5
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSFbGOmlN0BVp1kbwN3har8jbVvtKEKN
```

```
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:41:54 GMT
Content-Length: 250

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WebsiteConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <RedirectAllRequestsTo>
    <HostName>www.example.com</HostName>
  </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

5.3.3 删除桶的网站配置

功能介绍

删除指定桶的网站配置信息。

要正确执行此操作，需要确保执行者有 DeleteBucketWebsite 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
DELETE /?website HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /?website HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:44:37 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:AZ1b0N5eLknxNOe/c0BISV1bEqc=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF260000016436378623OE2001DC0807
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSFUG4fEyDRgzUiEY2i71bJndBCy+wUZ
Date: WED, 01 Jul 2015 03:44:37 GMT
```

5.3.4 设置桶的 CORS 配置

功能介绍

CORS（Cross Origin Resource Sharing），即跨域资源共享，是 W3C 标准化组织提出的一种规范机制，允许客户端的跨域请求的配置。在通常的网页请求中，由于安全策略 SOP（Same Origin Policy）的存在，一个网站的脚本和内容是不能与另一个网站的脚本和内容发生交互的。

OBS 允许在桶内保存静态的网页资源，在正确的使用下，OBS 的桶可以成为网站资源（请参见 5.3.1 设置桶的网站配置）。只有进行了适当的 CORS 配置，OBS 中的网站才能响应另一个网站的跨域请求。

典型的应用场景如下：

- 您可以使用 CORS 支持，使用 JavaScript 和 HTML 5 来构建 Web 应用，直接访问 OBS 中的资源，而不再需要代理服务器做中转。
- 可以使用 HTML 5 中的拖拽功能，直接向 OBS 上传文件，展示上传进度，或是直接从 Web 应用中更新内容。
- 托管在不同域中的外部网页、样式表和 HTML 5 应用，现在可以引用存储在 OBS 中的 Web 字体或图片，让这些资源能被多个网站共享。

要正确执行此操作，需要确保执行者有 PutBucketCORS 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
PUT /?cors HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
Content-MD5: MD5
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration>
   <CORSRule>
      <ID>id</ID>
      <AllowedMethod>method</AllowedMethod>
      <AllowedOrigin>origin</AllowedOrigin>
      <AllowedHeader>header</AllowedHeader>
      <MaxAgeSeconds>seconds</MaxAgeSeconds>
      <ExposeHeader>header</ExposeHeader>
   </CORSRule>
</CORSConfiguration>
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头外加 CORS 请求消息头，具体参见表 3-3 和表 5-91。

表 5-91 CORS 请求消息头

消息头名称	消息头类型	是否必选	描述
Content-MD5	String	是	**参数解释：** 按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128-bit MD5 值经过 base64 编码后得到的字符串。 类型：示例：n58IG6hfM7vqI4K0vnWpog== **约束限制：** 无 **取值范围：** 无 **默认取值：** 无

请求消息元素

在此请求中，需要在请求的消息体中配置桶的 CORS 配置信息。配置信息以 XML 格式上传，具体的配置元素如表 5-92 描述。

表 5-92 CORS 配置元素

元素名称	元素类型	是否必选	描述
CORSConfiguration	Container	是	**参数解释：** CORSRules 的根节点。 父节点：无 **约束限制：** 最大不超过 64KB。 **取值范围：** 无 **默认取值：** 无
CORSRule	Container	是	**参数解释：** CORS 规则。 父节点：CORSConfiguration **约束限制：** CORSConfiguration 下可最多包含 100 个 CORS 规则。 **取值范围：** 无 **默认取值：** 无
ID	String	否	**参数解释：** 一条 CORS 规则的标识。 父节点：CORSRule **约束限制：** ID 由不超过 255 个字符的字符串组成。 **取值范围：** 1~255 个字符的字符串组成。 **默认取值：** 无
AllowedMethod	String	是	**参数解释：** 指定允许的跨域请求 HTTP 方法，即桶和对象的几种操作类型。 父节点：CORSRule **约束限制：** 无

元素名称	元素类型	是否必选	描述
			取值范围： 支持以下 HTTP 方法： • GET • PUT • HEAD • POST • DELETE **默认取值：** 无
AllowedOrigin	String	是	**参数解释：** 指定允许的跨域请求的来源，即允许来自该域名下的请求访问该桶。 父节点：CORSRule。 **约束限制：** 表示域名的字符串，仅支持英文域名。通过正则表达式进行匹配，每个匹配规则允许使用最多一个“*”通配符。例如：https://*.vbs.example.com。 **取值范围：** 符合 CORS 协议的取值范围，长度为[0-20480]个字符。 **默认取值：** 无
AllowedHeader	String	否	**参数解释：** 指定允许的跨域请求的头域。配置 CORS 请求中允许携带的“Access-Control-Request-Headers”头域。如果一个请求带了“Access-Control-Request-Headers”头域，则只有匹配上 AllowedHeader 中的配置才认为是一个合法的 CORS 请求（通过正则表达式进行匹配）。 父节点：CORSRule **约束限制：** 最多可填写一个“*”通配符，不支持&、:、<、空格以及中文字符。 **取值范围：** 符合 CORS 协议的取值范围，长度为[0-20480]个字符。

元素名称	元素类型	是否必选	描述
			默认取值： 无
MaxAgeSeconds	Integer	否	**参数解释：** 请求来源的客户端可以对跨域请求返回结果的缓存时间。 父节点：CORSRule **约束限制：** 每个 CORSRule 可以包含至多一个 MaxAgeSeconds。 **取值范围：** 大于等于 0 的整型数，单位：秒。 **默认取值：** 3000
ExposeHeader	String	否	**参数解释：** CORS 规则允许响应中可返回的附加头域，给客户端提供额外的信息。默认情况下浏览器只能访问以下头域：Content-Length、Content-Type，如果需要访问其他头域，需要在附加头域中配置。 父节点：CORSRule **约束限制：** 不支持*、&、:、<、空格以及中文字符。 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无

响应消息样式

```
HTTP/1.1 status_code

Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /?cors HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:51:52 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:lq7BGoqE9yyhdEwE6KojJ7ysVxU=
Content-MD5: NGLzvw81f/A2C9PiGO0aZQ==
Content-Length: 617

<?xml version="1.0" encoding="utf-8"?>
<CORSConfiguration>
  <CORSRule>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedOrigin>www.example.com</AllowedOrigin>
    <AllowedHeader>AllowedHeader_1</AllowedHeader>
    <AllowedHeader>AllowedHeader_2</AllowedHeader>
    <MaxAgeSeconds>100</MaxAgeSeconds>
    <ExposeHeader>ExposeHeader_1</ExposeHeader>
    <ExposeHeader>ExposeHeader_2</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
```

响应示例

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643627112BD03512FC94A4
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSYi6wLC4bkrvuS9sqnlRjxK2a5Fe3ry
Date: WED, 01 Jul 2015 03:51:52 GMT
Content-Length: 0
```

5.3.5 获取桶的 CORS 配置

功能介绍

获取指定桶的 CORS 配置信息。

要正确执行此操作，需要确保执行者有 GetBucketCORS 权限。默认情况下只有桶的所有者可以执行此操作，也可以通过设置桶策略或用户策略授权给其他用户。

请求消息样式

```
GET /?cors HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type:  application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CORSConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <CORSRule>
      ...
   </CORSRule>
</CORSConfiguration>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

在此请求返回的响应消息体中包含的配置元素如下表 5-93 描述。

表 5-93 CORS 配置元素

名称	描述
CORSConfiguration	CORSRules 的根节点，最大不超过 64 KB。 类型：Container 父节点：无。
CORSRule	CORS 规则，CORSConfiguration 下可最多包含 100 个规则。

名称	描述
	类型：Container 父节点：CORSConfiguration。
ID	一条 Rule 的标识，由不超过 255 个字符的字符串组成。 类型：String 父节点：CORSRule。
AllowedMethod	CORS 规则允许的 Method。 类型：String 有效值：GET、PUT、HEAD、POST 、DELETE 父节点：CORSRule。
AllowedOrigin	CORS 规则允许的 Origin（表示域名的字符串），长度为[0-20480]个字符，每一个 AllowedOrigin 可以带最多一个“*”通配符。 类型：String 父节点：CORSRule。
AllowedHeader	配置 CORS 请求中允许携带的“Access-Control-Request-Headers”头域，长度为[0-20480]个字符。如果一个请求带了“Access-Control-Request-Headers”头域，则只有匹配上 AllowedHeader 中的配置才认为是一个合法的 CORS 请求。每一个 AllowedHeader 可以带最多一个“*”通配符，不可出现空格。 类型：String 父节点：CORSRule。
MaxAgeSeconds	客户端可以缓存的 CORS 响应时间，以秒为单位。 每个 CORSRule 可以包含至多一个 MaxAgeSeconds。 类型：Integer 父节点：CORSRule。
ExposeHeader	CORS 响应中带的附加头域，给客户端提供额外的信息，不可出现空格。 类型：String 父节点：CORSRule。

错误响应消息

此请求可能的特殊错误如下表 5-94 描述。

表 5-94 特殊错误

错误码	描述	HTTP 状态码

错误码	描述	HTTP 状态码
NoSuchCORSConfiguration	桶的 CORS 配置不存在	404 Not Found

其余错误已经包含在表 6-2 中。

请求示例

```
GET /?cors HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:54:36 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:WJGghTrPQQXRuCx5go1fHyE+Wwg=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164363593F10738B80CACBE
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSpngvwC5TskcLGh7Fz5KRmCFIayuY8p
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:54:36 GMT
Content-Length: 825

<?xml version="1.0" encoding="utf-8"?>
<CORSConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <CORSRule>
    <ID>783fc6652cf246c096ea836694f71855</ID>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>

    <AllowedOrigin>obs.example.com</AllowedOrigin>
    <AllowedOrigin>www.example.com</AllowedOrigin>
    <AllowedHeader>AllowedHeader_1</AllowedHeader>
    <AllowedHeader>AllowedHeader_2</AllowedHeader>
    <MaxAgeSeconds>100</MaxAgeSeconds>
    <ExposeHeader>ExposeHeader_1</ExposeHeader>
    <ExposeHeader>ExposeHeader_2</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
```

5.3.6 删除桶的 CORS 配置

功能介绍

删除指定桶的 CORS 配置信息。删除后桶以及桶中的对象将不能再被其他网址发送的请求访问。

要正确执行此操作，需要确保执行者有 PutBucketCORS 权限(在桶策略中配置删除桶 CORS 权限时和设置桶 CORS 使用同一个 Action)。

请求消息样式

```
DELETE /?cors HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体参见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type:  application/xml
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /?cors HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:56:41 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mKUs/uIPb8BP0ZhvMd4wEy+EbiI=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF26000001643639F290185BB27F793A
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSLWMRFJfckapW+ktT/+1AnAz7XlNU0b
Date: WED, 01 Jul 2015 03:56:41 GMT
```

5.3.7 OPTIONS 桶-OptionsBucket

功能介绍

OPTIONS，称为预请求，是客户端发送给服务端的一种请求，通常被用于检测客户端是否具有对服务端进行操作的权限。只有当预请求成功返回，客户端才开始执行后续的请求。

OBS 允许在桶内保存静态的网页资源，在正确的使用下，OBS 的桶可以成为网站资源。在这种使用场景下，OBS 中的桶作为服务端，需要处理客户端发送的 OPTIONS 预请求。

要处理 OPTIONS，OBS 的桶必须已经配置 CORS，关于 CORS 的使用说明，请参见章节 5.3.4 设置桶的 CORS 配置。

与 OPTIONS 对象的区别

OPTIONS 对象需在 URL 中指定对象名；OPTIONS 桶提交的 URL 为桶域名，无需指定对象名。两者的请求行分别为：

```
OPTIONS /object HTTP/1.1
OPTIONS / HTTP/1.1
```

请求消息样式

```
OPTIONS / HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Origin: origin
Access-Control-Request-Method: method
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用的消息头如下表 5-95 所示。

表 5-95 OPTIONS 请求消息头

消息头名称	描述	是否必选

消息头名称	描述	是否必选
Origin	预请求指定的跨域请求 Origin（通常为域名）。 类型：String	是
Access-Control-Request-Method	实际请求可以带的 HTTP 方法，可以带多个方法头域。 类型：String 有效值：GET、PUT、HEAD、POST 、DELETE	是
Access-Control-Request-Headers	实际请求可以带的 HTTP 头域，可以带多个头域。 类型：String	否

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: application/xml
Access-Control-Allow-Origin: origin
Access-Control-Allow-Methods: method
Access-Control-Allow-Header: header
Access-Control-Max-Age: time
Access-Control-Expose-Headers: header
Date: date
Content-Length: length
```

响应消息头

该响应使用的消息头如下表 5-96 所示。

表 5-96 CORS 响应消息头

消息头名称	描述
Access-Control-Allow-Origin	如果请求的 Origin 满足服务端的 CORS 配置，则在响应中包含这个 Origin。 类型：String
Access-Control-Allow-Headers	如果请求的 headers 满足服务端的 CORS 配置，则在响应中包含这个 headers。 类型：String
Access-Control-Max-Age	服务端 CORS 配置中的 MaxAgeSeconds。 类型：Integer
Access-Control-Allow-	如果请求的 Access-Control-Request-Method 满足服务端的

消息头名称	描述
Methods	CORS 配置，则在响应中包含这条 rule 中的 Methods。 类型：String 有效值：GET、PUT、HEAD、POST 、DELETE
Access-Control-Expose-Headers	服务端 CORS 配置中的 ExposeHeader。 类型：String

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

此请求可能的特殊错误如下表 5-97 描述。

表 5-97 特殊错误

错误码	描述	HTTP 状态码
Bad Request	Invalid Access-Control-Request-Method: null 桶配置了 CORS，OPTIONS 桶时，没有加入 method 头域。	400 BadRequest
Bad Request	Insufficient information. Origin request header needed. 桶配置了 CORS，OPTIONS 桶时，没有加入 origin 头域。	400 BadRequest
AccessForbidden	CORSResponse: This CORS request is not allowed. This is usually because the evaluation of Origin, request method / Access-Control-Request-Method or Access-Control-Request-Headers are not whitelisted by the resource's CORS spec. 桶配置了 CORS，OPTIONS 桶时，Origin、method、Headers 与任一 rule 匹配不上。	403 Forbidden

其余错误已经包含在表 6-2 中。

请求示例

```
OPTIONS / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:02:15 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:7RqP1vjemo6U+Adv9/Y6eGzWrzA=
```

```
Origin: www.example.com
Access-Control-Request-Method: PUT
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436314E8FF936946DBC9C
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT,DELETE
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: ExposeHeader_1,ExposeHeader_2
Access-Control-Allow-Credentials: true
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTlYimJvOyJncCLNm5y/iz6MAGLNxTuS
Date: WED, 01 Jul 2015 04:02:15 GMT
Content-Length: 0
```

5.3.8 OPTIONS 对象-OptionsObject

功能介绍

请参见章节 5.3.7 OPTIONS 桶-OptionsBucket。

与 OPTIONS 桶的区别

OPTIONS 对象需在 URL 中指定对象名；OPTIONS 桶提交的 URL 为桶域名，无需指定对象名。两者的请求行分别为：

```
OPTIONS /object HTTP/1.1
OPTIONS / HTTP/1.1
```

请求消息样式

```
OPTIONS /object HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Origin: origin
Access-Control-Request-Method: method
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该请求使用的消息头如下表 5-98 所示。

表 5-98 OPTIONS 请求消息头

消息头名称	描述	是否必选

消息头名称	描述	是否必选
Origin	预请求指定的跨域请求 Origin（通常为域名）。 类型：String	是
Access-Control-Request-Method	实际请求可以带的 HTTP 方法，可以带多个方法头域。 类型：String 有效值：GET、PUT、HEAD、POST 、DELETE	是
Access-Control-Request-Headers	实际请求可以带的 HTTP 头域，可以带多个头域。 类型：String	否

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: type
Access-Control-Allow-Origin: origin
Access-Control-Allow-Methods: method
Access-Control-Allow-Header: header
Access-Control-Max-Age: time
Access-Control-Expose-Headers: header
Date: date
Content-Length: length
```

响应消息头

该请求使用的消息头如下表 5-99 所示。

表 5-99 CORS 请求消息头

消息头名称	描述
Access-Control-Allow-Origin	如果请求的 Origin 满足服务端的 CORS 配置，则在响应中包含这个 Origin。 类型：String
Access-Control-Allow-Headers	如果请求的 headers 满足服务端的 CORS 配置，则在响应中包含这个 headers。 类型：String
Access-Control-Max-Age	服务端 CORS 配置中的 MaxAgeSeconds。 类型：Integer
Access-Control-Allow-	如果请求的 Access-Control-Request-Method 满足服务端的

消息头名称	描述
Methods	CORS 配置，则在响应中包含这条 rule 中的 Methods。 类型：String 有效值：GET、PUT、HEAD、POST 、DELETE
Access-Control-Expose-Headers	服务端 CORS 配置中的 ExposeHeader。 类型：String

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

此请求可能的特殊错误如下表 5-100 描述。

表 5-100 特殊错误

错误码	描述	HTTP 状态码
Bad Request	Invalid Access-Control-Request-Method: null 桶配置了 CORS，OPTIONS 桶时，没有加入 method 头域。	400 BadRequest
Bad Request	Insufficient information. Origin request header needed. 桶配置了 CORS，OPTIONS 桶时，没有加入 origin 头域。	400 BadRequest
AccessForbidden	CORSResponse: This CORS request is not allowed. This is usually because the evaluation of Origin, request method / Access-Control-Request-Method or Access-Control-Request-Headers are not whitelisted by the resource's CORS spec. 桶配置了 CORS，OPTIONS 桶时，Origin、method、Headers 与任一 rule 匹配不上。	403 Forbidden

其余错误已经包含在表 6-2 中。

请求示例

```
OPTIONS /object_1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:02:19 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:bQZG9c2aokAJsHOOkuVBK6cHZZQ=
```

```
Origin: www.example.com
Access-Control-Request-Method: PUT
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643632D12EFCE1C1294555
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT,DELETE
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: ExposeHeader_1,ExposeHeader_2
Access-Control-Allow-Credentials: true
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCS+DXV4zZetbTqFehhEcuXywTa/mi3T3
Date: WED, 01 Jul 2015 04:02:19 GMT
Content-Length: 0
```

5.4 对象操作

5.4.1 PUT 上传

功能介绍

用户在 OBS 系统中创建了桶之后，可以采用 PUT 操作的方式将对象上传到桶中。上传对象操作是指在指定的桶内增加一个对象，执行该操作需要用户拥有桶的写权限。

说明

同一个桶中存储的对象名是唯一的。

在桶未开启多版本的情况下，如果在指定的桶内已经有相同的对象键值的对象，用户上传的新对象会覆盖原来的对象；为了确保数据在传输过程中没有遭到破坏，用户可以在请求消息头中加入 Content-MD5 参数。在这种情况下，OBS 收到上传的对象后，会对对象进行 MD5 校验，如果不一致则返回出错信息。

用户还可以在上传对象时指定 x-obs-acl 参数，设置对象的权限控制策略。如果匿名用户在上传对象时未指定 x-obs-acl 参数，则该对象默认可以被所有 OBS 用户访问。

该操作支持服务端加密功能。

单次上传对象大小范围是[0, 5GB]，如果需要上传超过 5GB 的大文件，需要通过 5.5 多段操作来分段上传。

OBS 没有文件夹的概念。为了使用户更方便进行管理数据，OBS 提供了一种方式模拟文件夹：通过在对象的名称中增加“/”，例如“test/123.jpg”。此时，“test”就被模拟成了一个文件夹，“123.jpg”则模拟成“test”文件夹下的文件名了，而实际上，对象名称（Key）仍然是“test/123.jpg”。此类命名方式的对象，在控制台上会以文件夹的形式展示。当您上传此类方式命名的对象时，如果其大小不为 0，控制台上会展示为空文件夹，但是存储总用量为对象大小。

与 POST 上传的区别

PUT 上传中参数通过请求头域传递；POST 上传则作为消息体中的表单域传递。

PUT 上传需在 URL 中指定对象名；POST 上传提交的 URL 为桶域名，无需指定对象名。两者的请求行分别为：

```
PUT /ObjectName HTTP/1.1
POST / HTTP/1.1
```

说明

使用 PUT 上传请求时，消息体中如果使用 POST 格式，则上传到 OBS 中的对象会以表单形式呈现。

关于 POST 上传的更多详细信息，请参考 5.4.2 POST 上传。

多版本

如果桶的多版本状态是开启的，系统会自动为对象生成一个唯一的版本号，并且会在响应报头 x-obs-version-id 返回该版本号。如果桶的多版本状态是暂停的，则对象的版本号为 **null**。关于桶的多版本状态，参见 5.2.11 设置桶的多版本状态。

WORM

如果桶的 WORM 开关是开启的，则可以为对象配置 WORM。您可以通过携带头域 x-obs-object-lock-mode 和 x-obs-object-lock-retain-until-date 在上传对象的同时指定对象的保护策略，如果您不携带这些头域，但配置了桶级默认 WORM 策略，则新上传的对象会自动应用默认策略。您还可以在上传后配置或修改对象级 WORM 保护策略。

说明

在桶的 WORM 开关开启时，系统会自动打开多版本功能。WORM 保护是基于对象版本号的，配置 WORM 的版本受到 WORM 保护，没有配置 WORM 的版本可正常删除。例如，test.txt 001 受到 WORM 保护。此时再次上传同名文件，产生新的对象版本 test.txt 002，test.txt 002 并未配置 WORM，那么 test.txt 002 就不受保护可以正常删除。当您下载对象时，不指定版本号下载的是最新对象，也就是 test.txt 002。

请求消息样式

```
PUT /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Type: application/xml
Content-Length: length
Authorization: authorization
Date: date
<Optional Additional Header>
<object Content>
```

请求消息参数

该请求消息中不使用参数。

请求消息头

该请求使用公共的消息头，具体请参见表 3-3。该请求可以使用附加的消息头，具体如表 5-101 所示。

说明

OBS 支持在上传对象时在请求里携带 HTTP 协议规定的 6 个请求头：Cache-Control、Expires、Content-Encoding、Content-Disposition、Content-Type、Content-Language。如果上传 Object 时设置了这些请求头，OBS 会直接将这些头域的值保存下来。这 6 个值也可以通过 OBS 提供的修改对象元数据 API 接口进行修改。在该 Object 被下载或者 HEAD 的时候，这些保存的值将会被设置到对应的 HTTP 头域中返回客户端。

表 5-101 请求消息头

消息头名称	消息头类型	是否必选	描述
Content-MD5	String	否	**参数解释：** 按照 RFC 1864 标准计算出消息体的 MD5 摘要字符串，即消息体 128-bit MD5 值经过 base64 编码后得到的字符串。 示例：n58IG6hfM7vqI4K0vnWpog==。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-acl	String	否	**参数解释：** 创建对象时，可以加上此消息头设置对象的权限控制策略，使用的策略为预定义的常用策略。 **约束限制：** 预定义策略需要以字符串形式呈现。 **取值范围：** • private • public-read • public-read-write **默认取值：** private
x-obs-grant-read	String	否	**参数解释：** 创建对象时，使用此头域授权租户下所有用户有读对象和获取对象元数据的权限。

消息头名称	消息头类型	是否必选	描述
			示例：x-obs-grant-read: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-read-acp	String	否	**参数解释：** 创建对象时，使用此头域授权租户下所有用户有获取对象 ACL 的权限。 示例：x-obs-grant-read-acp: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-write-acp	String	否	**参数解释：** 创建对象时，使用此头域授权 domain 下所有用户有写对象 ACL 的权限。 示例：x-obs-grant-write-acp: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-full-control	String	否	**参数解释：** 创建对象时，使用此头域授权 domain 下所有用户有读对象、获取对象元数据、获取对象 ACL、写对象 ACL 的权限。

消息头名称	消息头类型	是否必选	描述
			示例：x-obs-grant-full-control: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-storage-class	String	否	**参数解释：** 创建对象时，可以加上此头域设置对象的存储类型。如果未设置此头域，则以桶的默认存储类型作为对象的存储类型。 示例：x-obs-storage-class: STANDARD **约束限制：** 设置该参数的值时请注意大小写敏感。 **取值范围：** • STANDARD：标准存储 • WARM：低频存储 • COLD：归档存储 **默认取值：** 默认继承桶的存储类型。
x-obs-meta-*	String	否	**参数解释：** 创建对象时，可以在 HTTP 请求中加入以“x-obs-meta-”开头的消息头，用来加入自定义的元数据，以便对对象进行自定义管理。当用户获取此对象或查询此对象元数据时，加入的自定义元数据将会在返回消息的头中出现。 示例：x-obs-meta-test: test metadata **约束限制：** 自定义元数据 key-value 对都必须符合 US-ASCII。 **取值范围：** 无 **默认取值：** 无
x-obs-	String	否	**参数解释：**

消息头名称	消息头类型	是否必选	描述
website-redirect-location			当桶设置了 Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的 URL，OBS 将这个值从头域中取出，保存在对象的元数据中。 例如，重定向请求到桶内另一对象： x-obs-website-redirect-location:/anotherPage.html 或重定向请求到一个外部 URL： x-obs-website-redirect-location:http://www.example.com/ **约束限制：** 必须以“/”、“http://”或“https://”开头，长度不超过 2KB。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption	String	否	**参数解释：** 配置服务端加密方式。示例：x-obs-server-side-encryption: kms **约束限制：** 无 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	否	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。

消息头名称	消息头类型	是否必选	描述
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密使用的密钥。该密钥用于加密对象。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-

消息头名称	消息头类型	是否必选	描述
			encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥的 MD5 值。 **默认取值：** 无
success-action-redirect	String	否	**参数解释：** 用于指定当此次请求操作成功响应后的重定向的地址。 • 如果此参数值有效且操作成功，响应码为 303，Location 头域由此参数以及桶名、对象名、对象的 ETag 组成。 • 如果此参数值无效，则 OBS 忽略此参数的作用，Location 头域为对象地址，响应码根据操作成功或失败正常返回。 **约束限制：** 需要为 URL 格式，如 http://domainname、https://domainname。 **取值范围：** URL **默认取值：** 无
x-obs-expires	Integer	否	**参数解释：** 表示对象的过期时间，单位是天。过期之后对象会被自动删除。（从对象创建时间开始计算） 此字段对于每个对象支持上传时配置，也支持后期通过修改元数据接口修改。 示例：x-obs-expires:3 **约束限制：** 设置的天数计算出的过期时间不能早于当前时间，如 10 天前上传的对象，不能设置小于 10 的值。 **取值范围：** 大于 0 的整数值。 **默认取值：** 无
x-obs-	String	否，携带	**参数解释：**

消息头名称	消息头类型	是否必选	描述
object-lock-mode		x-obs-object-lock-retain-until-date 头域时必带	要应用于此对象的 WORM 模式。 示例：x-obs-object-lock-mode:COMPLIANCE **约束限制：** • 仅支持 COMPLIANCE，即合规模式。 • 该参数需要和 x-obs-object-lock-retain-until-date 一同使用。 **取值范围：** COMPLIANCE **默认取值：** 无
x-obs-object-lock-retain-until-date	String	否，携带 x-obs-object-lock-mode 头域时必带	**参数解释：** 此对象的 WORM 策略过期的截止时间。 示例：x-obs-object-lock-retain-until-date:2015-07-01T04:11:15.297Z **约束限制：** • 格式要求为 UTC 时间，并符合 ISO 8601 标准。如：2015-07-01T04:11:15.297Z • 该参数需要和 x-obs-object-lock-mode 一同使用。 **取值范围：** 需要大于当前时间。 **默认取值：** 无

请求消息元素

该请求消息中不使用消息元素，在消息体中带的是对象的数据。

响应消息样式

```
HTTP/1.1 status_code
Content-Length: length
Content-Type: type
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如表 5-102 中的消息头。

表 5-102 附加响应消息头

消息头名称	消息头类型	描述
x-obs-version-id	String	**参数解释：** 对象的版本号。如果桶的多版本状态为开启，则会返回对象的版本号。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 服务端的加密方式。示例：x-obs-server-side-encryption:kms **约束限制：** 不涉及 **取值范围：** • kms：使用 SSE-KMS 加密方式。 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法。 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的

消息头名称	消息头类型	描述
		KMS 主密钥所属的项目 ID（非企业项目 ID）。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** 密钥的 MD5 值。 **默认取值：** 无
x-obs-storage-class	String	**参数解释：** 对象的存储类别。 **约束限制：** 对象为非标准存储对象时，会返回此头域。 **取值范围：** • WARM • COLD **默认取值：** 无

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

该请求的返回无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：上传对象

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:11:15 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:gYqplLq30dEX7GMi2qFWyjdFsyw=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

响应示例：上传对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164364C10805D385E1E3C67
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-id-2: 32AAAWJAMAABAAAQAAEAABAAAQAAEAABCTzu4Jp2lquWuXsjnLyPPiT3cfGhqPoY
Date: WED, 01 Jul 2015 04:11:15 GMT
Content-Length: 0
```

请求示例：上传对象的同时设置 ACL

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:13:55 GMT
x-obs-grant-
read:id=52f24s3593as5730ea4f722483579ai7,id=a93fcas852f24s3596ea8366794f7224
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:gYqplLq30dEX7GMi2qFWyjdFsyw=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

响应示例：上传对象的同时设置 ACL

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB7800000164845759E4F3B39ABEE55E
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSReVRNuas0knI+Y96iXrZA7BLUgj06Z
Date: WED, 01 Jul 2015 04:13:55 GMT
Content-Length: 0
```

请求示例：桶开启多版本时上传对象

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:17:12 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

响应示例：桶开启多版本时上传对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001439A51DB2B2577
ETag: "d41d8cd98f00b204e9800998ecf8427e"
X-OBS-ID-2: GcVgfeOJHx8JZHTHrRqkPsbKdB583fYbr3RBbHT6mMrBstReVILBZbMAdLiBYy1l
Date: WED, 01 Jul 2015 04:17:12 GMT
x-obs-version-id: AAABQ4q2M9_c0vycq3gAAAAAVURTRkha
Content-Length: 0
```

请求示例：上传对象时携带 MD5

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:17:50 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10
Content-MD5: 6Afx/PgtEy+bsBjKZzihnw==
Expect: 100-continue

1234567890
```

响应示例：上传对象时携带 MD5

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB7800000164B165971F91D82217D105
X-OBS-ID-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCSEKhBpS4BB3dSMNqMtuNxQDD9XvOw5h
ETag: "1072e1b96b47d7ec859710068aa70d57"
Date: WED, 01 Jul 2015 04:17:50 GMT
Content-Length: 0
```

请求示例：上传时配置 website 实现下载对象重定向

当桶设置了 Website 配置，您可以在上传对象时进行以下设置，设置后用户在下载对象时会重定向

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:17:12 GMT
x-obs-website-redirect-location: http://www.example.com/
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

响应示例：上传时配置 website 实现下载对象重定向

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001439A51DB2B2577
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTmxB5ufMj/7/GzP8TFwTbp33u0xhn2Z
ETag: "1072e1b96b47d7ec859710068aa70d57"
Date: WED, 01 Jul 2015 04:17:12 GMT
x-obs-version-id: AAABQ4q2M9_c0vycq3gAAAAAVURTRkha
Content-Length: 0
```

请求示例：在 URL 中携带签名并上传对象

```
PUT
/object02?AccessKeyId=H4IPJX0TQTHTHEBQQCEC&Expires=1532688887&Signature=EQmDuOhaLUr
zrzRNZxwS72CXeXM%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Content-Length: 1024

[1024 Byte data content]
```

响应示例：在 URL 中携带签名并上传对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001439A51DB2B2577
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTmxB5ufMj/7/GzP8TFwTbp33u0xhn2Z
ETag: "1072e1b96b47d7ec859710068aa70d57"
Date: Fri, 27 Jul 2018 10:52:31 GMT
x-obs-version-id: AAABQ4q2M9_c0vycq3gAAAAAVURTRkha
Content-Length: 0
```

请求示例：上传指定存储类型的对象

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:15:07 GMT
x-obs-storage-class: WARM
```

```
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

响应示例：上传指定存储类型的对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB7800000164846A2112F98BF970AA7E
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-id-2: a39E0UgAIAABAAAQAAEAABAAAQAAEAABCTPOUJu5XlNyU32fvKjM/92MQZK2gtoB
Date: WED, 01 Jul 2015 04:15:07 GMT
x-obs-storage-class: WARM
Content-Length: 0
```

请求示例：上传时配置对象级 WORM 保护策略

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:11:15 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:gYqplLq30dEX7GMi2qFWyjdFsyw=
Content-Length: 10240
x-obs-object-lock-mode:COMPLIANCE
x-obs-object-lock-retain-until-date:2022-09-24T16:10:25.297Z
Expect: 100-continue

[1024 Byte data content]
```

响应示例 ：上传时配置对象级 WORM 保护策略

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164364C10805D385E1E3C67
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-id-2: 32AAAWJAMAABAAAQAAEAABAAAQAAEAABCTzu4Jp2lquWuXsjnLyPPiT3cfGhqPoY
Date: WED, 01 Jul 2015 04:11:15 GMT
Content-Length: 0
```

5.4.2 POST 上传

功能介绍

上传对象操作是指在指定的桶内增加一个对象，执行该操作需要用户拥有桶的写权限。

说明

同一个桶中存储的对象名是唯一的。

在桶未开启多版本的情况下，如果在指定的桶内已经有相同的对象键值的对象，用户上传的新对象会覆盖原来的对象；为了确保数据在传输过程中没有遭到破坏，用户可

以在表单域中加入 Content-MD5 参数。在这种情况下，OBS 收到上传的对象后，会对对象进行 MD5 校验，如果不一致则返回出错信息。用户还可以在上传对象时指定 x-obs-acl 参数，设置对象的权限控制策略。

用户除了可以用 PUT 直接上传对象外，还可以使用 POST 上传对象。

单次上传对象大小范围是[0, 5GB]，如果需要上传超过 5GB 的大文件，需要通过 5.5 多段操作来分段上传。

该操作支持服务端加密功能。

与 PUT 上传的区别

PUT 上传中参数通过请求头域传递；POST 上传则作为消息体中的表单域传递。

PUT 上传需在 URL 中指定对象名；POST 上传提交的 URL 为桶域名，无需指定对象名。两者的请求行分别为：

```
PUT /ObjectName HTTP/1.1
POST / HTTP/1.1
```

关于 PUT 上传的更多详细信息，请参考 5.4.1 PUT 上传。

多版本

如果桶的多版本状态是开启的，系统会自动为对象生成一个唯一的版本号；如果桶的多版本状态是暂停的，则系统生成的对象版本号为 **null**，并由响应报头 x-obs-version-id 返回该版本号。关于桶的多版本状态，参见 5.2.11 设置桶的多版本状态。

WORM

如果桶的 WORM 开关是开启的，则可以为对象配置 WORM。您可以通过携带元素 x-obs-object-lock-mode 和 x-obs-object-lock-retain-until-date 在上传对象的同时指定对象的保护策略，如果您不携带这些元素，但配置了桶级默认 WORM 策略，则新上传的对象会自动应用默认策略。您还可以在上传后配置或修改对象级 WORM 保护策略。

说明

在桶的 WORM 开关开启时，系统会自动打开多版本功能。WORM 保护是基于对象版本号的，配置 WORM 的版本受到 WORM 保护，没有配置 WORM 的版本可正常删除。例如，test.txt 001 受到 WORM 保护。此时再次上传同名文件，产生新的对象版本 test.txt 002，test.txt 002 并未配置 WORM，那么 test.txt 002 就不受保护可以正常删除。当您下载对象时，不指定版本号下载的是最新对象，也就是 test.txt 002。

请求消息样式

```
POST / HTTP/1.1
Host: bucketname.obs.region.example.com
User-Agent: browser_data
Accept: file_types
Accept-Language: Regions
Accept-Encoding: encoding
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"

acl
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

success_redirect
--9431149156168
Content-Disposition: form-data; name="content-Type"

content_type
--9431149156168
Content-Disposition: form-data; name="x-obs-meta-uuid"

uuid
--9431149156168
Content-Disposition: form-data; name="x-obs-meta-tag"

metadata
--9431149156168
Content-Disposition: form-data; name="AccessKeyId"

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

encoded_policy
--9431149156168
Content-Disposition: form-data; name="signature"

signature=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename"
Content-Type: image/jpeg

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

Upload to OBS
--9431149156168--
```

请求消息参数

该请求消息中不使用参数。

请求消息头

该请求使用公共的消息头，具体请参见表 3-3。

如果想要获取 CORS 配置信息，则需要使用的消息头如下表 5-103 所示。

表 5-103 获取 CORS 配置的请求消息头

消息头名称	消息头类型	是否必选	描述
Origin	String	是	**参数解释：** 预请求指定的跨域请求 Origin（通常为域名）。 **约束限制：** 无 **取值范围：** 符合 http 协议的该头域的取值。 **默认取值：** 无
Access-Control-Request-Headers	String	否	**参数解释：** 实际请求可以带的 HTTP 头域，可以带多个头域。 **约束限制：** 无 **取值范围：** 符合 http 协议的该头域的取值。 **默认取值：** 无

请求消息元素

该请求消息的消息元素以表单形式组织，表单字段的具体含义如表 5-104 所示。

表 5-104 请求消息表单元素

元素名称	元素类型	是否必选	描述
file	二进制或文本类型	是	**参数解释：** 上传的对象内容。文件名与文件路径均会被忽略，不会作为对象名称。对象名称是另一参数 key 的值。 **约束限制：** • 此参数必须为最后一个参数，否则此参数之后的参数会被丢弃。 • 一个请求中只能含有一个 file 参数。 **取值范围：** 无

元素名称	元素类型	是否必选	描述
			默认取值： 无
key	String	是	**参数解释：** 通过此请求创建的对象名。对象名是对象在存储桶中的唯一标识。对象名是对象在桶中的完整路径，路径中不包含桶名。 **约束限制：** 无。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 无
AccessKeyId	String	是，有条件	**参数解释：** 用来指明请求发起者的访问密钥。 **约束限制：** 如果该请求包括安全策略参数 policy 或 signature 时，则必须包括此参数。 **取值范围：** 用户的 AK。 **默认取值：** 无
policy	String	是，有条件	**参数解释：** 该请求的安全策略描述。 **约束限制：** 当 Bucket 提供了 AccessKeyId（或 signature）表单域时，则必须包括此参数。 **取值范围：** 参见 3.2.4 基于浏览器上传的表单中携带签名章节中 policy 格式。 **默认取值：** 无
signature	String	是，有条件	**参数解释：** 根据 StringToSign 计算出的签名字符串。 **约束限制：** 当 Bucket 提供了 AccessKeyId(或 policy)表单域时，则必须包括此参数。 **取值范围：**

元素名称	元素类型	是否必选	描述
			无 **默认取值：** 无
token	String	否	**参数解释：** 用来统一指明请求发起者的访问密钥、请求签名和请求的安全策略。 **约束限制：** token 的优先级高于单独指定的访问密钥、请求签名和请求的安全策略。 示例： HTML 中： `<input type="text" name="token" value="ak:signature:policy" />` **取值范围：** 满足 ak:signature:policy 格式的值。 **默认取值：** 无
x-obs-acl	String	否	**参数解释：** 创建对象时，可以加上此消息头设置对象的权限控制策略，使用的策略为预定义的常用策略。 示例： POLICY 中：{"acl": "public-read" }, HTML 中： `<input type="text" name="acl" value="public-read" />` **约束限制：** 无 **取值范围：** • private • public-read • public-read-write • public-read-delivered • public-read-write-delivered **默认取值：** private
x-obs-grant-read	String	否	**参数解释：** 创建对象时，使用此头域授权 domain 下所有

元素名称	元素类型	是否必选	描述
			用户有读对象和获取对象元数据的权限。 示例： POLICY 中：{"grant-read": "id=domainId1" }, HTML 中： `<input type="text" name="grant-read" value="id=domainId1" />` **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** domainId 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-read-acp	String	否	**参数解释：** 创建对象时，使用此头域授权 domain 下所有用户有获取对象 ACL 的权限。 示例： POLICY 中：{"grant-read-acp": "id=domainId1" }, HTML 中： `<input type="text" name="grant-read-acp" value="id=domainId1" />` **约束限制：** 无 **取值范围：** domainId 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-write-acp	String	否	**参数解释：** 创建对象时，使用此头域授权 domain 下所有用户有写对象 ACL 的权限。 示例： POLICY 中：{"grant-write-acp": "id=domainId1" }, HTML 中： `<input type="text" name="grant-write-acp"`

元素名称	元素类型	是否必选	描述
			`value="id=domainId1" />` **约束限制：** 无 **取值范围：** domainId 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-grant-full-control	String	否	**参数解释：** 创建对象时，使用此头域授权 domain 下所有用户有读对象、获取对象元数据、获取对象 ACL、写对象 ACL 的权限。 示例： POLICY 中：{"grant-full-control": "id=domainId1" }, HTML 中： `<input type="text" name="grant-full-control" value="id=domainId1" />` **约束限制：** 无 **取值范围：** domainId 需要为正确的租户 ID，获取方式请参考 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 无
x-obs-storage-class	String	否	**参数解释：** 创建对象时，可以加上此头域设置对象的存储类型。 示例： POLICY 中：{"storage-class": "STANDARD" }, HTML 中： `<input type="text" name="x-obs-storage-class" value="STANDARD" />` **约束限制：** 设置该参数的值时请注意大小写敏感。 **取值范围：** • STANDARD

元素名称	元素类型	是否必选	描述
			• WARM • COLD **默认取值：** 如果未设置此头域，则以桶的默认存储类型作为对象的存储类型。
Cache-Control	String	否	**参数解释：** 该参数为 HTTP 标准消息头，OBS 将该参数记录下来，当用户下载此对象或 Head Object 时，在响应消息头中携带该参数。 **约束限制：** 无 **取值范围：** 参见 http 协议的 Cache-Control 头域。 **默认取值：** 无
Content-Type	String	否	**参数解释：** 该参数为 HTTP 标准消息头，OBS 将该参数记录下来，当用户下载此对象或 Head Object 时，在响应消息头中携带该参数。 示例： POLICY 中：["starts-with", "$Content-Type", "text/"], HTML 中： `<input type="text" name="content-type" value="text/plain" />` **约束限制：** 无 **取值范围：** Content-type 的常见取值参见《对象存储服务用户指南》中的“配置和查看对象元数据”章节内容。 **默认取值：** 无
Content-Disposition	String	否	**参数解释：** 该参数为 HTTP 标准消息头，OBS 将该参数记录下来，当用户下载此对象或 Head Object 时，在响应消息头中携带该参数。 **约束限制：**

元素名称	元素类型	是否必选	描述
			无 **取值范围：** 参见 http 协议的 Content-Disposition 头域。 **默认取值：** 无
Content-Encoding	String	否	**参数解释：** 该参数为 HTTP 标准消息头，OBS 将该参数记录下来，当用户下载此对象或 Head Object 时，在响应消息头中携带该参数。 **约束限制：** 无 **取值范围：** 参见 http 协议的 Content-Encoding 头域。 **默认取值：** 无
Expires	String	否	**参数解释：** 该参数为 HTTP 标准消息头，OBS 将该参数记录下来，当用户下载此对象或 Head Object 时，在响应消息头中携带该参数。 **约束限制：** 无 **取值范围：** 参见 http 协议的 Expires 头域。 **默认取值：** 无
success_action_redirect	String	否	**参数解释：** 用于指定当此次请求操作成功响应后的重定向的地址。 • 如果此参数值有效且操作成功，响应码为 303，Location 头域由此参数以及桶名、对象名、对象的 ETag 组成。 • 如果此参数值无效，则 OBS 忽略此参数的作用，Location 头域为对象地址，响应码根据操作成功或失败正常返回。 示例： POLICY 中：{"success_action_redirect": "http://123458.com"}, HTML 中：

元素名称	元素类型	是否必选	描述
			`<input type="text" name="success_action_redirect" value="http://123458.com" />` **约束限制：** 需要 http 和 https 开头。 **取值范围：** URL **默认取值：** 无
x-obs-meta-*	String	否	**参数解释：** 创建对象时，可以在 HTTP 请求中加入“x-obs-meta-”消息头或以“x-obs-meta-”开头的消息头，用来加入自定义的元数据，以便对对象进行自定义管理。当用户获取此对象或查询此对象元数据时，加入的自定义元数据将会在返回消息的消息头中出现。 示例： POLICY 中：{" x-obs-meta-test ": " test metadata " }, HTML 中： `<input type="text" name=" x-obs-meta-test " value=" test metadata " />` **约束限制：** 无 **取值范围：** 详见《对象存储服务用户指南》的“对象元数据”章节。 **默认取值：** 无
success_action_status	String	否	**参数解释：** 这个参数指定成功响应的状态码。 示例： POLICY 中：["starts-with", "$success_action_status", ""], HTML 中： `<input type="text" name="success_action_status" value="200" />` **约束限制：** • 如果此参数值被设定为 200 或 204，OBS 响应消息中 body 为空。

元素名称	元素类型	是否必选	描述
			• 如果此参数值被设定为 201，则 OBS 响应消息中包含一个 XML 文档描述此次请求的响应。 • 当请求不携带此参数或参数无效时，OBS 响应码为 204。 **取值范围：** • 200：请求成功。 • 201：请求成功并创建了新的资源。 • 204：请求成功，但未返回内容。 **默认取值：** 无
x-obs-website-redirect-location	String	否	***参数解释：*** *当桶设置了Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的URL，OBS 将这个值从头域中取出，保存在对象的元数据中。* ***约束限制：*** *必须以“/”、“http://”或“https://”开头，长度不超过 2K。* ***取值范围：*** *无* ***默认取值：*** *无*
x-obs-server-side-encryption	String	否。当使用 SSE-KMS 方式时，必选。	**参数解释：** 配置服务端加密方式。示例：x-obs-server-side-encryption：kms **约束限制：** 无 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	否	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：**

元素名称	元素类型	是否必选	描述
			当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密使用的密钥。该密钥用于加密对象。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-	String	否。当使用 SSE-C 方式时，	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。MD5

元素名称	元素类型	是否必选	描述
customer-key-MD5		必选。	值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥的 MD5 值。 **默认取值：** 无
x-obs-expires	Integer	否	**参数解释：** 表示对象的过期时间，单位是天。过期之后对象会被自动删除。（从对象最后修改时间开始计算） 此字段对于每个对象支持上传时配置，也支持后期通过修改元数据接口修改。 示例：x-obs-expires:3 **约束限制：** 设置的天数计算出的过期时间不能早于当前时间，如 10 天前上传的对象，不能设置小于 10 的值。 **取值范围：** 大于 0 的整数值。 **默认取值：** 无
x-obs-object-lock-mode	String	否，携带 x-obs-object-lock-retain-until-date 时必带。	**参数解释：** 要应用于此对象的 WORM 模式。 示例：x-obs-object-lock-mode:COMPLIANCE **约束限制：** 目前仅支持 COMPLIANCE，即合规模式。 该参数需要和 x-obs-object-lock-retain-until-date 一同使用。 **取值范围：** COMPLIANCE

元素名称	元素类型	是否必选	描述
			默认取值： 无
x-obs-object-lock-retain-until-date	String	否，携带 x-obs-object-lock-mode 时必带。	**参数解释：** 此对象的 WORM 策略过期的截止时间。 示例：x-obs-object-lock-retain-until-date:2015-07-01T04:11:15.297Z **约束限制：** 格式要求为 UTC 时间，并符合 ISO 8601 标准。如：2015-07-01T04:11:15.297Z 该参数需要和 x-obs-object-lock-mode 一同使用。 **取值范围：** 需要大于当前时间。 **默认取值：** 无

响应消息样式

```
HTTP/1.1 status_code
Content-Type: application/xml
Location: location
Date: date
ETag: etag
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如表 5-105 中的消息头。

表 5-105 附加响应消息头

消息头名称	消息头类型	描述
x-obs-version-id	String	**参数解释：** 对象的版本号。 **约束限制：** • 如果桶的多版本状态为开启，则会返回对象的版本号。 • 如果桶的多版本状态为暂停，则会返回 null。 **取值范围：** 服务端自动生成。

消息头名称	消息头类型	描述
		默认取值： 无
Access-Control-Allow-Origin	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 Origin 满足服务端的 CORS 配置，则在响应中包含这个 Origin。 **约束限制：** 无 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-Allow-Headers	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 headers 满足服务端的 CORS 配置，则在响应中包含这个 headers。 **约束限制：** 无 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-Max-Age	Integer	**参数解释：** 当桶设置了 CORS 配置，服务端 CORS 配置中的 MaxAgeSeconds。 **约束限制：** 无 **取值范围：** 大于等于 0 的整型数，单位：秒。 **默认取值：** 3000
Access-Control-Allow-Methods	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 Access-Control-Request-Method 满足服务端的 CORS 配置，则在响应中包含这条 rule 中的 Methods。 **约束限制：**

消息头名称	消息头类型	描述
		无 **取值范围：** • GET • PUT • HEAD • POST • DELETE **默认取值：** 无
Access-Control-Expose-Headers	String	**参数解释：** 桶 CORS 规则中的 ExposeHeader。ExposeHeader 是指 CORS 规则允许响应中可返回的附加头域，给客户端提供额外的信息。默认情况下浏览器只能访问以下头域：Content-Length、Content-Type，如果需要访问其他头域，需要在附加头域中配置。 **约束限制：** 不支持*、&、:、<、空格以及中文字符。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 服务端加密方式。示例：x-obs-server-side-encryption：kms **约束限制：** 无 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域

消息头名称	消息头类型	描述
		且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的 KMS 主密钥所属的项目 ID（非企业项目 ID）。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域 **取值范围：** 密钥的 MD5 值。 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：普通 *POST* 上传

```
POST / HTTP/1.1
Date: WED, 01 Jul 2015 04:15:23 GMT
Host: examplebucket.obs.region.example.com
Content-Type: multipart/form-data; boundary=7db143f50da2
Content-Length: 2424
Origin: www.example.com
Access-Control-Request-Headers:acc_header_1

--7db143f50da2
Content-Disposition: form-data; name="key"

object01
--7db143f50da2
Content-Disposition: form-data; name="acl"

public-read
--7db143f50da2
Content-Disposition: form-data; name="content-type"

text/plain
--7db143f50da2
Content-Disposition: form-data; name="expires"

WED, 01 Jul 2015 04:16:15 GMT
--7db143f50da2
Content-Disposition: form-data; name="AccessKeyId"

14RZT432N80TGDF2Y2G2
--7db143f50da2
Content-Disposition: form-data; name="policy"

ew0KICAiZXhaaXJhdGlvbiI6ICIyMDE1LTA3LTAxVDEyOjAwOjAwLjAwMFoiLA0KICAiY29uZGl0aW9ucyI
6IFsNCiAgICB7ImJ1Y2tldCI6ICJleG1hcGxlYnVja2V0IiB9LA0KICAgIHsiYWNsIjogInB1YmxpYy1yZW
FkIiB9LA0KICAgIHsiRXhaaXJlcyI6ICIxMDAwIiB9LA0KICAgIFsiZXEiLCAiJGtleSIsICJvYmplY3QwM
SJdLA0KICAgIFsic3RhcnRzLXdpdGgiLCAiJENvbnRlbnQtVHlwZSIsICJ0ZXh0LyJdLA0KICBdDQp9DQo=
--7db143f50da2
Content-Disposition: form-data; name="signature"

Vk6rwO0Nq09BLhvNSIYwSJTRQ+k=
--7db143f50da2
Content-Disposition: form-data; name="x-obs-persistent-headers"

test:dmFsdWUx
--7db143f50da2
```

```
Content-Disposition: form-data; name="x-obs-grant-read"

id=52f24s3593as5730ea4f722483579xxx
--7db143f50da2
Content-Disposition: form-data; name="x-obs-server-side-encryption"

kms
--7db143f50da2
Content-Disposition: form-data; name="x-obs-website-redirect-location"

http://www.example.com/
--7db143f50da2
Content-Disposition: form-data; name="file";
filename="C:\Testtools\UpLoadFiles\object\1024Bytes.txt"
Content-Type: text/plain

01234567890
--7db143f50da2
Content-Disposition: form-data; name="submit"

Upload
--7db143f50da2--
```

响应示例：普通 POST 上传

桶配置 cors 后，响应会包含 Access-Control-*的信息。

```
HTTP/1.1 204 No Content
x-obs-request-id: 90E2BA00C26C00000133B442A90063FD
x-obs-id-2: OTBFMkJBMDBDMjZDMDAwMDAxMzNCNDQyQTkwMDYzRkRBQUFBQUFBQWJiYmJiYmJi
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT
Access-Control-Allow-Headers: acc_header_01
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: exp_header_01
Content-Type: text/xml
Location: http://examplebucket.obs.region.example.com/object01
Date: WED, 01 Jul 2015 04:15:23 GMT
ETag: "ab7abb0da4bca5323ab6119bb5dcd296"
```

请求示例：带 x-obs-acl 头域并指定存储类型

带 x-obs-acl 头域并指定存储类型，重定向头域，上传对象

编码前，policy 的内容为

```
{
   "expiration":"2018-07-17T04:54:35Z",
   "conditions":[
      {
         "content-type":"text/plain"
      },
      {
         "x-obs-storage-class":"WARM"
```

```
      },
      {
         "success_action_redirect":"http://www.example.com"
      },
      {
         "x-obs-acl":"public-read"
      },
      [
         "starts-with",
         "$bucket",
         ""
      ],
      [
         "starts-with",
         "$key",
         ""
      ]
   ]
}
```

请求示例：

```
POST / HTTP/1.1
Host: examplebucket.obs.region.example.com
Accept-Encoding: identity
Content-Length: 947
Content-Type: multipart/form-data; boundary=9431149156168
User-Agent: OBS/Test

--9431149156168
Content-Disposition: form-data; name="x-obs-acl"

public-read
--9431149156168
Content-Disposition: form-data; name="AccessKeyId"

H4IPJX0TQTHTHEBQQCEC
--9431149156168
Content-Disposition: form-data; name="key"

my-obs-object-key-demo
--9431149156168
Content-Disposition: form-data; name="signature"

WNwv8P1ZiWdqPQqjXeLmAfzPDAI=
--9431149156168
Content-Disposition: form-data; name="policy"

eyJleHBpcmF0aW9uIjoiMjAxOC0wNy0xN1QwODozNDoyM1oiLCAiY29uZGl0aW9ucyI6W3siY29udGVudC1
0eXBlIjoidGV4dC9wbGFpbiJ9LHsieC1vYnMtYWNsIjoicHVibGljLXJlYWQifSxbInN0YXJ0cy13aXRoIi
wgIiRidWNrZXQiLCAiIl0sWyJzdGFydHMtd2l0aCIsICIka2V5IiwgIiJdXX0=
--9431149156168
Content-Disposition: form-data; name="content-type"

text/plain
--9431149156168
```

```
Content-Disposition: form-data; name="file"; filename="myfile"
Content-Type: text/plain

c2c6cd0f-898e-11e8-aab6-e567c91fb541
52b8e8a0-8481-4696-96f3-910635215a78

--9431149156168--
```

响应示例：带 x-obs-acl 头域并指定存储类型

```
HTTP/1.1 204 No Content
Server: OBS
Location: http://examplebucket.obs.region.example.com/my-obs-object-key-demo
ETag: "17a83fc8d431273405bd266114b7e034"
x-obs-request-id: 5DEB00000164A728A7C7F4E032214CFA
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCSwj2PcBE0YcoLHUDO7GSj+rVByzjflA
Date: Tue, 17 Jul 2018 07:33:36 GMT
```

请求示例：使用 token 进行鉴权

```
POST / HTTP/1.1
Content-Type:multipart/form-data; boundary=9431149156168
Content-Length: 634
Host: examplebucket.obs.region.example.com

--9431149156168
Content-Disposition: form-data; name="key"
obj01

--9431149156168
Content-Disposition: form-data; name="token"
UDSIAMSTUBTEST002538:XsVcTzR2/A284oE4VH9qPndGcuE=:eyJjb25kaXRpb25zIjogW3siYnVja2V0I
jogInRlc3QzMDAzMDU4NzE2NjI2ODkzNjcuMTIifSwgeyJDb250ZW50LVR5cGUiOiAiYXBwbGljYXRpb24v
eG1sIn0sIFsiZXEiLCAiJGtleSIsICJvYmoudRh0Il1dLCAiZXhwaXGhdGlvbiI6ICIyMDIyLTA5LTA5VDE
yOjA5OjI3WiJ9

--9431149156168
Content-Disposition: form-data; name="file"; filename="myfile"
Content-Type: text/plain
01234567890

--9431149156168--
Content-Disposition: form-data; name="submit"
Upload to OBS
```

响应示例：使用 token 进行鉴权

```
HTTP/1.1 204 No Content
Server: OBS
Location: http://examplebucket.obs.region.example.com/my-obs-object-key-demo
ETag: "7eda50a430fed940023acb9c4c6a2fff"
x-obs-request-id: 000001832010443D80F30B649B969C47
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCTj0yO9KJd5In+i9pzTgCDVG9vMnk7O/
Date: Fri,09Sep 2022 02: 24:40 GMT
```

请求示例：设置对象过期时间

```
POST / HTTP/1.1
Date: WED, 01 Jul 2015 04:15:23 GMT
Host: examplebucket.obs.region.example.com
Content-Type: multipart/form-data; boundary=148828969260233905620870
Content-Length: 1639
Origin: www.example.com
Access-Control-Request-Headers:acc_header_1

--148828969260233905620870
Content-Disposition: form-data; name="key"

object01
--148828969260233905620870
Content-Disposition: form-data; name="ObsAccessKeyId"

55445349414d535455425445535430303033033
--148828969260233905620870
Content-Disposition: form-data; name="signature"

396246666f6f42793872792f7a3958524f6c44334e4e69763950553d--7db143f50da2
--148828969260233905620870
Content-Disposition: form-data; name="policy"

65794a6c65484270636d463061573975496a6f694d6a41794d7930774e6930784e565178...
--148828969260233905620870
Content-Disposition: form-data; name="x-obs-expires"

4
--148828969260233905620870
Content-Disposition: form-data; name="file"; filename="test.txt"
Content-Type: text/plain

01234567890
--148828969260233905620870
Content-Disposition: form-data; name="submit"

Upload
--148828969260233905620870--
```

响应示例：设置对象过期时间

```
HTTP/1.1 204 No Content
Server: OBS
Date: Thu, 15 Jun 2023 12:39:03 GMT
Connection: keep-alive
Location: http://examplebucket.obs.region.example.com/my-obs-object-key-demo
x-obs-expiration: expiry-date="Tue, 20 Jun 2023 00:00:00 GMT"
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-request-id: 00000188BF11049553064911000FC30D
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCSwj2PcBE0YcoLHUDO7GSj+rVByzjflA
x-forward-status: 0x40020000000001
x-dae-api-type: REST.POST.OBJECT
```

请求示例：指定状态码

指定成功响应的状态码为 200

```
POST /srcbucket HTTP/1.1
User-Agent: PostmanRuntime/7.26.8
Accept: */*
Postman-Token: 667dcc44-1c48-41ba-9e41-9f87d8975089
Host: obs.region.example.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Type: multipart/form-data; boundary=--------------------------
285613759795901770404350
Content-Length: 1134

----------------------------285613759795901770404350
Content-Disposition: form-data; name="key"

obj
----------------------------285613759795901770404350
Content-Disposition: form-data; name="ObsAccessKeyId"

XXXXXXXXXXXXXXX000003
----------------------------285613759795901770404350
Content-Disposition: form-data; name="signature"

9rc4bVhDPQ7eHtw17hWtYxLnBWU=
----------------------------285613759795901770404350
Content-Disposition: form-data; name="policy"

eyJleHBpcmF0aW9uIjoiMjAyMy0wNi0xNVQxNDoxMTozNFoiLCAiY29uZGl0aW9ucyI6W3siYnVja2V0Ijo
ic3JjYnVja2V0MiJ9LHsic3VjY2Vzc19hY3Rpb25fc3RhdHVzIjoiMjAwIn0seyJjb250ZW50LXR5cGUiOi
J0ZXh0L3BsYWluIn0seyJrZXkiOiIzMzMifSxdfQ==
----------------------------285613759795901770404350
Content-Disposition: form-data; name="success_action_status"

200
----------------------------285613759795901770404350
Content-Disposition: form-data; name="file"; filename="test.txt"
Content-Type: text/plain


----------------------------285613759795901770404350
Content-Disposition: form-data; name="submit"

Upload to OBS
----------------------------285613759795901770404350--
```

响应示例：指定状态码

指定成功响应的状态码为 200，响应消息

```
HTTP/1.1 200 OK
Server: OBS
Date: Thu, 15 Jun 2023 13:12:51 GMT
```

```
Content-Length: 0
Connection: keep-alive
Location: http://obs.regionexample.com/srcbucket/obj
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-request-id: 00000188BF2FF55F5306426E000FE366
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCScDjcXgZ7oMYSVnZnk4+HrClVwLVPTi
x-forward-status: 0x40020000000001
x-dae-api-type: REST.POST.OBJECT
```

请求示例：上传时配置对象级 WORM 保护策略

```
POST /srcbucket HTTP/1.1
User-Agent: PostmanRuntime/7.26.8
Accept: */*
Postman-Token: 4c2f4c7e-2e0b-46c0-b1a7-4a5da560b6a1
Host: obs.regionexample.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Type: multipart/form-data; boundary=--------------------------
940435396775653808840608
Content-Length: 1409

----------------------------940435396775653808840608
Content-Disposition: form-data; name="key"

obj
----------------------------940435396775653808840608
Content-Disposition: form-data; name="ObsAccessKeyId"

XXXXXXXXXXXXXXX000003
----------------------------940435396775653808840608
Content-Disposition: form-data; name="signature"

X/7QiyMYUvxUWk0R5fToeTcgMMU=
----------------------------940435396775653808840608
Content-Disposition: form-data; name="policy"

eyJleHBpcmF0aW9uIjoiMjAyMy0wNi0xNVQxNDoyMjo1MVoiLCAiY29uZGl0aW9ucyI6W3sieC1vYnMtb2J
qZWN0LWxvY2stcmV0YWluLXVudGlsLWRhdGUiOiJUaHUsIDIwIEp1biAyMDIzIDEzOjEyOjUxIEdNVCJ9LH
sieC1vYnMtb2JqZWN0LWxvY2stbW9kZSI6IkNPTVBMSUFOQ0UifSx7ImJ1Y2tldCI6InNyY2J1Y2tldDIif
Sx7ImNvbnRlbnQtdHlwZSI6InRleHQvcGxhaW4ifSx7ImtleSI6IjMzMyJ9LF19
----------------------------940435396775653808840608
Content-Disposition: form-data; name="x-obs-object-lock-mode"

COMPLIANCE
----------------------------940435396775653808840608
Content-Disposition: form-data; name="x-obs-object-lock-retain-until-date"

2022-09-24T16:10:25.297Z
----------------------------940435396775653808840608
Content-Disposition: form-data; name="file"; filename="test.txt"
Content-Type: text/plain


----------------------------940435396775653808840608
```

```
Content-Disposition: form-data; name="submit"

Upload to OBS
-----------------------------940435396775653808840608--
```

响应示例：上传时配置对象级 WORM 保护策略

```
HTTP/1.1 204 No Content
Server: OBS
Date: Thu, 15 Jun 2023 13:24:03 GMT
Connection: keep-alive
Location: http://obs.region.example.com/srcbucket/obj
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-request-id: 00000188BF3A36EE5306427D000FEE0A
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCS/5pj0p0hAQcDVI3B6E5y167zy4eAQv
x-forward-status: 0x40020000000001
x-dae-api-type: REST.POST.OBJECT
```

5.4.3 复制对象

功能介绍

复制对象（Copy Object）特性用来为 OBS 上已经存在的对象创建一个副本。

当进行复制对象操作时，目标对象默认复制源对象的元数据；用户也可以将目标对象的元数据替换为本次请求中所带的元数据。新建的目标对象不会复制源对象的 ACL 信息，默认的新建对象的 ACL 是 private，用户可以使用设置 ACL 的操作接口来重新设定新对象的 ACL。

复制对象操作的请求需要通过头域携带拷贝的源桶和对象信息，不能携带消息实体。

该操作支持服务端加密功能。

目标对象大小范围是[0, 5GB]，如果源对象大小超过 5GB，只能使用 5.5.4 拷贝段功能拷贝部分对象。

说明

复制对象的结果不能仅根据 HTTP 返回头域中的 status_code 来判断请求是否成功，头域中 status_code 返回 200 时表示服务端已经收到请求，且开始处理复制对象请求。复制是否成功会在响应消息的 body 中，只有 body 体中有 ETag 标签才表示成功，否则表示复制失败。

多版本

默认情况下，x-obs-copy-source 标识复制源对象的最新版本。如果源对象的最新版本是删除标记，则认为该对象已删除。要复制指定版本的对象，可以在 x-obs-copy-source 请求消息头中携带 versionId 参数。

如果目标对象的桶的多版本状态是开启的，系统为目标对象生成唯一的版本号（此版本号与源对象的版本号不同），并且会在响应报头 x-obs-version-id 返回该版本号。如果目标对象的桶的多版本状态是暂停的，则目标对象的版本号为 **null**。

须知

在桶没有开启多版本的情况下，将源对象 object_A 复制为目标对象 object_B，如果在复制操作之前对象 object_B 已经存在，复制操作执行之后老对象 object_B 则会被新复制对象 object_B 覆盖，复制成功后，只能下载到新的对象 object_B，老对象 object_B 将会被删除。因此在使用 copy 接口时请确保目标对象不存在或者已无价值，避免因 copy 导致数据误删除。复制过程中源对象 object_A 无任何变化。

WORM

如果桶的 WORM 开关是开启的，则可以为对象配置 WORM。您可以通过携带头域 x-obs-object-lock-mode 和 x-obs-object-lock-retain-until-date 在复制对象的同时指定对象的保护策略，如果您不携带这些头域，但配置了桶级默认 WORM 策略，则复制的对象会自动应用默认策略。您还可以在复制后配置或修改对象级 WORM 保护策略。

说明

在复制对象时，新建的目标对象与源对象的 WORM 保护状态相互独立，复制时不会复制源对象的 WORM 保护状态。在复制完成后，对源对象的 WORM 策略进行配置也不会影响新建的目标对象。

对象

如果源对象是归档存储对象，需要判断源对象的恢复状态，只有当源对象处于已恢复状态时，才能复制成功。源对象未恢复或者正在恢复时，会复制失败，返回错误 403 Forbidden。异常描述为：

ErrorCode: InvalidObjectState

ErrorMessage: Operation is not valid for the source object's storage class

请求消息样式

```
PUT /destinationObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
x-obs-copy-source: /sourceBucket/sourceObject
x-obs-metadata-directive: metadata_directive
x-obs-copy-source-if-match: etag
x-obs-copy-source-if-none-match: etag
x-obs-copy-source-if-unmodified-since: time_stamp
x-obs-copy-source-if-modified-since: time_stamp
Authorization: signature
Date: date
```

请求消息参数

该请求消息中不使用消息参数。

请求消息头

该消息可以带附加的消息头指定复制的信息，具体如表 3-3 所示。

表 5-106 请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-acl	String	否	**参数解释：** 复制对象时，可以加上此消息头设置对象的权限控制策略，使用的策略为预定义的常用策略。 示例：x-obs-acl: acl **约束限制：** 预定义策略为字符串形式。 **取值范围：** • private • public-read • public-read-write **默认取值：** private
x-obs-grant-read	String	否	**参数解释：** 复制对象时，使用此头域授权读对象和获取对象元数据的权限给 domain 下的所有用户。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-grant-read-acp	String	否	**参数解释：** 复制对象时，使用此头域授权获取对象 ACL 的权限给 domain 下的所有用户。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-grant-write-acp	String	否	**参数解释：** 复制对象时，使用此头域授权写对象 ACL 的权限给 domain 下的所有用户。 **约束限制：**

消息头名称	消息头类型	是否必选	描述
			无 **取值范围：** 无 **默认取值：** 无
x-obs-grant-full-control	String	否	**参数解释：** 复制对象时，使用此头域授权以下权限给domain 下的所有用户： 读对象、获取对象元数据、获取对象 ACL、写对象 ACL 的权限。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-copy-source	String	是	**参数解释：** 用来指定复制对象操作的源桶名以及源对象名。当源对象存在多个版本时，通过versionId 参数指定版本源对象。 示例：x-obs-copy-source: /source_bucket/sourceObject **约束限制：** 中文字符与%，需要进行 URL 编码。 **取值范围：** 无 **默认取值：** 无
x-obs-metadata-directive	String	否	**参数解释：** 此参数用来指定新对象的元数据是从源对象中复制，还是用请求中的元数据替换。 示例：x-obs-metadata-directive: metadata_directive **约束限制：** 如果此参数的值不是 COPY 或 REPLACE，则 OBS 立即返回 400 错误；如果用户进行修改元数据操作（源对象与目标对象相同），

消息头名称	消息头类型	是否必选	描述
			则此参数只能为 REPLACE，否则此请求作为无效请求，服务端响应 400。此参数不支持将加密的对象更改成非加密对象（源对象与目标对象相同）。如果用户使用此参数更改加密的对象，系统将返回 400。 **取值范围：** • COPY • REPLACE **默认取值：** COPY
x-obs-copy-source-if-match	String	否	**参数解释：** 只有当源对象的 Etag 与此参数指定的值相等时才进行复制对象操作，否则返回 412（前置条件不满足）。 示例：x-obs-copy-source-if-match: etag **约束限制：** 此参数可与 x-obs-copy-source-if-unmodified-since 一起使用，但不能与其它条件复制参数一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-copy-source-if-none-match	String	否	**参数解释：** 只有当源对象的 Etag 与此参数指定的值不相等时才进行复制对象操作，否则返回 412（前置条件不满足）。 示例：x-obs-copy-source-if-none-match: etag **约束限制：** 此参数可与 x-obs-copy-source-if-modified-since 一起使用，但不能与其它条件复制参数一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-copy-source-if-	String	否	**参数解释：** 只有当源对象在此参数指定的时间之后没有

消息头名称	消息头类型	是否必选	描述
unmodified-since			修改过才进行复制对象操作，否则返回 412（前置条件不满足）。 **约束限制：** • 此参数指定的时间不能晚于当前的服务器时间（GMT 时间），否则参数不生效。 • 此参数可与 x-obs-copy-source-if-match 一起使用，但不能与其它条件复制参数一起使用 • 格式：符合 http://www.ietf.org/rfc/rfc2616.txt 规定格式的 HTTP 时间字符串。 1. EEE, dd MMM yyyy HH:mm:ss z 2. EEEE, dd-MMM-yy HH:mm:ss z 3. EEE MMM dd HH:mm:ss yyyy 对应示例： 1. x-obs-copy-source-if-unmodified-since: Sun, 06 Nov 1994 08:49:37 GMT 2. x-obs-copy-source-if-unmodified-since: Sunday, 06-Nov-94 08:49:37 GMT 3. x-obs-copy-source-if-unmodified-since: Sun Nov 6 08:49:37 1994 **取值范围：** 无 **默认取值：** 无
x-obs-copy-source-if-modified-since	String	否	**参数解释：** 只有当源对象在此参数指定的时间之后修改过才进行复制对象操作，否则返回 412（前置条件不满足）。 **约束限制：** • 此参数指定的时间不能晚于当前的服务器时间（GMT 时间），否则参数不生效。 • 此参数可与 x-obs-copy-source-if-none-match 一起使用，但不能与其它条件复制参数一起使用 • 格式：符合 http://www.ietf.org/rfc/rfc2616.txt 规定格式的 HTTP 时间字符串。 1. EEE, dd MMM yyyy HH:mm:ss z 2. EEEE, dd-MMM-yy HH:mm:ss z

消息头名称	消息头类型	是否必选	描述
			3. EEE MMM dd HH:mm:ss yyyy 对应示例： 1. x-obs-copy-source-if-modified-since: Sun, 06 Nov 1994 08:49:37 GMT 2. x-obs-copy-source-if-modified-since: Sunday, 06-Nov-94 08:49:37 GMT 3. x-obs-copy-source-if-modified-since: Sun Nov 6 08:49:37 1994 **取值范围：** 无 **默认取值：** 无
x-obs-storage-class	String	否	**参数解释：** 复制对象时，可以加上此头域设置目的对象的存储类型。如果未设置此头域，则以目的对象所在桶的默认存储类型作为对象的存储类型。 示例：x-obs-storage-class: STANDARD **约束限制：** 设置该参数的值时请注意大小写敏感。 **取值范围：** • STANDARD • WARM • COLD **默认取值：** 默认继承桶的存储类型。
x-obs-website-redirect-location	String	否	**参数解释：** 当桶设置了 Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的 URL，OBS 将这个值从头域中取出，保存在对象的元数据中。 **约束限制：** 必须以“/”、“http://”或“https://”开头，长度不超过 2K。 **取值范围：** 无 **默认取值：** 无

消息头名称	消息头类型	是否必选	描述
x-obs-server-side-encryption	String	否。当使用 SSE-KMS 方式时，必选。	**参数解释：** 使用该头域表示服务端加密是 SSE-KMS 方式。目标对象使用 SSE-KMS 方式加密。 示例：x-obs-server-side-encryption: kms **约束限制：** 无 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	否	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密目标对象使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-	String	否。当使	**参数解释：**

消息头名称	消息头类型	是否必选	描述
encryption-customer-key		用 SSE-C 方式时，必选。	该头域表示加密目标对象使用的密钥。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密目标对象使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥的 MD5 值。 **默认取值：** 无
x-obs-copy-source-server-side-encryption-customer-algorithm	String	否。当拷贝源对象使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示解密源对象使用的算法。 示例：x-obs-copy-source-server-side-encryption-customer-algorithm: AES256 **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 需要和 x-obs-copy-source-server-side-

消息头名称	消息头类型	是否必选	描述
			encryption-customer-key，x-obs-copy-source-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-copy-source-server-side-encryption-customer-key	String	否。当拷贝源对象使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示解密源对象使用的密钥。 示例：x-obs-copy-source-server-side-encryption-customer-key: K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-copy-source-server-side-encryption-customer-algorithm，x-obs-copy-source-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-copy-source-server-side-encryption-customer-key-MD5	String	否。当拷贝源对象使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示解密源对象使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-copy-source-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-copy-source-server-side-encryption-customer-algorithm，x-obs-copy-source-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥的 MD5 值。

消息头名称	消息头类型	是否必选	描述
			默认取值： 无
success-action-redirect	String	否	**参数解释：** 用于指定当此次请求操作成功响应后的重定向的地址。 • 如果此参数值有效且操作成功，响应码为303，Location 头域由此参数以及桶名、对象名、对象的 ETag 组成。 • 如果此参数值无效，则 OBS 忽略此参数的作用，Location 头域为对象地址，响应码根据操作成功或失败正常返回。 **约束限制：** 需要为 URL 格式，如 http://domainname、https://domainname。 **取值范围：** URL **默认取值：** 无
x-obs-object-lock-mode	String	否，携带 x-obs-object-lock-retain-until-date 头域时必带	**参数解释：** 要应用于此对象的 WORM 模式。 示例：x-obs-object-lock-mode:COMPLIANCE **约束限制：** 该参数需要和 x-obs-object-lock-retain-until-date 一同使用。 **取值范围：** COMPLIANCE **默认取值：** 无
x-obs-object-lock-retain-until-date	String	否，携带 x-obs-object-lock-mode 头域时必带	**参数解释：** 此对象的 WORM 策略过期的截止时间。 示例：x-obs-object-lock-retain-until-date:2015-07-01T04:11:15.297Z **约束限制：** • 格式要求为 UTC 时间，并符合 ISO 8601 标准。如：2015-07-01T04:11:15.297Z • 该参数需要和 x-obs-object-lock-mode 一同使用。

消息头名称	消息头类型	是否必选	描述
			取值范围： 需要大于当前时间。 **默认取值：** 无

其他消息头请参见表 3-3 章节。

请求消息元素

该请求在消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CopyObjectResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <LastModified>modifiedDate</LastModified>
   <ETag>etagValue</ETag>
</CopyObjectResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如下表 5-107 中的消息头。

表 5-107 附加响应消息头

消息头名称	消息头类型	描述
x-obs-copy-source-version-id	String	**参数解释：** 源对象的版本号。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-version-id	String	**参数解释：** 目标对象的版本号。

消息头名称	消息头类型	描述
		约束限制： 无 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 该头域表示服务端的加密方式。 示例：x-obs-server-side-encryption: kms **约束限制：** 如果服务端加密是 SSE-KMS 方式，响应包含该头域。 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的 KMS 主密钥所属的项目 ID（非企业项目 ID）。

消息头名称	消息头类型	描述
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** AES256：使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无
x-obs-storage-class	String	**参数解释：** 对象的存储类型。 **约束限制：** 对象为非标准存储对象时，会返回此头域。 **取值范围：** • WARM：低频存储 • COLD：归档存储 **默认取值：** 返回对象的存储类型。

响应消息元素

该请求的响应消息通过消息元素来返回复制结果，元素具体含义如表 5-108 所示。

表 5-108 响应消息元素

元素名称	元素类型	描述
CopyObjectResult	XML	**参数解释：** 复制对象结果的容器。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
LastModified	String	**参数解释：** 对象最近一次被修改的时间（UTC 时间）。 **约束限制：** 日期格式为 ISO8601 的格式。 例如：2018-01-01T00:00:00.000Z，表示最后一次修改时间为 2018-01-01T00:00:00.000Z。 **取值范围：** 无 **默认取值：** 无
ETag	String	**参数解释：** 新对象的 base64 编码的 128 位 MD5 摘要。ETag 是对象内容的唯一标识，可以通过该值识别对象内容是否有变化。比如上传对象时 ETag 为 A，下载对象时 ETag 为 B，则说明对象内容发生了变化。 **约束限制：** 当对象是服务端加密的对象时，ETag 值不是对象的 MD5 值。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：普通复制

普通对象拷贝，将桶 bucket 中的对象 srcobject 拷贝到桶 examplebucket 中 destobject 对象

```
PUT /destobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:19:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:2rZR+iaH8xUewvUKuicLhLHpNoU=
x-obs-copy-source: /bucket/srcobject
```

响应示例：普通复制

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 001B21A61C6C00000134031BE8005293
x-obs-id-2: MDAxQjIxQTYxQzZDMDAwMDAxMzQwMzFCRTgwMDUyOTNBQUFBQUFBQWJiYmJiYmJi
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 249

<?xml version="1.0" encoding="utf-8"?>
<CopyObjectResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <LastModified>2015-07-01T00:48:07.706Z</LastModified>
  <ETag>"507e3fff69b69bf57d303e807448560b"</ETag>
</CopyObjectResult>
```

请求示例：复制多版本对象

拷贝一个多版本对象，将桶 bucket 中的版本号为 AAABQ4uBLdLc0vycq3gAAAAEVURTRkha 的对象 srcobject 拷贝到桶 examplebucket 中 destobject 对象

```
PUT /destobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:20:29 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:4BLYv+1UxfRSHBMvrhVLDszxvcY=
x-obs-copy-source: /bucket/srcobject?versionId=AAABQ4uBLdLc0vycq3gAAAAEVURTRkha
```

响应示例：复制多版本对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001438B8A9C898B79
x-obs-id-2: DB/qBZmbN6AIoX9mrrSNYdLxwvbO0tLR/l6/XKTT4NmZspzharwp5Z74ybAYVOgr
Content-Type: application/xml
x-obs-version-id: AAABQ4uKnOrc0vycq3gAAAAFVURTRkha
x-obs-copy-source-version-id: AAABQ4uBLdLc0vycq3gAAAAEVURTRkha
Date: WED, 01 Jul 2015 04:20:29 GMT
Transfer-Encoding: chunked

<?xml version="1.0" encoding="utf-8"?>
```

```
<CopyObjectResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <LastModified>2015-07-01T01:48:07.706Z</LastModified>
  <ETag>"507e3fff69b69bf57d303e807448560b"</ETag>
</CopyObjectResult>
```

5.4.4 下载对象

功能介绍

GET 操作从对象存储下载对象。使用 GET 接口前，请确认必须拥有对象的 READ 权限。如果对象 Owner 向匿名用户授予 READ 访问权限，则可以在不使用鉴权头域的情况下访问该对象。

服务端加密

如果客户端的对象上传时，使用了客户提供的加密密钥进行服务端加密，当下载对象时，同样也必须在消息中提供密钥。

多版本

默认情况下，获取的是最新版本的对象。如果最新版本的对象是删除标记，则返回对象不存在。如果要获取指定版本的对象，请求可携带 versionId 消息参数。

归档存储对象

如果要下载的对象是归档存储类对象，由于对象存储在存档设备中，您必须先使用对象恢复，然后才能下载该归档存储对象。对象处于不同的恢复状态时，给出不同响应：如果对象已恢复，下载对象成功时需要返回 x-obs-restore 头域指示恢复失效时间。对未恢复或正在恢复的归档存储对象发送下载请求时，会返回错误 403 Forbidden。

请求消息样式

```
GET /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Range:bytes=byte_range
<Optional Additional Header>
```

说明

其中 Range 字段可选，如果没有的话得到全部内容。

请求消息参数

GET 操作获取对象内容时，允许用户通过请求参数的方式对一些消息头值进行重写，可以重写的消息头有：Content-Type、Content-Language、Expires、Cache-Control、Content-Disposition 以及 Content-Encoding 共 6 个。另外所需恢复的对象拥有多个版本时，可以通过 versionId 参数，指定需要下载的版本。具体的说明如表 5-109 所示。

说明

OBS 不会处理请求中携带的 Accept-Encoding，也不会对上传的数据做任何压缩、解压的操作，压缩解压的操作由客户端决定。某些 HTTPClient 在默认情况下可能会根据服务端返回的 Content-Encoding 对数据做相应的解压处理，客户端程序需要根据自己的需求决定是否做解压处理以及如何解压（修改 OBS 端保存的对象元数据 Content-Encoding 或者在下载对象时对 Content-Encoding 进行重写）。如果在下载对象的请求中指明了重写消息头，OBS 返回的 HTTP 标准消息头中将以请求中指定的重写内容为准。

表 5-109 请求消息参数

参数名称	参数类型	是否必选	描述
response-content-type	String	否	**参数解释：** 下载对象时重写响应中的 Content-Type 头。 **约束限制：** 该参数不允许匿名用户使用，您必须在请求中携带签名，否则请求将报错 400 ForbidAnonReqOverwriteRespHead。 **取值范围：** 参见 HTTP 标准头域 Content-Type 的取值。 **默认取值：** 无
response-content-language	String	否	**参数解释：** 下载对象时重写响应中的 Content-Language 头。 **约束限制：** 该参数不允许匿名用户使用，您必须在请求中携带签名，否则请求将报错 400 ForbidAnonReqOverwriteRespHead。 **取值范围：** 参见 HTTP 标准头域 Content-Language 的取值。 **默认取值：** 无
response-expires	String	否	**参数解释：** 下载对象时重写响应中的 Expires 头。 **约束限制：**

参数名称	参数类型	是否必选	描述
			该参数不允许匿名用户使用，您必须在请求中携带签名，否则请求将报错 400 ForbidAnonReqOverwriteRespHead。 **取值范围：** 参见 HTTP 标准头域 Expires 的取值。 **默认取值：** 无
response-cache-control	String	否	**参数解释：** 下载对象时重写响应中的 Cache-Control 头。 **约束限制：** 该参数不允许匿名用户使用，您必须在请求中携带签名，否则请求将报错 400 ForbidAnonReqOverwriteRespHead。 **取值范围：** 参见 HTTP 标准头域 Cache-control 的取值。 **默认取值：** 无
response-content-disposition	String	否	**参数解释：** 下载对象时重写响应中的 Content-Disposition 头。 示例： response-content-disposition=attachment; filename*=utf-8"name1 下载对象重命名为“name1”，如果 name1 中存在中文，需要将中文进行 URL 编码。 **约束限制：** 该参数不允许匿名用户使用，您必须在请求中携带签名，否则请求将报错 400 ForbidAnonReqOverwriteRespHead。 **取值范围：**

参数名称	参数类型	是否必选	描述
			无 **默认取值：** 无
response-content-encoding	String	否	**参数解释：** 下载对象时重写响应中的 Content-Encoding 头。 **约束限制：** 该参数不允许匿名用户使用，您必须在请求中携带签名，否则请求将报错 400 ForbidAnonReqOverwriteRespHead。 **取值范围：** 参见 HTTP 标准头域 Content-Encoding 的取值。 **默认取值：** 无
versionId	String	否	**参数解释：** 对象的版本号，用于指定获取对象的版本号。 **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无，如果不设置则默认获取最新版本的对象。
x-image-process	String	否	**参数解释：** 图片处理服务，描述针对对象的图片处理命令或处理样式。 示例： 命令方式：x-image-process=**image/commands** 样式方式：x-image-process=**style/stylename** 详见《对象存储服务图片处理特性指南》。 **约束限制：**

参数名称	参数类型	是否必选	描述
			无 **取值范围：** 命令方式：image/命令参数。 样式方式：style/样式名称。 **默认取值：** 如果不输入处理命令，将返回原图。
attname	String	否	**参数解释：** 下载对象时重写响应中的 Content-Disposition 头。 示例： attname=name1 下载对象重命名为“name1”。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无

请求消息头

该请求除使用公共消息头外，还可以使用附加的消息头来完成获取对象的功能，消息头的意义如表 5-110 所示。

表 5-110 请求消息头

消息头名称	消息头类型	是否必选	描述
Range	String	否	**参数解释：** 获取对象时，获取在 Range 范围内的对象内容。 **约束限制：** • Range 有三种格式： **“bytes=起始值-结束值”**(表示的 Range 范围从起始值开始，到结束值终止。） **“bytes=起始值-”**（表示的 Range 范围从起始值开始，到“对象长度-1”终止。） **“bytes=-结束值”**（表示的 Range 范围从“对象长度-结束值”开始，到“对象长度-

消息头名称	消息头类型	是否必选	描述
			1”终止。） 如果未按照以上 3 种标准格式配置参数，则**忽略 Range 字段获取整个对象**。 示例 1：bytes=0-4（下载第 1~5 个字节的数据） 示例 2：bytes=4-（下载第 5 个字节到结尾的数据） 示例 3：bytes=-4（下载最后 4 个字节的数据） 示例 4：bytes=10-20,30-40（表示多个区间） • 携带 Range 头域后，响应消息的 ETag 仍是对象的 ETag，而不是 Range 范围内对象的 ETag。 • 下载对象单个区间内容时，Range 区间长度不能超过对象内容长度；下载对象多个区间内容时，所有 Range 区间的总长度不能超过对象总长度。 **取值范围：** • 起始值取值范围：[0, 对象长度-1]。 如果起始值小于 0，**系统将忽略 Range 字段，返回整个对象**。 如果起始值超过"对象长度-1"，**则请求报错，返回 416 状态码（InvalidRange）**。 • 结束值必须大于起始值 如果结束值不大于起始值，系统将忽略 Range 字段，返回整个对象。 结束值如果大于"对象长度-1"，系统会自动修正为"对象长度-1"。 **默认取值：** 无
If-Modified-Since	符合 http://www.ietf.org/rfc/rfc2616.txt 规定格式的 HTTP 时间字符串。	否	**参数解释：** 如果对象在请求中指定的时间之后有修改，则返回对象内容；否则的话返回 304（not modified）。 **约束限制：** 此参数指定的时间不能晚于当前的服务器时间（GMT 时间），否则参数不生效。 **取值范围：** 符合 http://www.ietf.org/rfc/rfc2616.txt 规定格式

消息头名称	消息头类型	是否必选	描述
			的 HTTP 时间字符串。 1. EEE, dd MMM yyyy HH:mm:ss z 2. EEEE, dd-MMM-yy HH:mm:ss z 3. EEE MMM dd HH:mm:ss yyyy 对应示例： 1. if-modified-since: Sun, 06 Nov 1994 08:49:37 GMT 2. if-modified-since: Sunday, 06-Nov-94 08:49:37 GMT 3. if-modified-since: Sun Nov 6 08:49:37 1994 **默认取值：** 无
If-Unmodified-Since	符合 http://www.ietf.org/rfc/rfc2616.txt 规定格式的 HTTP 时间字符串。	否	**参数解释：** 如果对象在请求中指定的时间之后没有修改，则返回对象内容；否则的话返回 412（precondition failed）。 **约束限制：** 此参数指定的时间不能晚于当前的服务器时间（GMT 时间），否则参数不生效。 **取值范围：** 符合 http://www.ietf.org/rfc/rfc2616.txt 规定格式的 HTTP 时间字符串。 1. EEE, dd MMM yyyy HH:mm:ss z 2. EEEE, dd-MMM-yy HH:mm:ss z 3. EEE MMM dd HH:mm:ss yyyy 对应示例： 1. if-unmodified-since: Sun, 06 Nov 1994 08:49:37 GMT 2. if-unmodified-since: Sunday, 06-Nov-94 08:49:37 GMT 3. if-unmodified-since: Sun Nov 6 08:49:37 1994 **默认取值：** 无
If-Match	String	否	**参数解释：** 如果对象的 ETag 和请求中指定的 ETag 相同，则返回对象内容，否则的话返回 412（precondition failed）。 （ETag 值，例：0f64741bf7cb1089e988e4585d0d3434）

消息头名称	消息头类型	是否必选	描述
			约束限制： 无 **取值范围：** 对象的 ETag **默认取值：** 无
If-None-Match	String	否	**参数解释：** 如果对象的 ETag 和请求中指定的 ETag 不相同，则返回对象内容，否则的话返回 304（not modified）。 （ETag 值，例：0f64741bf7cb1089e988e4585d0d3434） **约束限制：** 无 **取值范围：** 对象的 ETag **默认取值：** 无
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 需要和 x-obs-server-side-encryption-customer-key、 x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key	String	否。当使用 SSE-C 方式时，	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的密钥。该密钥用于解密对象。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw=

消息头名称	消息头类型	是否必选	描述
		必选。	**约束限制：** 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm、x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm、x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: type
Date: date
Content-Length: length
Etag:  etag
Last-Modified:  time

<Object Content>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如下表 5-111 中的消息头。

表 5-111 附加响应消息头

消息头名称	消息头类型	描述
x-obs-expiration	String	**参数解释：** 对象的过期时间。 **约束限制：** 当对象单独设置了对象 lifecycle，过期时间以对象 lifecycle 为准，该消息头用 expiry-date 描述对象的详细过期信息；如果对象没有设置对象 lifecycle，设置了桶级别 lifecycle，过期时间以桶级别 lifecycle 为准，该消息头用 expiry-date 和 rule-id 两个键值对描述对象的详细过期信息；否则不显示该头域。 **取值范围：** 时间格式为：EEE, dd MMM yyyy HH:mm:ss z 示例：expiry-date=Sun, 06 Nov 1994 08:49:37 GMT **默认取值：** 无
x-obs-website-redirect-location	String	**参数解释：** 当桶设置了 Website 配置，就可以设置对象元数据的这个属性，Website 接入点返回 301 重定向响应，将获取这个对象的请求重定向到该属性指定的桶内另一个对象或一个外部的 URL，该参数指明对象的重定向地址。 例如，重定向请求到桶内另一对象： x-obs-website-redirect-location:/anotherPage.html 或重定向请求到一个外部 URL： x-obs-website-redirect-location:http://www.example.com/ OBS 将这个值从头域中取出，保存在对象的元数据“WebsiteRedirectLocation”中。 **约束限制：** • 必须以“/”、“http://”或“https://”开头，长度不超过 2KB。 • OBS 仅支持为桶根目录下的对象设置

消息头名称	消息头类型	描述
		重定向，不支持为桶中文件夹下的对象设置重定向。 **默认取值：** 无
x-obs-delete-marker	Boolean	**参数解释：** 标识删除的对象是否是删除标记。如果不是，则响应中不会出现该消息头。 **取值范围：** • true：是删除标记。 • false：不是删除标记。 **默认取值：** false
x-obs-version-id	String	**参数解释：** 对象的版本号。 **约束限制：** 如果该对象无版本号，则响应中不会出现该消息头。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 配置服务端加密方式。示例：x-obs-server-side-encryption：kms **约束限制：** 无 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption

消息头名称	消息头类型	描述
		头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示解密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 无 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示解密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无
x-obs-object-type	String	**参数解释：** 对象的类型。 **约束限制：** 对象为非 Normal 对象时，会返回此头域。 **取值范围：** • Appendable：通过追加写生成的对象。

消息头名称	消息头类型	描述
		默认取值： 无
x-obs-next-append-position	Integer	**参数解释：** 指明下一次请求应该提供的 position。 **约束限制：** 对象为 Appendable 对象时，会返回此头域。 **取值范围：** 无 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例：下载整个对象

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

响应示例：下载整个对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
Accept-Ranges: bytes
ETag: "3b46eaf02d3b6b1206078bb86a7b7013"
Last-Modified: WED, 01 Jul 2015 01:20:29 GMT
Content-Type: binary/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQwxJ2I1VvxD/Xgwuw2G2RQax30gdXU
Date: WED, 01 Jul 2015 04:24:33 GMT
Content-Length: 4572

[4572 Bytes object content]
```

请求示例：指定 Range 下载对象

指定 Range 下载对象（下载对象单个区间内容）

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Sep 2020 09:59:04 GMT
Range:bytes=20-30
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mNPLWQMDWg30PTkAWiqJaLl3ALg=
```

指定 Range 下载对象（下载对象多个区间内容）

说明

所有 Range 区间的总长度不能超过对象内容总长度。

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Sep 2020 10:02:43 GMT
Range:bytes=20-30,40-50
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:ZwM7Vk2d7sD9o8zRsRKehgKQDkk=
```

响应示例：指定 Range 下载对象

指定 Range 下载对象（下载对象单个区间内容）

```
HTTP/1.1 206 Partial Content
Server: OBS
x-obs-request-id: 000001748C0DBC35802E360C9E869F31
Accept-Ranges: bytes
ETag: "2200446c2082f27ed2a569601ca4e360"
Last-Modified: Mon, 14 Sep 2020 01:16:20 GMT
Content-Range: bytes 20-30/4583
Content-Type: binary/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSn2JHu4okx9NBRNZAvBGawa3lt3g31g
Date: Mon, 14 Sep 2020 09:59:04 GMT
Content-Length: 11

[ 11 Bytes object content]
```

指定 Range 下载对象（下载对象多个区间内容）

```
HTTP/1.1 206 Partial Content
Server: OBS
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
Accept-Ranges: bytes
ETag: "2200446c2082f27ed2a569601ca4e360"
Last-Modified: Mon, 14 Sep 2020 01:16:20 GMT
Content-Type: multipart/byteranges;boundary=35bcf444-e65f-4c76-9430-7e4a68dd3d26
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSIBWFOVW8eeWujkqSnoIANC2mNR1cdF
Date: Mon, 14 Sep 2020 10:02:43 GMT
Content-Length: 288
```

```
--35bcf444-e65f-4c76-9430-7e4a68dd3d26
Content-type: binary/octet-stream
Content-range: bytes 20-30/4583
[ 11 Bytes object content]
--35bcf444-e65f-4c76-9430-7e4a68dd3d26
Content-type: binary/octet-stream
Content-range: bytes 40-50/4583
[ 11 Bytes object content]
--35bcf444-e65f-4c76-9430-7e4a68dd3d26
```

请求示例：下载缩放图片

```
GET /example.jpg?x-image-process=image/resize,w_100 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:20:51 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:9Nsx45WjaVxlLnxlO9awasXn83N=
```

响应示例：下载缩放图片

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
x-image-process:image/resize,w_100
Accept-Ranges: bytes
ETag: "3b46eaf02d3b6b1206078bb86a7b7013"
Last-Modified: WED, 01 Jul 2015 01:20:29 GMT
Content-Type: image/jpeg
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQwxJ2I1VvxD/Xgwuw2G2RQax30gdXU
Date: WED, 01 Jul 2015 04:20:51 GMT
Content-Length: 49

[ 49 Bytes object content]
```

请求示例：判断对象 *Etag* 值

如果对象 Etag 值匹配则下载该对象

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
If-Match: 682e760adb130c60c120da3e333a8b09
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

响应示例：判断对象 *Etag* 值，*Etag* 不匹配

如果存储的对象的 Etag 值不是 682e760adb130c60c120da3e333a8b09，则提示下载失败

```
HTTP/1.1 412 Precondition Failed
Server: OBS
```

```
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
Content-Type: application/xml
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQwxJ2I1VvxD/Xgwuw2G2RQax30gdXU
Date: WED, 01 Jul 2015 04:20:51 GMT

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Error>
  <Code>PreconditionFailed</Code>
  <Message>At least one of the pre-conditions you specified did not hold</Message>
  <RequestId>8DF400000163D3F2A89604C49ABEE55E</RequestId>
  <HostId>ha0ZGaSKVm+uLOrCXXtx4Qn1aLzvoeblctVXRAqA7pty10mzUUW/yOzFue04lBqu</HostId>
  <Condition>If-Match</Condition>
</Error>
```

响应示例：判断对象 *Etag* 值匹配，下载成功

如果存储的对象的 Etag 值是 682e760adb130c60c120da3e333a8b09，则下载成功

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 5DEB00000164A21E1FC826C58F6BA001
Accept-Ranges: bytes
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSbkdml1sLSvKnoHaRcOwRI+6+ustDwk
Date: Mon, 16 Jul 2015 08:04:00 GMT
Content-Length: 8

[ 8 Bytes object content]
```

请求示例：在 *URL* 中携带签名下载对象

```
GET
/object02?AccessKeyId=H4IPJX0TQTHTHEBQQCEC&Expires=1532688887&Signature=EQmDuOhaLUr
zrzRNZxwS72CXeXM%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Fri, 27 Jul 2018 10:52:31 GMT
```

响应示例：在 *URL* 中携带签名下载对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Date: Fri, 27 Jul 2018 10:52:31 GMT
Content-Length: 8
```

```
[ 8 Bytes object content]
```

请求示例：下载对象并重命名，使用 response-content-disposition 参数

下载对象并重命名，使用 response-content-disposition 参数实现

```
GET /object01?response-content-disposition=attachment; filename*=utf-8''name1
HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

响应示例：下载对象并重命名，使用 response-content-disposition 参数

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Date: Fri, 27 Jul 2018 10:52:31 GMT
Content-Length: 8
Content-Disposition: attachment; filename*=utf-8''name1

[ 8 Bytes object content]
```

请求示例：下载对象并重命名，使用 attname 参数

下载对象并重命名，使用 attname 参数实现

```
GET /object01?attname=name1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

响应示例：下载对象并重命名，使用 attname 参数

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Date: Fri, 27 Jul 2018 10:52:31 GMT
Content-Length: 8
Content-Disposition: attachment; filename*=utf-8''name1
```

```
[ 8 Bytes object content]
```

5.4.5 获取对象元数据

功能介绍

拥有对象读权限的用户可以执行 HEAD 操作命令获取对象元数据，返回信息包含对象的元数据信息。

获取采用 SSE-C 加密的对象的元数据时，需要携带 SSE-C 相关头域，详见表 5-113。

多版本

默认情况下，获取的是最新版本的对象元数据。如果最新版本的对象是删除标记，则返回 404。如果要获取指定版本的对象元数据，请求可携带 versionId 消息参数。

请求消息样式

```
HEAD /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

请求参数说明如表 5-112 所示。

表 5-112 请求消息参数

参数名称	参数类型	是否必选	描述
versionId	String	否	**参数解释：** 对象的版本号。 **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

另外该请求可以使用附加的消息头，具体如表 5-113 所示。

表 5-113 请求消息头

消息头名称	消息头类型	是否必选	描述
Origin	String	否	**参数解释：** 预请求指定的跨域请求 Origin（通常为域名）。 **约束限制：** 允许多条匹配规则，以回车换行为间隔。每个匹配规则允许使用最多一个“*”通配符。 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-Request-Headers	String	否	**参数解释：** 实际请求可以使用的 HTTP 头域，可以带多个头域。 **约束限制：** 允许的头域可设置多个，多个头域之间换行隔开，每行最多可填写一个*符号，不支持&、:、<、空格以及中文字符。 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示解密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-	String	否。当使	**参数解释：**

消息头名称	消息头类型	是否必选	描述
side-encryption-customer-key		用 SSE-C 方式时，必选。	在 SSE-C 加密方式下使用该头域，该头域表示解密使用的密钥。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示解密使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Content-Type: type
```

```
Date: date
Content-Length: length
Etag: etag
Last-Modified: time
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如下表 5-114 中的消息头。

表 5-114 附加响应消息头

消息头名称	消息头类型	描述
x-obs-expiration	String	**参数解释：** 对象的详细过期信息。 **约束限制：** 无 **取值范围：** 当对象单独设置了对象 lifecycle，过期时间以对象 lifecycle 为准，该消息头用 expiry-date 描述对象的详细过期信息；如果对象没有设置对象 lifecycle，设置了桶级别 lifecycle，过期时间以桶级别 lifecycle 为准，该消息头用 expiry-date 和 rule-id 两个键值对描述对象的详细过期信息；否则不显示该头域。 **默认取值：** 无
x-obs-website-redirect-location	String	**参数解释：** 当桶设置了 Website 配置，就可以设置对象元数据的这个属性，Website 接入点返回 301 重定向响应，将获取这个对象的请求重定向到该属性指定的桶内另一个对象或一个外部的 URL，该参数指明对象的重定向地址。 例如，重定向请求到桶内另一对象： x-obs-website-redirect-location:/anotherPage.html 或重定向请求到一个外部 URL： x-obs-website-redirect-location:http://www.example.com/ OBS 将这个值从头域中取出，保存在对象的元数据“WebsiteRedirectLocation”中。 **约束限制：** • 必须以“/”、“http://”或“https://”开

消息头名称	消息头类型	描述
		头，长度不超过 2KB。 • OBS 仅支持为桶根目录下的对象设置重定向，不支持为桶中文件夹下的对象设置重定向。 **默认取值：** 无
x-obs-version-id	String	**参数解释：** 对象的版本号。如果该对象无版本号，则响应中不会出现该消息头。 **约束限制：** 长度为 32 的字符串。 **取值范围：** 无 **默认取值：** 无
Access-Control-Allow-Origin	String	**参数解释：** 当桶设置了 CORS 配置，如果请求中的 Origin 满足服务端的 CORS 配置，则在响应中包含这个 Origin。 **约束限制：** 无 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-Allow-Headers	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 headers 满足服务端的 CORS 配置，则在响应中包含这个 headers。 **约束限制:** 最多可填写一个“*”通配符，不支持&、:、<、空格以及中文字符。 **取值范围：** 符合 CORS 协议的取值范围。 **默认取值：** 无
Access-Control-	Integer	**参数解释：**

消息头名称	消息头类型	描述
Max-Age		桶 CORS 规则中的 MaxAgeSeconds。MaxAgeSeconds 指请求来源的客户端可以对跨域请求返回结果的缓存时间。 **约束限制：** 每个 CORSRule 可以包含至多一个 MaxAgeSeconds。 **取值范围：** 大于等于 0 的整型数，单位：秒。 **默认取值：** 3000
Access-Control-Allow-Methods	String	**参数解释：** 当桶设置了 CORS 配置，如果请求的 Access-Control-Request-Method 满足服务端的 CORS 配置，则在响应中包含这条 rule 中的 Methods。 **取值范围：** • GET • PUT • HEAD • POST • DELETE
Access-Control-Expose-Headers	String	**参数解释：** 桶 CORS 规则中的 ExposeHeader。ExposeHeader 是指 CORS 规则允许响应中可返回的附加头域，给客户端提供额外的信息。默认情况下浏览器只能访问以下头域：Content-Length、Content-Type，如果需要访问其他头域，需要在附加头域中配置。 **约束限制：** 不支持*、&、:、<、空格以及中文字符。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 该头域表示服务端的加密方式。 示例：x-obs-server-side-encryption：kms **约束限制：** 如果服务端加密是 SSE-KMS 方式，响应包含

消息头名称	消息头类型	描述
		该头域。 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示解密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 无 **取值范围：** AES256：AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示解密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 由密钥值经过 MD5 加密再经过 Base64 编码后得到，示例：4XvB3tbNTN+tIEVa0/fGaQ==

消息头名称	消息头类型	描述
		取值范围： 密钥 ID MD5 的 base64 值。 **默认取值：** 无
x-obs-storage-class	String	**参数解释：** 对象的存储类型。 **约束限制：** 仅当对象为非标准存储类型时，会返回此头域。 **取值范围：** • WARM：低频存储 • COLD：归档存储 **默认取值：** 无
x-obs-restore	String	**参数解释：** 标识对象的恢复状态。 示例：正在恢复 ongoing-request="true"；已恢复 ongoing-request="false", expiry-date="Wed, 7 Nov 2012 00:00:00 GMT"。其中 expiry-date 表示对象恢复后的失效时间。 **约束限制：** • 仅当对象为归档存储类型，并且处于正在恢复或已经恢复时，会返回此头域。 **取值范围：** 无 **默认取值：** 无
x-obs-object-type	String	**参数解释：** 对象的类型。 **约束限制：** 仅当对象为非 Normal 对象时，会返回此头域。 **取值范围：** • Appendable：通过追加写生成的对象。 **默认取值：** 无
x-obs-next-append-	Integer	**参数解释：**

消息头名称	消息头类型	描述
position		指明下一次请求应该提供的 position。 **约束限制：** 仅当对象为 Appendable 对象时，会返回此头域。 **取值范围：** 无 **默认取值：** 无
x-obs-uploadId	String	**参数解释：** 表示对应的多段任务 ID。 **约束限制：** 仅当对象由多段上传任务合而来时，会返回此头域。 **取值范围：** 无 **默认取值：** 无
x-obs-object-lock-mode	String	**参数解释：** 应用于此对象的 WORM 模式。 示例：x-obs-object-lock-mode:COMPLIANCE **约束限制：** • 仅当对象有配置过对象级 WORM 保护策略或应用桶级默认 WORM 策略时返回该参数。 • 该用户具有 GetObjectRetention 权限。 **取值范围：** 目前仅支持 COMPLIANCE，即合规模式。 **默认取值：** 无
x-obs-object-lock-retain-until-date	String	**参数解释：** 此对象的锁定过期的截止时间。 示例：x-obs-object-lock-retain-until-date:2015-07-01T04:11:15.297Z **约束限制：** • 格式要求为 UTC 时间，并符合 ISO 8601 标准。如：2015-07-01T04:11:15.297Z。 • 仅当对象有配置过对象级 WORM 保护策

消息头名称	消息头类型	描述
		略或应用桶级默认 WORM 策略时返回该参数。 • 该用户具有 GetObjectRetention 权限。 **取值范围：** 设置该值时需要大于当前时间。 **默认取值：** 无
x-obs-replication-status	String	**参数解释：** 对象的复制状态。 **约束限制：** 仅记录最近一次的复制状态，不支持查看对象的历史复制状态。 **取值范围：** • FAILED： 源对象的状态，表示复制任务失败。 • COMPLETED： 源对象的状态，表示复制任务成功。 • REPLICA：目标对象的状态，表示该对象是副本对象。 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误；所有错误已经包含在表 6-2 中。

请求示例

```
HEAD /object1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:19:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:/cARjk81l2iExMfQqn6iT3qEZ74=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E4BB5905C41B6E65B6
```

```
Accept-Ranges: bytes
ETag: "3b46eaf02d3b6b1206078bb86a7b7013"
Last-Modified: WED, 01 Jul 2015 01:19:21 GMT
Content-Type: binary/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSD3nAiTaBoeyt9oHp9vTYtXnLDmwV6D
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 4572
```

5.4.6 删除对象

功能介绍

删除对象的操作。如果要删除的对象不存在，则仍然返回成功信息。

多版本

当桶的多版本状态是开启时，不指定版本删除对象将产生一个带唯一版本号的删除标记，并不删除对象；当桶的多版本状态是 Suspended 时，不指定版本删除将删除版本号为 null 的对象，并将产生一个版本号为 null 的删除标记。

如果要删除指定版本的对象，请求可携带 versionId 消息参数。

WORM

由于打开了 WORM 开关的桶默认开启了多版本，当您不指定版本号进行删除对象操作时，由于多版本机制并不会真正删除此对象，您的操作可以成功并为最新版本的对象产生一个带唯一版本号的删除标记。当您指定版本号进行删除对象操作时，如果对象处于 WORM 保护状态，WORM 的保护模式会限制对象的删除并返回 403。删除标记不会被 WORM 保护。

请求消息样式

```
DELETE /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

请求参数说明如表 5-115 所示。

须知

删除对象时，请求消息参数仅支持表 5-115 中列出的参数信息，如果包含了 OBS 无法识别的参数信息，服务端将返回 400 错误。

表 5-115 请求消息参数

参数名称	描述	是否必选

参数名称	描述	是否必选
versionId	**参数解释：** 待删除对象的版本号。 类型：String **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无	否

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，如果开启了多版本功能，还可能使用如下表 5-116 中的消息头。

表 5-116 附加响应消息头

消息头名称	描述
x-obs-delete-marker	**参数解释：** 标识对象是否标记删除。如果不是，则响应中不会出现该消息头。 类型：Boolean **约束限制：** 无 **取值范围：** • true：是标记删除 • false：不是标记删除 **默认取值：**

消息头名称	描述
	false
x-obs-version-id	**参数解释：** 对象的版本号。如果该对象无版本号，则响应中不会出现该消息头。 类型：String **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
DELETE /object2 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:19:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:MfK9JCnSFHCrJmjv7iRkRrrce2s=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 8DF400000163D3F51DEA05AC9CA066F1
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSgkM4Dij80gAeFY8pAZIwx72QhDeBZ5
Date: WED, 01 Jul 2015 04:19:21 GMT
```

5.4.7 批量删除对象

功能介绍

批量删除对象特性用于将一个桶内的多个对象一次性删除，删除后不可恢复。

批量删除对象支持两种响应方式：

- quiet 模式（Quiet 参数设置为 True）：quiet 模式是指在返回响应时，只返回删除失败的对象结果，没有返回的认为删除成功。
- 非 quiet 模式（系统默认，或 Quiet 参数设置为 False）：非 quiet 模式是指在返回响应时，不管对象是否删除成功都将删除结果包含在 XML 响应里。

批量删除的请求消息头中必须包含 Content-MD5 以及 Content-Length，用以保证请求的消息体在服务端检测到网络传输如果有错，则可以检测出来。

请求消息样式

```
POST /?delete HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-MD5: MD5
Content-Length: length

<?xml version="1.0" encoding="UTF-8"?>
<Delete>
   <Quiet>true</Quiet>
   <Object>
      <Key>Key</Key>
      <VersionId>VersionId</VersionId>
   </Object>
   <Object>
      <Key>Key</Key>
   </Object>
</Delete>
```

请求消息参数

该请求的请求消息中不使用消息参数。

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求消息元素

该请求通过在请求消息的消息元素中指定要批量删除的对象列表，元素的具体含义如表 5-117 所示。

表 5-117 请求消息元素

参数名称	是否必选	参数类型	描述
Delete	是	XML	**参数解释：** 待删除的对象列表，Delete 是 Object、Quiet 的父节点。 **约束限制：** 不涉及

参数名称	是否必选	参数类型	描述
			取值范围： 请详见表 5-118。 **默认取值：** 不涉及

表 5-118 Delete 参数说明

参数名称	是否必选	参数类型	描述
Quiet	否	Boolean	**参数解释：** 用于指定使用 quiet 模式，只返回删除失败的对象结果。 **约束限制：** 如果不设置此参数，取值默认为 False，取值为 False 则被系统忽略掉。 **取值范围：** • True：只返回删除失败的对象结果。 • False：返回所有删除对象结果。 **默认取值：** False
Object	是	XML	**参数解释：** 待删除的对象，Object 是 Key、VersionId 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-119。 **默认取值：** 不涉及

表 5-119 Object 参数说明

参数名称	是否必选	参数类型	描述
Key	是	String	**参数解释：** 待删除的对象名。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：**

参数名称	是否必选	参数类型	描述
			不涉及
VersionId	否	String	**参数解释：** 待删除的对象版本号。获取对象版本号的步骤请参考 5.1.3 列举桶内对象。 **约束限制：** 不涉及 **取值范围：** 长度为 32 的字符串。 **默认取值：** 不涉及

批量删除对象一次能接收最大对象数目为 1000 个，如果超出限制，服务端会返回请求不合法。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Type: application/xml
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DeleteResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Deleted>
   <Key>Key</Key>
</Deleted>
<Error>
   <Key>Key</Key>
   <Code>ErrorCode</Code>
   <Message>Message</Message>
</Error>
</DeleteResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应通过消息元素返回删除的结果，元素的具体意义如表 5-120 中所示。

表 5-120 响应消息元素

参数名称	参数类型	描述
DeleteResult	Container	**参数解释：**

参数名称	参数类型	描述
		批量删除对象响应消息的根节点，DeleteResult 是 Deleted、Error、EncodingType 的父节点。 **取值范围：** 请详见表 5-121。

表 5-121 DeleteResult 参数说明

参数名称	参数类型	描述
Deleted	Container	**参数解释：** 删除成功结果的容器，Deleted 是 Key、VersionId、DeleteMarker、DeleteMarkerVersionId 的父节点。 **取值范围：** 请详见表 5-122。
Error	Container	**参数解释：** 删除失败结果的容器，Error 是 Key、VersionId、Code、Message 的父节点。 **取值范围：** 请详见表 5-123。

表 5-122 Deleted 参数解释

参数名称	参数类型	描述
Key	String	**参数解释：** 每个删除结果的对象名。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。
VersionId	String	**参数解释：** 删除对象的版本号。 **取值范围：** 长度为 32 的字符串。
DeleteMarker	Boolean	**参数解释：** 当批量删除请求访问的桶是多版本桶时，如果创建或删除一个删除标记，返回消息中该元素的值为 true。 当桶的多版本状态是开启时，不指定版本删除对象将产生一个带唯一版本号的删除标记，并不删除对象；当桶的多版本状态是 Suspended 时，不指定版本删除将删除版本号为 null 的对象，并将产生一个版本号为 null 的删除标记。

参数名称	参数类型	描述
		取值范围： true
DeleteMarkerVersionId	String	**参数解释：** 请求创建或删除的删除标记版本号。 当批量删除请求访问的桶是多版本桶时，如果创建或删除一个删除标记，响应消息会返回该元素。该元素在以下两种情况中会出现： • 用户发送不带版本删除请求，即请求只有对象名，无版本号。这种情况下，系统会创建一个删除标记，并在响应中返回该删除标记的版本号。 • 用户发送带版本删除请求，即请求同时包含对象名以及版本号，但是该版本号标识一个删除标记。这种情况下，系统会删除此删除标记，并在响应中返回该删除标记的版本号。 **取值范围：** 长度为 32 的字符串。

表 5-123 Error 参数解释

参数名称	参数类型	描述
Key	String	**参数解释：** 每个删除结果的对象名。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。
Code	String	**参数解释：** 删除失败结果的错误码。 **取值范围：** 不涉及
Message	String	**参数解释：** 删除失败结果的错误消息。 **取值范围：** 不涉及
VersionId	String	**参数解释：** 删除对象的版本号。 **取值范围：** 长度为 32 的字符串。

错误响应消息

1、用户收到请求后首先进行 XML 的解析，如果超过 1000 个对象返回 400 Bad Request。

2、如果 XML 消息体中包含的对象 Key 不合法，比如长度超过了经过 UTF-8 编码后的 1024 字符序列，OBS 返回 400 Bad Request。

3、如果请求消息头中不包含 Content-MD5，OBS 则返回 400 Bad Request。

其他错误已经包含在表 6-2 中。

请求示例

```
POST /test333?delete HTTP/1.1
User-Agent: curl/7.29.0
Host: 127.0.0.1
Accept: */*
Date: WED, 01 Jul 2015 04:34:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:8sjZWJlWmYmYnK5JqXaFFQ+vHEg=
Content-MD5: ZPzz8L+hdRJ6qCqYbU/pCw==
Content-Length: 188

<?xml version="1.0" encoding="utf-8"?>
<Delete>
  <Quiet>true</Quiet>
  <Object>
    <Key>obja02</Key>
  </Object>
  <Object>
    <Key>obja02</Key>
  </Object>
</Delete>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3FE4CE80340D30B0542
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCRhY0FBWRm6qjOE1ACBZwS+0KYlPBq0f
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:34:21 GMT
Content-Length: 120

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DeleteResult xmlns="http://obs.example.com/doc/2015-06-30/"/>
```

相关文档

调用批量删除对象接口操作涉及计费，该操作相关计费说明请参考。

5.4.8 恢复归档存储对象

功能介绍

如果要获取归档存储对象的内容，需要先将对象恢复，然后再执行下载数据的操作。对象恢复后，会产生一个标准存储类型的对象副本，也就是说会同时存在标准存储类型的对象副本和归档存储类型的对象，在恢复对象的保存时间到期后标准存储类型的对象副本会自动删除。

多版本

默认情况下，恢复的是最新版本的对象。如果最新版本的对象是删除标记，则返回 404。如果要恢复指定版本的对象，请求可携带 versionId 消息参数。

请求消息样式

```
POST /ObjectName?restore&versionId=VersionID HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
Content-MD5: MD5

<RestoreRequest>
   <Days>NumberOfDays</Days>
   <RestoreJob>
      <Tier>RetrievalOption</Tier>
   </RestoreJob>
</RestoreRequest>
```

请求消息参数

参数名称	参数类型	是否必选	描述
versionId	String	否	**参数解释：** 待恢复归档存储对象的版本号。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无，如果不设置则默认指定最新版本的对象。

请求消息头

该请求使用公共消息头，具体请参见表 3-3。

请求消息元素

表 5-124 请求消息元素表

元素名称	元素类型	是否必选	描述
RestoreRequest	Container	是	**参数解释：** 恢复信息的容器。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Days	Integer	是	**参数解释：** 恢复对象后，会生成一个对象的标准存储副本，此参数指定恢复有效期，即标准存储副本的保存时间。 **约束限制：** 保存时间是从对象发起取回后的第一个北京时间 08:00 开始计算的。 以保存时间为一天为例，取回发起时间在 27 号 08:00 到 28 号 08:00 之间的取回，实际过期时间都是 29 号的 08:00。 **取值范围：** [1, 30]，单位：天。 **默认取值：** 无
RestoreJob	Container	否	**参数解释：** 恢复选项的容器。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Tier	String	否	**参数解释：** 恢复选项，用户可根据需求选择恢复选项，不同的恢复选项恢复耗时不同。

元素名称	元素类型	是否必选	描述
			取值范围： • Expedited：表示快速恢复对象，归档存储恢复耗时 1~5 min。 • Standard：表示标准恢复对象，归档存储恢复耗时 3~5 h。 **默认取值：** Standard

响应消息样式

```
HTTP/1.1 status_code
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带有响应元素。

错误响应消息

表 5-125 对象存储访问服务错误码列表

错误码	描述	HTTP 状态码
RestoreAlreadyInProgress	**参数解释：** 对象正在恢复，请求冲突。 ErrorMessage: Object restore is already in progress	409 Conflict
ObjectHasAlreadyRestored	**参数解释：** 已经恢复的对象，禁止缩短恢复保存时间。 ErrorMessage:After restoring an archived object, you cannot shorten the restoration period of the archived object	409 Conflict
MalformedXML	**参数解释：** Days 字段不合法（不为整数）。 ErrorMessage: The XML you provided was not well-formed or did not validate against our published schema	400 Bad Request
InvalidArgument	**参数解释：** Days 字段取值超出范围（1<=days<=30)。	400 Bad Request

错误码	描述	HTTP 状态码
	ErrorMessage: restoration days should be at least 1 and at most 30	
MalformedXML	**参数解释：** Tier 字段不合法。 ErrorMessage: The XML you provided was not well-formed or did not validate against our published schema	400 Bad Request
InvalidObjectState	**参数解释：** 恢复的对象不是归档存储对象。 ErrorMessage: Restore is not allowed, as object's storage class is not COLD	403 Forbidden

请求示例

```
POST /object?restore HTTP/1.1
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:39:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:kaEwOixnSVuS6If3Q0Lnd6kxm5A=
Content-Length: 183

<RestoreRequest>
   <Days>2</Days>
   <RestoreJob>
     <Tier>Expedited</Tier>
   </RestoreJob>
</RestoreRequest>
```

响应示例

```
HTTP/1.1 202 Accepted
Server: OBS
x-obs-request-id: A2F500000163F374CCBB2063F834C6C4
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSLbWIs23RR95NVpkbWlJdlm8Dq+wQBw
Date: WED, 01 Jul 2015 04:39:46 GMT
Content-Length: 0
```

5.4.9 追加写对象

功能介绍

追加写对象操作是指在指定桶内的一个对象尾追加上传数据，不存在相同对象键值的对象则创建新对象。

通过 Append Object 操作创建的 Object 类型为 Appendable Object，而通过 Put Object 上传的 Object 是 Normal Object。

说明

用户上传的对象存储在桶中。用户必须对桶有 WRITE 权限，才可以在桶中上传对象。同一个桶中存储的对象名必须是唯一的。

为了确保数据在传输过程中没有遭到破坏，用户可以在请求消息头中加入 Content-MD5 参数，OBS 收到上传数据后，会对数据进行 MD5 校验，如果不一致则返回出错信息。

该操作支持在创建 Appendable 对象时指定 x-obs-acl 参数，设置对象的权限控制策略。

该操作支持服务端加密功能。

和其他操作的关系

1. 对一个已经存在的 Appendable 对象进行 Put Object 操作，那么该 Appendable 对象会被新 Object 覆盖，类型变为 Normal 对象，反之出错。
2. Appendable 对象复制后变成 Normal 对象，不支持 Appendable 对象复制成 Appendable 对象。

WORM

在开启了 WORM 开关的桶中，使用追加写上传对象将失败并返回 403。

约束

1. 每次追加上传都会更新该对象的最后修改时间。
2. 服务端加密 SSE-C 方式，那么追加上传和初始化段一样，设置诸如 x-obs-server-side-encryption 之类的请求 Header，后续追加上传也必须携带。
3. 服务端加密 SSE-KMS 方式，有且只有第一次上传且桶内不存在同名对象时，才设置诸如 x-obs-server-side-encryption 之类的请求 Header，后续追加上传不携带。
4. 每次追加上传的长度不能超过对象长度上限 5G 的限制。
5. 每个 Appendable 对象追加写次数最多为 10000 次。
6. 如果对象存储类型为 COLD（归档存储），则不能调用该接口。
7. 如果桶设置了跨区域复制配置，则不能调用该接口。
8. 并行文件系统不支持追加写对象。

请求消息样式

```
POST /ObjectName?append&position=Position HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Type: application/xml
Content-Length: length
Authorization: authorization
Date: date
<Optional Additional Header>
<object Content>
```

请求消息参数

该请求需要在消息中指定参数，表明这是追加写上传，同时指定本次追加上传位置，参数的具体意义如表 5-126 所示

表 5-126 请求消息参数

参数名称	参数类型	是否必选	描述
append	String	是	**参数解释：** 表明这是以追加写方式上传对象。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
position	Integer	是	**参数解释：** 追加写位置。 **约束限制：** 不能超过 5GB。 **取值范围：** 需要追加写的对象首次上传时就必须指定 position 为 0，下次追加需要填写的 position 会在服务端本次上传成功返回消息的头域 x-obs-next-append-position 中携带。 **默认取值：** 无

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求参数 position=0 时，该请求可以使用的附加消息头，具体如表 5-127 所示。

该请求可以使用的服务端加密请求消息头，具体如表 5-128 所示。

表 5-127 请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-acl	String	否	**参数解释：** 第一次写时，可以加上此消息头设置对象的权限控制策略，使用的策略为预定义的常用策

消息头名称	消息头类型	是否必选	描述
			略。 **约束限制：** 预定义策略为字符串形式。 **取值范围：** • private • public-read • public-read-write **默认取值：** private
x-obs-grant-read	String	否	**参数解释：** 第一次写时，使用此头域可以授权读对象、获取对象元数据的权限给 domain 下的所有用户。 示例：x-obs-grant-read: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 无 **默认取值：** 无
x-obs-grant-read-acp	String	否	**参数解释：** 第一次写时，使用此头域授权获取对象 ACL 的权限给 domain 下的所有用户。 示例：x-obs-grant-read-acp: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 无 **默认取值：** 无
x-obs-grant-write-acp	String	否	**参数解释：** 第一次写时，使用此头域授权写对象 ACL 的权限给 domain 下的所有用户。 示例：x-obs-grant-write-acp: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。

消息头名称	消息头类型	是否必选	描述
			取值范围： 无 **默认取值：** 无
x-obs-grant-full-control	String	否	**参数解释：** 第一次写时，使用此头域授权读对象、获取对象元数据、获取对象 ACL、写对象 ACL 的权限给 domain 下的所有用户。 示例：x-obs-grant-full-control: id=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 无 **默认取值：** 无
x-obs-storage-class	String	否	**参数解释：** 第一次写时，可以加上此头域设置对象的存储类型。 示例：x-obs-storage-class：STANDARD **约束限制：** • 归档存储对象不支持追加上传。 • 设置对象的存储类型时请注意大小写敏感。 **取值范围：** • STANDARD • WARM **默认取值：** 如果未设置此头域，则以桶的默认存储类型作为对象的存储类型。
x-obs-meta-*	String	否	**参数解释：** 第一次写时，可以在 HTTP 请求中加入以“x-obs-meta-”开头的消息头，用来加入自定义的元数据，以便对对象进行自定义管理。当用户获取此对象或查询此对象元数据时，加入的自定义元数据将会在返回消息的头中出现。 示例：x-obs-meta-test：test metadata **约束限制：** HTTP 请求不包含消息体，长度不能超过

消息头名称	消息头类型	是否必选	描述
			8KB。 **取值范围：** 无 **默认取值：** 无
x-obs-website-redirect-location	String	否	**参数解释：** 当桶设置了 Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的 URL，OBS 将这个值从头域中取出，保存在对象的元数据中。 **约束限制：** 必须以“/”、“http://”或“https://”开头，长度不超过 2K。 **取值范围：** 无 **默认取值：** 无
x-obs-expires	Integer	否	**参数解释：** 表示对象的过期时间，单位是天。过期之后对象会被自动删除。（从对象最后修改时间开始计算） 示例：x-obs-expires：3 **约束限制：** 设置的天数计算出的过期时间不能早于当前时间，如 10 天前上传的对象，不能设置小于 10 的值。 **取值范围：** 大于 0 的整数值。 **默认取值：** 无

表 5-128 服务端加密请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-server-side-encryption	String	否。当使用 SSE-KMS 方式时，必	**参数解释：** 使用该头域表示服务端加密是 SSE-KMS 方式。

消息头名称	消息头类型	是否必选	描述
		选。	示例：x-obs-server-side-encryption：kms **约束限制：** 无 **取值范围：** • kms • AES256 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	否	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的密钥。该密钥用于加密对象。 示例：x-obs-server-side-encryption-

消息头名称	消息头类型	是否必选	描述
			customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中是否出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
ETag: etag
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

说明

ETag 返回的是本次追加上传数据的 Hash 值，不是整个对象的 Hash 值。

表 5-129 附加响应消息头

消息头名称	消息头类型	描述
x-obs-version-id	String	**参数解释：** 对象的版本号。如果桶的多版本状态为开启，则会返回对象的版本号。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 该头域表示服务端的加密方式。 示例：x-obs-server-side-encryption：kms **约束限制：** 如果服务端加密是 SSE-KMS 方式，响应包含该头域。 **取值范围：** • kms • AES256 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。

消息头名称	消息头类型	描述
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 无 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示加密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 无 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无
x-obs-next-append-position	Integer	**参数解释：** 指明下一次请求应该提供的 position。 **约束限制：** 对象为 Appendable 对象时，会返回此头域。 **取值范围：** 无 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

1. 如果本次追加上传使对象长度超过对象长度限制，OBS 返回 400 Bad Request，错误码为 AppendTooLarge。
2. 如果 position 的值和当前对象的原始长度不一致，OBS 返回 409 Conflict，错误码为 PositionNotEqualToLength。
3. 如果指定桶中存在相同对象键值的对象，对象类型非 Appendable，OBS 返回 409 Conflict，错误码为 ObjectNotAppendable。
4. 如果对象追加写次数超过 10000 次，OBS 返回 409 Conflict，错误码为 ObjectNotAppendable。
5. 如果对象存储类型为 COLD（归档存储），则不能调用该接口，否则 OBS 返回 409 Conflict，错误码为 ObjectNotAppendable。
6. 如果桶设置了跨区域复制配置，则不能调用该接口，否则 OBS 返回 400 Bad Request，错误码为 OperationNotSupported。

其他错误已包含在表 6-2 中。

请求示例：普通追加写对象

```
POST /object?append&position=0 HTTP/1.1
Host: examplebucket.obs.region.example.com
Expires: Wed, 27 Jun 2015 13:45:50 GMT
Date: Wed, 08 Jul 2015 06:57:01 GMT
Content-Type: image/jpg
Content-Length: 1458
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:kZoYNv66bsmc10+dcGKw5x2PRrk=

[1458 bytes of object data]
```

响应示例：普通追加写对象

```
HTTP/1.1 200 OK
Date: Wed, 27 Jun 2015 13:45:50 GMT
ETag: "d41d8cd98f00b204e9800998ecf8427e"
Content-Length: 0
Server: OBS
x-obs-request-id: 8DF400000163D3F0FD2A03D2D30B0542
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCTjCqTmsA1XRpIrmrJdvcEWvZyjbztdd
x-obs-next-append-position: 1458
```

请求示例：带 *redirect* 和自定义头域追加写对象

用户存在桶 examplebucket，对象 obj001 不存在，通过追加写接口创建新对象， 设置重定向头域"x-obs-website-redirect-location":"http://www.example.com/"，自定义头域"x-obs-meta-redirect":"redirect"，请求为

```
POST /obj001?append&position=0 HTTP/1.1
Host: examplebucket.obs.region.example.com
Expires: Wed, 27 Jun 2015 13:45:50 GMT
Date: Wed, 08 Jul 2015 06:57:01 GMT
x-obs-website-redirect-location: http://www.example.com/
x-obs-meta-redirect: redirect
```

```
Content-Length: 6
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:kZoYNv66bsmc10+dcGKw5x2PRrk=

[6 bytes of object data]
```

响应示例：带 redirect 和自定义头域追加写对象

```
HTTP/1.1 200 OK
Date: Wed, 27 Jun 2015 13:45:50 GMT
ETag: "9516dfb15f51c7ee19a4d46b8c0dbe1d"
Content-Length: 0
Server: OBS
x-obs-request-id: 5DEB00000164A3150AC36F8F0C120D50
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSrVlTYwsA4p9GEW+LYqotSl5BYDxHfT
x-obs-next-append-position: 6
```

5.4.10 设置对象 ACL

功能介绍

OBS 支持对对象的操作进行权限控制。默认情况下，只有对象的创建者才有该对象的读写权限。用户也可以设置其他的访问策略，比如对一个对象可以设置公共访问策略，允许所有人对其都有读权限。SSE-KMS 方式加密的对象即使设置了 ACL，跨租户也不生效。

OBS 用户在上传对象时可以设置权限控制策略，也可以通过 ACL 操作 API 接口对已存在的对象更改或者获取 ACL(access control list) 。一个对象的 ACL 最多支持 100 条 Grant 授权。

本节将介绍如何更改对象 ACL，改变对象的访问权限。

多版本

默认情况下，更改的是最新版本的对象 ACL。要设置指定版本的对象 ACL，请求可以带参数 versionId。

请求消息格式

```
PUT /ObjectName?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization

<AccessControlPolicy>
   <Owner>
      <ID>ID</ID>
   </Owner>
   <Delivered>true</Delivered>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>ID</ID>
```

```
            </Grantee>
            <Permission>permission</Permission>
        </Grant>
    </AccessControlList>
</AccessControlPolicy>
```

请求消息参数

请求参数说明如表 5-130 所示。

表 5-130 请求消息参数

参数名称	是否必选	参数类型	描述
versionId	否	String	**参数解释：** 对象的版本号。表示更改指定版本对象的 ACL。如何获取对象的版本 ID 请参见 5.1.3 列举桶内对象。 **约束限制：** 不涉及 **取值范围：** 长度为 32 的字符串。 **默认取值：** 不涉及，如果不设置则默认修改最新版本的对象。

请求消息头

该请求使用公共请求消息头，具体参见表 3-3。

请求消息元素

该请求消息通过带消息元素来传递对象的 ACL 信息，元素的意义如表 5-131 所示。

表 5-131 请求消息元素

参数名称	是否必选	参数类型	描述
AccessControlList	是	XML	**参数解释：** 访问控制列表，AccessControlList 是 Grant、 Grantee、Permission 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-132。

参数名称	是否必选	参数类型	描述
			默认取值： 不涉及
Owner	是	XML	**参数解释：** 桶的所有者信息，Owner 是 ID 的父节点。 **约束限制：** 不涉及 **取值范围：** 请详见表 5-133。 **默认取值：** 不涉及
Canned	否	String	**参数解释：** 向所有人授予权限。 **约束限制：** 不涉及 **取值范围：** Everyone **默认取值：** 不涉及

表 5-132 AccessControlList 参数说明

参数名称	是否必选	参数类型	描述
Grant	否	XML	**参数解释：** 用于标记用户及用户的权限，Grant 是 Grantee、Delivered 的父节点。 **约束限制：** 单个对象的 ACL，Grant 元素不能超过 100 个。 **取值范围：** 请详见表 5-134。 **默认取值：** 不涉及
Grantee	否	XML	**参数解释：** 记录被授权用户信息。 **约束限制：** 不涉及

参数名称	是否必选	参数类型	描述
			取值范围： 不涉及 **默认取值：** 不涉及
Permission	否	String	**参数解释：** 授予对象的权限。 **约束限制：** 不涉及 **取值范围：** • READ：允许被授权者获取对象内容和元数据。 • READ_ACP：允许被授权者读取对象 ACL 属性。 • WRITE_ACP：允许被授权者更新对象 ACL 属性。 • FULL_CONTROL：允许被授予者对象的 READ、READ_ACP 和 WRITE_ACP 权限。 **默认取值：** 不涉及

表 5-133 Owner 参数说明

参数名称	是否必选	参数类型	描述
ID	是	String	**参数解释：** 被授权用户的租户 ID。 **约束限制：** 不涉及 **取值范围：** 如何获取用户的 DomainId 请详见 8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID。 **默认取值：** 不涉及

表 5-134 Grant 参数说明

参数名称	是否必选	参数类型	描述
Grantee	否	XML	**参数解释：**

参数名称	是否必选	参数类型	描述
			记录被授权用户信息。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及
Delivered	否	Boolean	**参数解释：** 对象 ACL 是否继承桶的 ACL。 **约束限制：** 不涉及 **取值范围：** • true：对象继承桶 ACL • false：对象不继承桶 ACL **默认取值：** true

响应消息样式

```
HTTP/1.1 status_code
Content-Length: length
Content-Type: application/xml
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如表 5-135 中的消息头。

表 5-135 附加响应消息头

参数名称	参数类型	描述
x-obs-version-id	String	**参数解释：** 被更改 ACL 的对象的版本号。 **取值范围：** 长度为 32 的字符串。

响应消息元素

该请求的响应消息中不带有消息元素。

错误响应消息

该请求的响应无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /obj2?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:42:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:8xAODun1ofjkwHm8YhtN0QEcy9M=
Content-Length: 727

<AccessControlPolicy xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <Delivered>false</Delivered>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <Canned>Everyone</Canned>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3F0FD2A03D2D30B0542
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCTjCqTmsA1XRpIrmrJdvcEWvZyjbztdd
Date: WED, 01 Jul 2015 04:42:34 GMT
Content-Length: 0
```

5.4.11 获取对象 ACL

功能介绍

用户执行获取对象 ACL 的操作，返回信息包含指定对象的权限控制列表信息。用户必须拥有对指定对象读 ACP(access control policy)的权限，才能执行获取对象 ACL 的操作。

多版本

默认情况下，获取最新版本的对象 ACL。如果最新版本的对象是删除标记，则返回 404。如果要获取指定版本的对象 ACL，请求可携带 versionId 消息参数。

请求消息样式

```
GET /ObjectName?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求需要在请求消息参数中指定是在获取对象 ACL，参数意义如表 5-136 所示。

表 5-136 请求消息参数

参数名称	是否必选	参数类型	描述
acl	是	String	**参数解释：** 指定该请求是获取对象的 ACL。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及
versionId	否	String	**参数解释：** 指定对象的版本号。如何获取对象的版本 ID 请参见 5.1.3 列举桶内对象。 **约束限制：** 不涉及 **取值范围：** 长度为 32 的字符串。 **默认取值：** 不涉及，如果不设置则默认修改最新版本的对

参数名称	是否必选	参数类型	描述
			象。

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Owner>
      <ID>id</ID>
   </Owner>
   <Delivered>true</Delivered>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>id</ID>
         </Grantee>
         <Permission>permission</Permission>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如下表 5-137 中的消息头。

表 5-137 附加响应消息头

参数名称	参数类型	描述
x-obs-version-id	String	**参数解释：** 指定对象的版本号。 **取值范围：** 长度为 32 的字符串。

响应消息元素

该请求的响应消息中通过消息元素返回对象的 ACL 信息，元素的具体意义如表 5-138 所示。

表 5-138 响应消息元素

参数名称	参数类型	描述
AccessControlList	XML	**参数解释：** 访问控制列表，记录了对该桶有访问权限的用户列表。AccessControlList 是 Grant、 Grantee、Permission 的父节点。 **取值范围：** 请详见表 5-139。
ID	String	**参数解释：** 用户的 DomainId。 **取值范围：** 不涉及
Delivered	Boolean	**参数解释：** 对象 ACL 是否继承桶的 ACL。 **取值范围：** • true：对象继承桶 ACL • false：对象不继承桶 ACL

表 5-139 AccessControlList 参数说明

参数名称	参数类型	描述
Grant	XML	**参数解释：** 用于标记用户及用户的权限，Grant 是 Grantee、Delivered 的父节点。 **取值范围：** 不涉及
Grantee	XML	**参数解释：** 记录用户信息。 **取值范围：** 不涉及
Permission	String	**参数解释：** 指定的用户对该对象所具有的操作权限。 **取值范围：** • READ：允许被授权者获取对象内容和元数

参数名称	参数类型	描述
		据。 • READ_ACP：允许被授权者读取对象 ACL 属性。 • WRITE_ACP：允许被授权者更新对象 ACL 属性。 • FULL_CONTROL：允许被授予者对象的 READ、READ_ACP 和 WRITE_ACP 权限。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
GET /object011?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:45:55 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:YcmvNQxItGjFeeC1K2HeUEp8MMM=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E650F3065C2295674C
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCS+wsHqRuA2Tx+mXUpNtBbWLPMle9CIx
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:45:55 GMT
Content-Length: 769

<?xml version="1.0" encoding="utf-8"?>
<AccessControlPolicy xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <Delivered>false</Delivered>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
      </Grant>
    <Grant>
```

```
    <Grantee>
      <Canned>Everyone</Canned>
    </Grantee>
    <Permission>READ_ACP</Permission>
   </Grant>
  </AccessControlList>
</AccessControlPolicy>
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSmpL2dv6zZLM2HmUrXKTAi258MPqmrp
x-obs-request-id: 0000018A2A73AF59D3085C8F8ABF0C65
Server: OBS
Content-Length: 0
Date: WED, 01 Jul 2015 02:37:22 GMT
x-obs-version-id: G001118A6803675AFFFFD3043F7F91D0

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy  xmlns="http://obs.myhwclouds.com/doc/2015-06-30/">
   <Owner>
      <ID>d6s58yhnm83f3081577800575ee4cf</ID>
   </Owner>
   <Delivered>false</Delivered>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>f262a63g69422e8f330af1349c588f</ID>
         </Grantee>
         <Permission>READ</Permission>
      </Grant>
      <Grant>
         <Grantee>
            <ID>c965gfda2a849422e8f3985562432dsaa</ID>
         </Grantee>
         <Permission>FULL_CONTROL</Permission>
      </Grant>
      <Grant>
         <Grantee>
            <Canned>Everyone</Canned>
         </Grantee>
         <Permission>READ</Permission>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

5.4.12 修改对象元数据

功能介绍

用户可以通过本接口添加、修改或删除桶中已经上传的对象的元数据。

请求消息样式

```
PUT /ObjectName?metadata HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Type: application/xml
Content-Length: length
```

```
Authorization: authorization
Date: date
<Optional Additional Header>
<object Content>
```

请求消息参数

表 5-140 请求消息参数

参数名称	参数类型	是否必选	描述
versionId	String	否	**参数解释：** 对象的版本号。 **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

请求消息头

说明

OBS 支持在修改对象元数据的请求里携带 HTTP 协议规定的 6 个请求头：Cache-Control、Expires、Content-Encoding、Content-Disposition、Content-Type、Content-Language，OBS 会直接将这些头域的值保存在对象元数据中，在下载对象或者 HEAD 对象的时候，这些保存的值将会被设置到对应的 HTTP 头域中返回客户端。

表 5-141 请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-metadata-directive	String	是	**参数解释：** 元数据操作指示符。 **约束限制：** 如果您想要通过修改对象元数据的方式修改对象存储类别，x-obs-metadata-directive 必须使用 REPLACE_NEW。 **取值范围：** • REPLACE_NEW：表示对于已经存在值的元数据进行替换，不存在值的元数据进行赋值，未指定的元数据保持不变（自定义头域作替换处理）。 • REPLACE：表示使用当前请求中携带的头域完整替换，未指定的元数据（本表中

消息头名称	消息头类型	是否必选	描述
			除 x-obs-storage-class 的其它元数据）会被删除。 **默认取值：** 无
Cache-Control	String	否	**参数解释：** 指定对象被下载时的网页的缓存行为。 **约束限制：** 无 **取值范围：** 参见 HTTP 标准头域 Cache-Control 的取值。 **默认取值：** 无
Content-Disposition	String	否	**参数解释：** 指定对象被下载时的名称。 **约束限制：** 无 **取值范围：** 参见 HTTP 标准头域 Content-Disposition 的取值。 **默认取值：** 无
Content-Encoding	String	否	**参数解释：** 指定对象被下载时的内容编码格式。 **约束限制：** 无 **取值范围：** 参见 HTTP 标准头域 Content-Encoding 的取值。 **默认取值：** 无
Content-Language	String	否	**参数解释：** 指定对象被下载时的内容语言格式。 **约束限制：** 无 **取值范围：**

消息头名称	消息头类型	是否必选	描述
			参见 HTTP 标准头域 Content-Language 的取值。 **默认取值：** 无
Content-Type	String	否	**参数解释：** 指定对象被下载时的文件类型。 **约束限制：** 无 **取值范围：** Content-type 的常见取值参见《对象存储服务用户指南》中的“配置和查看对象元数据”章节内容。 **默认取值：** 无
Expires	String	否	**参数解释：** 指定对象被下载时的网页的缓存过期时间。 **注意** 此参数不用于设置对象过期时间。 **约束限制：** 无 **取值范围：** 参见 HTTP 标准头域 Expires 的取值。 **默认取值：** 无
x-obs-website-redirect-location	String	否	**参数解释：** 当桶设置了 Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的 URL。 例如，重定向请求到桶内另一对象： x-obs-website-redirect-location:/anotherPage.html 或重定向请求到一个外部 URL： x-obs-website-redirect-location:http://www.example.com/ **约束限制：** 必须以“/”、“http://”或“https://”开头，长度不超过 2KB。

消息头名称	消息头类型	是否必选	描述
			取值范围： 无 **默认取值：** 无
x-obs-storage-class	String	否	**参数解释：** 指定对象的存储类型。 示例：x-obs-storage-class: STANDARD **约束限制：** 指定对象的存储类型时请注意大小写敏感。 **取值范围：** • STANDARD • WARM • COLD **默认取值：** 无
x-obs-meta-*	String	否	**参数解释：** 对象的自定义元数据。OBS 支持用户使用以“x-obs-meta-”开头的消息头来加入自定义的元数据，以便对对象进行自定义管理。当用户获取此对象或查询此对象元数据时，加入的自定义元数据将会在返回的消息头中出现。 示例：x-obs-meta-test: test metadata **约束限制：** • 所有自定义元数据大小的总和不超过8K。单个自定义元数据大小的计算方式为：每个键和值的 UTF-8 编码中的字节总数。 • 自定义元数据的 key 值不区分大小写，OBS 统一转为小写进行存储。value 值区分大小写。 • 自定义元数据 key-value 对都必须符合 US-ASCII。如果一定要使用非 ASCII 码或不可识别字符，需要客户端自行做编解码处理，可以采用 URL 编码或者 Base64 编码，服务端不会做解码处理。例如 x-obs-meta-中文：中文经 URL 编码后发送，“中文”的 URL 编码为：%E4%B8%AD%E6%96%87，则响应为 x-obs-meta-

消息头名称	消息头类型	是否必选	描述
			%E4%B8%AD%E6%96%87: %E4%B8%AD%E6%96%87 **取值范围：** 无 **默认取值：** 无

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Etag: etag
Last-Modified: time
```

响应消息头

表 5-142 附加响应消息头

消息头名称	消息头类型	描述
x-obs-metadata-directive	String	**参数解释：** 元数据操作指示符。 **取值范围：** • REPLACE_NEW：表示对于已经存在值的元数据进行替换，不存在值的元数据进行赋值，未指定的元数据保持不变（自定义头域作替换处理）。 • REPLACE：表示使用当前请求中携带的头域完整替换，未指定的元数据（本表中除 x-obs-storage-class 的其它元数据）会被删除。 **默认取值：** 无
Cache-Control	String	**参数解释：** 指定对象被下载时的网页的缓存行为。 **约束限制：** 如果请求携带了此头域，那么响应的消息中应该包含此消息头。

消息头名称	消息头类型	描述
		取值范围： 参见 HTTP 标准头域 Cache-control 的取值。 **默认取值：** 无
Content-Disposition	String	**参数解释：** 指定对象被下载时的名称。 **约束限制：** 如果请求携带了此头域，那么响应的消息中应该包含此消息头。 **取值范围：** 参见 HTTP 标准头域 Content-Disposition 的取值。 **默认取值：** 无
Content-Encoding	String	**参数解释：** 指定对象被下载时的内容编码格式。 **约束限制：** 如果请求携带了此头域，那么响应的消息中应该包含此消息头。 **取值范围：** 参见 HTTP 标准头域 Content-Encoding 的取值。 **默认取值：** 无
Content-Language	String	**参数解释：** 指定对象被下载时的内容语言格式。 **约束限制：** 如果请求携带了此头域，那么响应的消息中应该包含此消息头。 **取值范围：** 参见 HTTP 标准头域 Content-Language 的取值。 **默认取值：** 无
Expires	String	**参数解释：** 指定对象被下载时的网页的缓存过期时间。 **约束限制：**

消息头名称	消息头类型	描述
		如果请求携带了此头域，那么响应的消息中应该包含此消息头。 **取值范围：** 参见 HTTP 标准头域 Expires 的取值。 **默认取值：** 无
x-obs-website-redirect-location	String	**参数解释：** 当桶设置了 Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的 URL。 例如，重定向请求到桶内另一对象： x-obs-website-redirect-location:/anotherPage.html 或重定向请求到一个外部 URL： x-obs-website-redirect-location:http://www.example.com/ **约束限制：** • 如果请求携带了此头域，那么响应的消息中应该包含此消息头。 • 必须以“/”、“http://”或“https://”开头，长度不超过 2KB。 **取值范围：** 无 **默认取值：** 无
x-obs-storage-class	String	**参数解释：** 指定对象的存储类型。 **约束限制：** • 如果请求携带了此头域，那么响应的消息中应该包含此消息头。 • 指定对象的存储类型时请注意大小写敏感。 **取值范围：** • STANDARD • WARM • COLD **默认取值：** 无
x-obs-meta-*	String	**参数解释：** 对象的自定义元数据。OBS 支持用户使用以

消息头名称	消息头类型	描述
		“x-obs-meta-”开头的消息头来加入自定义的元数据，以便对对象进行自定义管理。当用户获取此对象或查询此对象元数据时，加入的自定义元数据将会在返回的消息头中出现。 **约束限制：** • 如果请求携带了此头域，那么响应的消息中应该包含此消息头。 • 所有自定义元数据大小的总和不超过 8K。单个自定义元数据大小的计算方式为：每个键和值的 UTF-8 编码中的字节总数。 • 自定义元数据的 key 值不区分大小写，OBS 统一转为小写进行存储。value 值区分大小写。 • 自定义元数据 key-value 对都必须符合 US-ASCII。如果一定要使用非 ASCII 码或不可识别字符，需要客户端自行做编解码处理，可以采用 URL 编码或者 Base64 编码，服务端不会做解码处理。例如 x-obs-meta-中文：中文经 URL 编码后发送，“中文”的 URL 编码为：%E4%B8%AD%E6%96%87，则响应为 x-obs-meta-%E4%B8%AD%E6%96%87: %E4%B8%AD%E6%96%87 **取值范围：** 无 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误；所有错误已经包含在表 6-2 中。

请求示例：添加对象元数据

给对象 object 添加元数据：Content-Type:application/zip 和 x-obs-meta-test:meta。

```
PUT /object?metadata HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 14:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

```
x-obs-metadata-directive:REPLACE_NEW
Content-Type:application/zip
x-obs-meta-test:meta
```

响应示例：添加对象元数据

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E4BB5905C41B6E65B6
Accept-Ranges: bytes
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSD3nAiTaBoeyt9oHp9vTYtXnLDmwV6D
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 0
x-obs-metadata-directive:REPLACE_NEW
x-obs-meta-test:meta
```

请求示例：修改对象元数据

对象 object 已存在元数据 x-obs-meta-test:testmeta，且 x-obs-storage-class 为 WARM，将对象 object 的元数据 x-obs-meta-test 修改为 newmeta，x-obs-storage-class 修改为 COLD。

```
PUT /object?metadata HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 14:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
x-obs-metadata-directive:REPLACE_NEW
x-obs-meta-test:newmeta
x-obs-storage-class:COLD
```

响应示例：修改对象元数据

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E4BB5905C41B6E65B6
Accept-Ranges: bytes
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSD3nAiTaBoeyt9oHp9vTYtXnLDmwV6D
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 0
x-obs-metadata-directive:REPLACE_NEW
x-obs-meta-test:newmeta
x-obs-storage-class:COLD
```

请求示例：删除对象元数据

对象 object 已存在元数据 x-obs-meta-test:newmeta，Content-Type:application/zip，删除元数据 x-obs-meta-test。

```
PUT /object?metadata HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 14:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

```
x-obs-metadata-directive:REPLACE
Content-Type:application/zip
```

响应示例：删除对象元数据

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E4BB5905C41B6E65B6
Accept-Ranges: bytes
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSD3nAiTaBoeyt9oHp9vTYtXnLDmwV6D
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 0
x-obs-metadata-directive:REPLACE
```

5.4.13 修改写对象

功能介绍

修改写对象操作是指将指定文件桶内的一个对象从指定位置起修改为其他内容。

说明

目前接口仅在并行文件系统支持，创建并行文件系统的方法详见请求示例：创建并行文件系统。

请求消息样式

```
PUT /ObjectName?modify&position=Position HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Type: type
Content-Length: length
Authorization: authorization
Date: date
<object Content>
```

请求消息参数

该请求需要在消息中指定参数，表明这是修改写上传，同时指定本次修改上传位置。参数说明如表 5-143 所示。

表 5-143 请求消息参数

参数名称	是否必选	参数类型	描述
modify	是	String	**参数解释：** 携带 modify 参数表示该请求以修改写方式上传。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：**

参数名称	是否必选	参数类型	描述
			不涉及
position	是	Integer	**参数解释：** 用于指定从某个位置进行修改写操作（从第一个字符开始数）。单位：字节。 **约束限制：** 不涉及 **取值范围：** 大于或等于 0 的整数，且修改写后文件的长度不超过 54975581388800（50TB）。 **默认取值：** 不涉及

请求消息头

该请求使用公共的请求消息头，具体如表 3-3 所示。

请求消息元素

该请求消息头中不带消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: Date
ETag: etag
Content-Length: length
Server: OBS
x-obs-request-id: request-id
x-obs-id-2: id
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /ObjectName?modify&position=Position HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: Wed, 08 Jul 2015 06:57:01 GMT
Content-Type: image/jpg
Content-Length: 1458
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:kZoYNv66bsmc10+dcGKw5x2PRrk=

[1458 bytes of object data]
```

响应示例

```
HTTP/1.1 200
Date: Wed, 08 Jul 2015 06:57:02 GMT
ETag: "d41d8cd98f00b204e9800998ecf8427e"
Content-Length: 0
Server: OBS
x-obs-request-id: 8DF400000163D3F0FD2A03D2D30B0542
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCTjCqTmsA1XRpIrmrJdvcEWvZyjbztd
```

5.4.14 截断对象

功能介绍

截断对象操作是指将指定文件桶内的一个已有对象截断到指定大小。

说明

目前接口仅在并行文件系统支持，创建并行文件系统的方法详见请求示例：创建并行文件系统。

请求消息样式

```
PUT /ObjectName?truncate&length=Length HTTP/1.1
Host: bucketname.obs.region.example.com
Authorization: authorization
Content-Length: length
Date: date
```

请求消息参数

该请求需要在消息中指定参数，表明这是截断写上传，同时指定本次截断后的对象大小。参数说明如表 5-144 所示。

表 5-144 请求消息参数

参数名称	是否必选	参数类型	描述
truncate	是	String	**参数解释：** 携带 truncate 参数表示该请求以截断方式上传。 **约束限制：**

参数名称	是否必选	参数类型	描述
			不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及
length	是	Integer	**参数解释：** 截断后对象的大小。单位：字节。 **约束限制：** 不涉及 **取值范围：** [0-54975581388800（50TB）] **默认取值：** 不涉及

请求消息头

该请求使用公共的请求消息头，具体如表 3-3 所示。

请求消息元素

该请求消息头中不带消息元素。

响应消息样式

```
HTTP/1.1 204 status_code
Server: OBS
x-obs-request-id: request-id
x-obs-id-2: id
Date: Date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
PUT /ObjectName?truncate&length=1000 HTTP/1.1
Host: examplebucket.obs.region.example.com
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 1
Date: WED, 01 Jul 2015 04:19:20 GMT
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 8DF400000163D3F51DEA05AC9CA066F1
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSgkM4Dij80gAeFY8pAZIwx72QhDeBZ5
Date: WED, 01 Jul 2015 04:19:21 GMT
```

5.4.15 重命名对象

功能介绍

重命名对象操作是指将指定文件桶内的一个对象重命名为其他对象名。

说明

目前接口仅在并行文件系统支持，创建并行文件系统的方法详见请求示例：创建并行文件系统。
重命名对象操作为非幂等操作。

请求消息样式

```
POST /ObjectName?name=Name&rename HTTP/1.1
Host: bucketname.obs.region.example.com
Authorization: authorization
Date: date
```

请求消息参数

该请求需要在消息中指定参数，表明这是重命名操作，同时指定本次重命名后的对象名称。参数说明如表 5-145 所示。

表 5-145 请求消息参数

参数名称	是否必选	参数类型	描述
name	是	String	**参数解释：** 重命名后的对象名称。 **约束限制：** 需要使用绝对路径。例如：name=/dir1/file1。 **取值范围：** 不涉及 **默认取值：**

参数名称	是否必选	参数类型	描述
			不涉及
rename	是	String	**参数解释：** 携带 rename 参数表示该请求是对对象进行重命名的操作。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及

请求消息头

该请求使用公共的请求消息头，具体如表 3-3 所示。

请求消息元素

该请求消息头中不带消息元素。

响应消息样式

```
HTTP/1.1 204 status_code
Server: OBS
x-obs-request-id: request-id
x-obs-id-2: id
Date: Date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

无特殊错误，所有错误已经包含在表 6-2 中。

请求示例

```
POST /ObjectName?name=file2&rename HTTP/1.1
Host: examplebucket.obs.region.example.com
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Date: WED, 01 Jul 2015 04:19:20 GMT
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 8DF400000163D3F51DEA05AC9CA066F1
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSgkM4Dij80gAeFY8pAZIwx72QhDeBZ5
Date: WED, 01 Jul 2015 04:19:21 GMT
```

5.4.16 配置对象级 WORM 保护策略

功能介绍

开启了 WORM 开关的桶，上传的对象支持配置或修改对象保护期限。

- 如果上传对象时没有配置保护期限或自动应用桶级默认保护策略，您可以通过该操作配置对象保护期限。
- 如果上传对象时配置了保护期限或自动应用了默认保护期限，允许用户通过该操作延长保护期限。
- 对象保护期限仅允许修改，不允许删除。

说明

用户需要拥有“PutObjectRetention”权限才能配置或修改对象保护期限。

多版本

开启了 WORM 开关的桶默认开启了多版本，因此桶内对象在上传时会具备版本号。您在配置对象级 WORM 保护策略时可以指定版本号来为特定版本的对象配置，如果您不指定版本号，则改动只会对同名对象的最新版本生效。WORM 功能不会对带唯一版本号的删除标记生效。

多段操作

多段上传的对象在合并前不会自动应用桶级默认 WORM 策略，也无法通过在上传或合并时指定头域来配置对象级 WORM 保护策略，指定已上传的段作为此接口的目标对象也无法进行配置。如果您需要对多段对象进行保护，您可以在合并多段对象后通过此接口为其配置对象级 WORM 保护策略。

请求消息样式

```
PUT /ObjectName?retention&versionId=versionid HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization

<Retention>
   <Mode>String</Mode>
   <RetainUntilDate>Timestamp</RetainUntilDate>
</Retention>
```

请求消息参数

请求参数说明如表 5-146 所示。

表 5-146 请求消息参数

参数名称	是否必选	参数类型	描述
retention	是	String	**参数解释：** 表示这是配置或修改对象保护期限操作。 **约束限制：** 不涉及 **取值范围：** 不涉及 **默认取值：** 不涉及
versionId	否	String	**参数解释：** 对象的版本号。表示更改指定版本对象的 WORM 策略。不携带 versionId 参数则为最新版本。 **约束限制：** 不涉及 **取值范围：** 长度为 32 的字符串。 **默认取值：** 不涉及，如果不设置则默认修改最新版本的对象。

请求消息头

该请求使用公共请求消息头，具体参见表 3-3。

请求消息元素

表 5-147 请求消息元素

参数名称	是否必选	参数类型	描述
Retention	是	Container	**参数解释：** 对象级 WORM 保护策略配置的容器，Retention 是 Mode、RetainUntilDate 的父节点。 **约束限制：**

参数名称	是否必选	参数类型	描述
			不涉及 **取值范围：** 请详见表 5-148。 **默认取值：** 不涉及

表 5-148 Retention 参数说明

参数名称	是否必选	参数类型	描述
Mode	是	String	**参数解释：** 对象的保护策略。 **约束限制：** 不涉及 **取值范围：** COMPLIANCE：合规模式 **默认取值：** 不涉及
RetainUntilDate	是	Long	**参数解释：** 对象的保护期限，时间戳格式，精确到毫秒级，如 2015 年 7 月 1 日 13 点 20 分 35 秒对应的值为 1435728035000。 示例：1435728035000 **约束限制：** 该字段必须晚于当前时间，且仅可延长不能缩短。 **取值范围：** 不涉及 **默认取值：** 不涉及

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息不带消息元素。

错误响应消息

此请求可能的特殊错误如下表 5-149 描述。

表 5-149 错误响应消息

错误码	描述	HTTP 状态码
InvalidRequest	目标桶没有开启桶级 WORM 开关	400
InvalidRequest	保护期限设置错误	400
MalformedObjectLockError	策略配置格式错误	400

其余错误已经包含在表 6-2 中。

请求示例

```
PUT /objectname?retention HTTP/1.1
Host: bucketname.obs.region.example.com
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Type: application/xml
Content-Length: 157
<Retention>
   <Mode>COMPLIANCE</Mode>
   <RetainUntilDate>1435728035000</RetainUntilDate>
</Retention>
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

5.5 多段操作

5.5.1 列举桶中已初始化多段任务

功能介绍

用户可以通过本接口查询一个桶中所有的初始化后还未合并以及未取消的多段上传任务。

请求消息样式

```
GET /?uploads&max-uploads=max HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求可以通过在请求消息中指定参数，查询指定范围的多段上传任务，请求参数说明如表 5-150 所示。

表 5-150 请求消息参数

参数名称	是否必选	参数类型	描述
delimiter	否	String	**参数解释：** 对于名字中包含 delimiter 的对象的任务，其对象名（如果请求中指定了 prefix，则此处的对象名需要去掉 prefix）中从首字符至第一个 delimiter 之间的字符串将作为 CommonPrefix 在响应中返回。对象名包含 CommonPrefix 的任务被视为一个分组，作为一条记录在响应中返回，该记录不包含任务的信息，仅用于提示用户该分组下存在多段上传任务。 **约束限制：** 不涉及 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 不涉及
prefix	否	String	**参数解释：** 列举桶内多段任务列表时，指定一个前缀，限定返回的多段任务的对象名必须带有 prefix 前缀。如果请求中指定了 prefix，则响应中仅包

参数名称	是否必选	参数类型	描述
			含对象名以 prefix 开始的任务信息。 **约束限制：** 满足对象名的格式。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。 **默认取值：** 不涉及
max-uploads	否	Integer	**参数解释：** 单次列举的多段任务的最大条目。单位：个。 **约束限制：** 不涉及 **取值范围：** [1,1000]，当超出范围时，按照默认的 1000 进行处理。 **默认取值：** 1000
key-marker	否	String	**参数解释：** 列举时返回指定的 key-marker 之后的多段任务。 **约束限制：** 不涉及 **取值范围：** 上次请求返回体的 NextKeyMarker 值。 **默认取值：** 不涉及
upload-id-marker	否	String	**参数解释：** 列举时返回指定的 key-marker 的 upload-id-marker 之后的多段任务。 **约束限制：** 需要和 key-marker 一起使用才有意义。 **取值范围：** 上次请求返回体的 NextUploadIdMarker 值。 **默认取值：** 不涉及

请求消息头

该请求使用公共请求消息头，具体请见表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListMultipartUploadsResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Bucket>bucketname</Bucket>
   <KeyMarker/>
   <UploadIdMarker/>
   <NextKeyMarker>nextMarker</NextKeyMarker>
   <NextUploadIdMarker>idMarker</NextUploadIdMarker>
   <MaxUploads>maxUploads</MaxUploads>
   <IsTruncated>true</IsTruncated>
   <Upload>
      <Key>key</Key>
      <UploadId>uploadID</UploadId>
      <Initiator>
         <ID>domainID/domainID:userID/userID</ID>
      </Initiator>
      <Owner>
         <ID>ownerID</ID>
      </Owner>
      <StorageClass>storageclass</StorageClass>
      <Initiated>initiatedDate</Initiated>
   </Upload>
</ListMultipartUploadsResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中通过消息元素返回多段上传任务，元素的具体意义如表 5-151 所示。

表 5-151 响应消息元素

参数名称	参数类型	描述
ListMultipartUploadsResult	Container	**参数解释：** 保存列举多段上传请求结果的容器，ListMultipartUploadsResult 是 Bucket、

参数名称	参数类型	描述
		KeyMarker、UploadIdMarker、NextKeyMarker、NextUploadIdMarker、MaxUploads、IsTruncated、Upload、Prefix、Delimiter、CommonPrefixes 的父节点。 **取值范围：** 请详见表 5-152。

表 5-152 ListMultipartUploadsResult 响应消息元素说明

参数名称	参数类型	描述
Bucket	String	**参数解释：** 初始化任务所在的桶名。 **取值范围：** 长度为 3~63 的字符串。
KeyMarker	String	**参数解释：** 列举时的起始对象位置。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。
UploadIdMarker	String	**参数解释：** 列举时的起始 UploadId 位置。 **取值范围：** 长度大于 0 且不超过 32 的字符串。
NextKeyMarker	String	**参数解释：** 如果本次没有返回全部结果，响应请求中将包含 NextKeyMarker 字段，用于标明接下来请求的 KeyMarker 值。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。
NextUploadIdMarker	String	**参数解释：** 如果本次没有返回全部结果，响应请求中将包含 NextUploadIdMarker 元素，用于标明接下来请求的 UploadMarker 值。 **取值范围：** 长度大于 0 且不超过 32 的字符串。
MaxUploads	Integer	**参数解释：** 单次返回的最大多段上传任务数目。单位：个。

参数名称	参数类型	描述
		取值范围： [1，1000]，当超出范围时默认为 1000。
IsTruncated	Boolean	**参数解释：** 表明是否本次返回的分段上传结果列表被截断。 **取值范围：** • true：表示本次没有返回全部结果 • false：表示本次已经返回了全部结果。
Upload	Container	**参数解释：** 保存分段上传任务信息的容器，Upload 是 Key、UploadId、Initiator、Owner、StorageClass、Initiated 的父节点。 **取值范围：** 请详见表 5-153。
ListMultipartUploadsResult.Prefix	String	**参数解释：** 请求中携带的 prefix。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。
Delimiter	String	**参数解释：** 请求中带的 Delimiter。 **取值范围：** 长度大于且不超过 1024 的字符串。
CommonPrefixes	Container	**参数解释：** 请求中带 Delimiter 参数时，返回消息带 CommonPrefixes 分组信息，CommonPrefixes 是 Prefix 的父节点。 **取值范围：** 请详见表 5-156。

表 5-153 Upload 响应消息元素说明

参数名称	参数类型	描述
Key	String	**参数解释：** 初始化多段上传任务的对象名称。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。

参数名称	参数类型	描述
UploadId	String	**参数解释：** 多段上传任务的 ID。 **取值范围：** 长度大于 0 且不超过 32 的字符串。
Initiator	Container	**参数解释：** 多段上传任务的创建者，Initiator 是 ID 的父节点。 **取值范围：** 请详见表 5-154。
Owner	Container	**参数解释：** 段的所有者，Owner 是 ID 的父节点。 **取值范围：** 请详见表 5-155。
StorageClass	String	**参数解释：** 表明待多段上传的对象存储类型。 **取值范围：** • STANDARD：标准存储 • WARM：低频访问存储 • COLD：归档存储
Initiated	Date	**参数解释：** 多段上传任务的初始化时间（UTC 时间）。日期格式为 ISO8601 的格式，例如：2018-01-01T00:00:00.000Z。 **取值范围：** 不涉及

表 5-154 Initiator 响应消息元素说明

参数名称	参数类型	描述
ID	String	**参数解释：** 多段任务创建者的 urn。格式为 domainID/{domainId}:userID/{userId}。 **取值范围：** 不涉及

表 5-155 Owner 响应消息元素说明

参数名称	参数类型	描述
ID	String	**参数解释：** 多段任务创建者的 DomainId，格式为{domainId}。 **取值范围：** 不涉及

表 5-156 CommonPrefixes 响应消息元素说明

参数名称	参数类型	描述
CommonPrefixes.Prefix	String	**参数解释：** CommonPrefixes 分组信息中，表明不同的前缀。 **取值范围：** 不涉及

错误响应消息

OBS 系统对 maxUploads 进行判断，如果 maxUploads 不为整数类型或者小于 0，OBS 返回 400 Bad Request。

其他错误已经包含在表 6-2 中。

请求示例：不带任何参数列举已初始化的段任务

```
GET /?uploads HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:51:21 GMT
Authorization: OBS UDSIAMSTUBTEST000008:XdmZgYQ+ZVy1rjxJ9/KpKq+wrU0=
```

响应示例：不带任何参数列举已初始化的段任务

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D405534D046A2295674C
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSDaHP+a+Bp0RI6Mm9XvCOrf7q3qvBQW
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:51:21 GMT
Content-Length: 681

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListMultipartUploadsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Bucket>examplebucket</Bucket>
  <KeyMarker/>
  <UploadIdMarker/>
```

```
  <Delimiter/>
  <Prefix/>
  <MaxUploads>1000</MaxUploads>
  <IsTruncated>false</IsTruncated>
  <Upload>
    <Key>obj2</Key>
    <UploadId>00000163D40171ED8DF4050919BD02B8</UploadId>
    <Initiator>

<ID>domainID/b4bf1b36d9ca43d984fbcb9491b6fce9:userID/71f390117351534r88115ea2c26d19
99</ID>
    </Initiator>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <Initiated>2015-07-01T02:30:54.582Z</Initiated>
  </Upload>
</ListMultipartUploadsResult>
```

请求示例：带 *prefix* 和 *delimeter* 列举已初始化的段任务

例如，用户桶 examplebucket 中 2 个段任务，对象名分别为 multipart-object001 和 part2-key02，列举段任务时，设置 prefix 为“multipart"，delimeter 设置为 object001，列举已初始化的段任务。

```
GET /?uploads&delimiter=object001&prefix=multipart HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:51:21 GMT
Authorization: OBS UDSIAMSTUBTEST000008:XdmZgYQ+ZVy1rjxJ9/KpKq+wrU0=
```

响应示例：带 *prefix* 和 *delimeter* 列举已初始化的段任务

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 5DEB00000164A27A1610B8250790D703
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSq3ls2ZtLDD6pQLcJq1yGITXgspSvBR
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:51:21 GMT
Content-Length: 681
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListMultipartUploadsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Bucket>newbucket0001</Bucket>
  <KeyMarker></KeyMarker>
  <UploadIdMarker>
  </UploadIdMarker>
  <Delimiter>object</Delimiter>
  <Prefix>multipart</Prefix>
  <MaxUploads>1000</MaxUploads>
  <IsTruncated>false</IsTruncated>
  <CommonPrefixes>
    <Prefix>multipart-object001</Prefix>
```

```
  </CommonPrefixes>
</ListMultipartUploadsResult>
```

5.5.2 初始化上传段任务

功能介绍

使用多段上传特性时，用户必须首先调用创建多段上传任务接口创建任务，系统会给用户返回一个全局唯一的多段上传任务号，作为任务标识。后续用户可以根据这个标识发起相关的请求，如：上传段、合并段、列举段等。创建多段上传任务不影响已有的同名对象；同一个对象可以同时存在多个多段上传任务；每个多段上传任务在初始化时可以附加消息头信息，包括 acl、用户自定义元数据和通用的 HTTP 消息头 contentType、contentEncoding 等，这些附加的消息头信息将先记录在多段上传任务元数据中。

该操作支持服务端加密功能。

WORM

如果桶的 WORM 开关是开启的，则可以在初始化多段任务时配置对象级 WORM 保护策略。您可以通过携带头域 x-obs-object-lock-mode 和 x-obs-object-lock-retain-until-date 在初始化多段任务的同时指定最终合并对象的保护策略，如果您不携带这些头域，但配置了桶级默认 WORM 策略，则合并后的对象会自动应用默认策略。您还可以在合并后配置或修改对象级 WORM 保护策略。

与使用 PUT 和 POST 方法上传对象不同，多段上传对 x-obs-object-lock-retain-until-date 头域的约束只需要您提供不晚于初始化时间的日期，而无需晚于合并多段任务的时间。应用桶级默认 WORM 策略时，保护的起始时间从合并时间开始计算，即合并时间+桶级默认保护期限。在合并前，多段上传任务可以被取消，不会受到 WORM 功能影响。

请求消息样式

```
POST /ObjectName?uploads  HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

请求消息参数

该请求需要在消息中指定参数，表明这是多段上传，参数意义如表 5-157 所示。

表 5-157 请求消息参数

参数名称	参数类型	是否必选	描述
uploads	String	是	**参数解释：** 表明这是多段上传任务。 **约束限制：** • 该参数为空字符串。

参数名称	参数类型	是否必选	描述
			• 如果请求未设置此参数，则为普通 POST 上传任务。 **取值范围：** 取值为空。 **默认取值：** 无

请求消息头

该请求可以使用附加的消息头，具体如表 5-158 所示。

表 5-158 请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-acl	String	否	**参数解释：** 初始化多段上传任务时，可以加上此消息头设置对象的权限控制策略。 示例：x-obs-acl:public-read-write。 **约束限制：** 字符串形式的预定义策略。 **取值范围：** • private • public-read • public-read-write **默认取值：** private
x-obs-grant-read	String	否	**参数解释：** 初始化多段上传任务时，使用此头域授权读对象和获取对象元数据的权限给 domain 下的所有用户。 示例：x-obs-grant-read: ID=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 无 **默认取值：** 无

消息头名称	消息头类型	是否必选	描述
x-obs-grant-read-acp	String	否	**参数解释：** 初始化多段上传任务时，使用此头域授权获取对象 ACL 的权限给 domain 下的所有用户。 示例：x-obs-grant-read-acp: ID=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 无 **默认取值：** 无
x-obs-grant-write-acp	String	否	**参数解释：** 初始化多段上传任务时，使用此头域授权写对象 ACL 的权限给 domain 下的所有用户。 示例：x-obs-grant-write-acp: ID=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：** 无 **默认取值：** 无
x-obs-grant-full-control	String	否	**参数解释：** 初始化多段上传任务时，使用此头域授权以下权限给 domain 下的所有用户： 读对象、获取对象元数据、获取对象 ACL、写对象 ACL 的权限。 示例：x-obs-grant-full-control: ID=domainID。 **约束限制：** 如果授权给多个租户，需要通过“,”分隔。 **取值范围：**

消息头名称	消息头类型	是否必选	描述
			无 **默认取值：** 无
x-obs-storage-class	String	否	**参数解释：** 初始化多段上传任务时，可以加上此头域设置对象的存储类型。 示例：x-obs-storage-class: STANDARD **约束限制：** • 如果未设置此头域，则以桶的默认存储类型作为对象的存储类型。 • 设置该参数的值时请注意大小写敏感。 **取值范围：** • STANDARD • WARM • COLD **默认取值：** 继承桶的存储类型
x-obs-website-redirect-location	String	否	**参数解释：** 当桶设置了 Website 配置，可以将获取这个对象的请求重定向到桶内另一个对象或一个外部的 URL。 例如，重定向请求到桶内另一对象： x-obs-website-redirect-location:/anotherPage.html 或重定向请求到一个外部 URL： x-obs-website-redirect-location:http://www.example.com/ OBS 将这个值从头域中取出，保存在对象的元数据“WebsiteRedirectLocation”中。 **约束限制：** • 必须以“/”、“http://”或“https://”开头，长度不超过 2KB。 • OBS 仅支持为桶根目录下的对象设置重定向，不支持为桶中文件夹下的对象设置重定向。 **默认取值：** 无
x-obs-server-	String	否。当使	**参数解释：**

消息头名称	消息头类型	是否必选	描述
side-encryption		用 SSE-KMS 方式时，必选。	使用该头域表示服务端加密是 SSE-KMS 方式。 示例：x-obs-server-side-encryption: kms **约束限制：** 无 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	否	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-	String	否。当使	**参数解释：**

消息头名称	消息头类型	是否必选	描述
side-encryption-customer-key		用 SSE-C 方式时，必选。	该头域表示加密使用的密钥，该密钥用于加密对象。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选。	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无
x-obs-expires	Integer	否	**参数解释：** 表示对象的过期时间，单位是天。过期之后对象会被自动删除。（从对象最后修改时间开始计算） 示例：x-obs-expires:3

消息头名称	消息头类型	是否必选	描述
			约束限制： 此字段对于每个对象仅支持上传时配置，不支持后期通过修改元数据接口修改。 **取值范围：** 大于等于 0 的整型数，单位：天。 **默认取值：** 无
x-obs-object-lock-mode	String	否，携带 x-obs-object-lock-retain-until-date 头域时必带	**参数解释：** 要应用于对象的 WORM 模式。 示例：x-obs-object-lock-mode:COMPLIANCE **约束限制：** 该参数需要和 x-obs-object-lock-retain-until-date 一同使用。 **取值范围：** 仅支持 COMPLIANCE，即合规模式。 **默认取值：** 无
x-obs-object-lock-retain-until-date	String	否，携带 x-obs-object-lock-mode 头域时必带	**参数解释：** 对象的 WORM 策略过期的截止时间。 示例：x-obs-object-lock-retain-until-date:2015-07-01T04:11:15.297Z **约束限制：** • 格式要求为 UTC 时间，并符合 ISO 8601 标准。如：2015-07-01T04:11:15.297Z。 • 该参数需要和 x-obs-object-lock-mode 一同使用。 **取值范围：** 需要大于当前时间。 **默认取值：** 无
x-obs-meta-*	String	否	**参数解释：** 初始化上传段任务时，可以在 HTTP 请求中加入以“x-obs-meta-”开头的消息头，用来加入自定义的元数据，以便对对象进行自定义管理。当用户获取此对象或查询此对象元数据时，加入的自定义元数据将

消息头名称	消息头类型	是否必选	描述
			会在返回消息的头中出现。 示例：x-obs-meta-test: test metadata **约束限制：** HTTP 请求不包含消息体，长度不能超过8KB。 **取值范围：** 无 **默认取值：** 无

其他公共消息头请参考表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Connection: status

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Bucket>BucketName</Bucket>
   <Key>ObjectName</Key>
   <UploadId>uploadID</UploadId>
</InitiateMultipartUploadResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

表 5-159 附加响应消息头

消息头名称	消息头类型	描述
x-obs-server-side-encryption	String	**参数解释：** 该头域表示服务端的加密方式。 示例：x-obs-server-side-encryption: kms **约束限制：** 如果服务端加密是 SSE-KMS 方式，响应包含该头域。

消息头名称	消息头类型	描述
		取值范围： • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的 KMS 主密钥所属的项目 ID（非企业项目 ID）。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。

消息头名称	消息头类型	描述
		示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无

响应消息元素

该请求响应消息中通过返回消息元素，返回本次多段上传任务的多段上传任务号、桶名、对象名，供后续上传段、合并段使用，元素的具体意义如表 5-160 所示。

表 5-160 响应消息元素

元素名称	元素类型	描述
InitiateMultipartUploadResult	XML	**参数解释：** 描述多段上传任务的容器。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
Bucket	String	**参数解释：** 多段上传任务的桶名。 **约束限制：** • 桶的名字需全局唯一，不能与已有的任何桶名称重复，包括其他用户创建的桶。 • 桶命名规则如下： – 3～63 个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。 – 禁止使用 IP 地址。 – 禁止以“-”或“.”开头及结尾。 – 禁止两个“.”相邻（如：“my..bucket”）。 – 禁止“.”和“-”相邻（如：“my-

元素名称	元素类型	描述
		.bucket”和“my.-bucket”）。 • 同一用户在同一个区域多次创建同名桶不会报错，创建的桶属性以第一次请求为准。 **取值范围：** 无 **默认取值：** 无
Key	String	**参数解释：** 多段上传任务的对象名。对象名是对象在存储桶中的唯一标识。对象名是对象在桶中的完整路径，路径中不包含桶名。 **约束限制：** 无。 **取值范围：** 长度大于 0 且不超过 1024 的字符。 **默认取值：** 无
UploadId	String	**参数解释：** 多段上传任务的 ID，后面进行多段上传时，可利用这个 ID 指定多段上传任务。 **约束限制：** 无 **取值范围：** 长度大于 0 且不超过 32 的字符串。 **默认取值：** 无

错误响应消息

1. 如果 AccessKey 或签名无效，OBS 返回 403 Forbidden，错误码为 AccessDenied。
2. 如果桶不存在，OBS 返回 404 Not Found，错误码为 NoSuchBucket。
3. 检查用户是否具有指定桶的写权限，如果没有权限，OBS 返回 403 Forbidden，错误码为 AccessDenied。

其他错误已包含在表 6-2 中。

请求示例：初始化段

```
POST /objectkey?uploads  HTTP/1.1
Host: examplebucket.obs.region.example.com
```

```
Date: WED, 01 Jul 2015 05:14:52 GMT
Authorization: OBS AKIAIOSFODNN7EXAMPLE:VGhpcyBtZXNzYWdlIHNpZ25lZGGieSRlbHZpbmc=
```

响应示例：初始化段

```
HTTP/1.1 200 OK
Server: OBS
x-obs-id-2: Weag1LuByRx9e6j5Onimru9pO4ZVKnJ2Qz7/C1NPcfTWAtRPfTaOFg==
x-obs-request-id: 996c76696e6727732072657175657374
Date: WED, 01 Jul 2015 05:14:52 GMT
Content-Length: 303

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-
30/">
  <Bucket>bucketname</Bucket>
  <Key>objectkey</Key>
  <UploadId>DCD2FC98B4F70000013DF578ACA318E7</UploadId>
</InitiateMultipartUploadResult>
```

请求示例：初始化段的同时携带 ACL

```
POST /objectkey?uploads  HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 05:15:43 GMT
x-obs-grant-write-
acp:ID=52f24s3593as5730ea4f722483579ai7,ID=a93fcas852f24s3596ea8366794f7224
Authorization: OBS AKIAIOSFODNN7EXAMPLE:VGhpcyBtZXNzYWdlIHNpZ25lZGGieSRlbHZpbmc=
```

响应示例：初始化段的同时携带 ACL

```
HTTP/1.1 200 OK
Server: OBS
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTnv+daB51p+IVhAvWN7s5rSKhcWqDFs
x-obs-request-id: BB78000001648457112DF37FDFADD7AD
Date: WED, 01 Jul 2015 05:15:43 GMT
Content-Length: 303

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-
30/">
  <Bucket>bucketname</Bucket>
  <Key>objectkey</Key>
  <UploadId>000001648453845DBB78F2340DD460D8</UploadId>
</InitiateMultipartUploadResult>
```

5.5.3 上传段

功能介绍

多段上传任务创建后，用户可以通过指定多段上传任务号，通过上传段接口为特定的任务上传段，从客户端上传新数据。同一个对象的同一个多段上传任务在上传段时，上传的顺序对后续的合并操作没有影响，也即支持多个段并发上传。

请确保段大小范围是[100KB，5GB]，最后一个段的大小范围为[0,5GB]，否则在进行合并段操作时会报错。上传的段的编号也有范围限制，其范围是[1,10000]。

该操作支持服务端加密功能。

须知

段任务中的 partNumber 是唯一的，重复上传相同 partNumber 的段，后一次上传会覆盖前一次上传内容。多并发上传同一对象的同一 partNumber 时，服务端遵循 Last Write Win 策略，但“Last Write”的时间定义为段元数据创建时间。为了保证数据准确性，客户端需要加锁保证同一对象的同一个段上传的并发性。同一对象的不同段并发上传不需要加锁。

请求消息样式

```
PUT /ObjectName?partNumber=partNum&uploadId=uploadID  HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Length: length
Authorization: authorization
Content-MD5: md5
<object Content>
```

请求消息参数

在上传段的时候需要通过在消息参数中指定多段上传任务号和段号来上传指定段，参数的具体意义如表 5-161 所示。

表 5-161 请求消息参数

参数名称	参数类型	是否必选	描述
partNumber	Integer	是	**参数解释：** 上传段的段号。 **约束限制：** 无 **取值范围：** 从 1 到 10000 的整数。 **默认取值：** 无
uploadId	String	是	**参数解释：** 多段上传任务 Id。 **约束限制：** 无 **取值范围：** 无

参数名称	参数类型	是否必选	描述
			默认取值： 无

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

表 5-162 请求消息头

消息头名称	消息头类型	是否必选	描述
x-obs-server-side-encryption-customer-algorithm	String	否。当使用 SSE-C 方式时，必选，需要与初始化上传段任务使用相同的算法。	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** • 在 SSE-C 加密方式下使用该头域，该头域表示加密使用的算法。 • 需要和 x-obs-server-side-encryption-customer-key， x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key	String	否。当使用 SSE-C 方式时，必选，需要与初始化上传段任务使用相同的密钥。	**参数解释：** 该头域表示加密使用的密钥，该密钥用于加密对象。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** • 仅在 SSE-C 加密方式下使用该头域。 • 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key-MD5 一起使用。

消息头名称	消息头类型	是否必选	描述
			取值范围： 无 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	否。当使用 SSE-C 方式时，必选，需要与初始化上传段任务使用相同的 MD5 值。	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。MD5 值用于验证密钥传输过程中没有出错。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 仅在 SSE-C 加密方式下使用该头域。 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm，x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
ETag: etag
Content-Length: length
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

表 5-163 附加响应消息头

消息头名称	消息头类型	描述
x-obs-server-side-	String	**参数解释：**

消息头名称	消息头类型	描述
encryption		该头域表示服务端的加密方式。 示例：x-obs-server-side-encryption：kms **约束限制：** 如果服务端加密是 SSE-KMS 方式，响应包含该头域。 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的 KMS 主密钥所属的项目 ID（非企业项目 ID）。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应

消息头名称	消息头类型	描述
		包含该头域。 **取值范围：** AES256 **默认取值：** 无
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 该头域表示加密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** 密钥 ID MD5 的 base64 值。 **默认取值：** 无

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息说明

1. 如果段序号超过范围[1,10000]，则返回错误 400 Bad Request。
2. 如果段大小超过 5G，则返回错误 400 Bad Request。
3. 如果 AccessKey 或签名无效，OBS 返回 403 Forbidden， 错误码为 AccessDenied。
4. 查询桶是否存在，如果桶不存在，OBS 返回 404 Not Found，错误码为 NoSuchBucket。
5. 检查桶的 ACL，判断用户 DomainId 是否具有指定桶的写权限，如果没有权限，则 OBS 返回 403 Forbidden，错误码为 AccessDenied。
6. 检查多段上传任务是否存在，如果不存在，OBS 返回 404 Not Found，错误码为 NoSuchUpload。
7. 检查请求用户是否是多段上传任务的发起者（Initiator），如果不是，OBS 返回 403 Forbidden， 错误码为 AccessDenied。

其他错误已包含在表 6-2 中。

请求示例

```
PUT /object02?partNumber=1&uploadId=00000163D40171ED8DF4050919BD02B8 HTTP/1.1
User-Agent: curl/7.29.0
```

```
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:15:55 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:ZB0hFwaHubi1aKHv7dSZjJts40g=
Content-Length: 102015348

[102015348 Byte part content]
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D40956A703289CA066F1
ETag: "b026324c6904b2a9cb4b88d6d61c81d1"
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCUQu/EOEVSMa04GXVwy0z9WI+BsDKvfh
Date: WED, 01 Jul 2015 05:15:55 GMT
Content-Length: 0
```

5.5.4 拷贝段

功能介绍

多段上传任务创建后，用户可以通过指定多段上传任务号，为特定的任务上传段。添加段的方式还包括调用段拷贝接口。允许用户将已上传对象的一部分或全部拷贝为段。

该操作支持服务端加密功能。

须知

拷贝段的结果不能仅根据 HTTP 返回头域中的 status_code 来判断请求是否成功，头域中 status_code 返回 200 时表示服务端已经收到请求，且开始处理拷贝段请求。拷贝是否成功会在响应消息的 body 中，只有 body 体中有 ETag 标签才表示成功，否则表示拷贝失败。

将源对象 object 拷贝为一个段 part1，如果在拷贝操作之前 part1 已经存在，拷贝操作执行之后老段数据 part1 会被新拷贝的段数据覆盖。拷贝成功后，只能列举到最新的段 part1，老段数据将会被删除。因此在使用拷贝段接口时请确保目标段不存在或者已无价值，避免因拷贝段导致数据误删除。拷贝过程中源对象 object 无任何变化。

归档存储对象

如果源对象是归档存储对象，需要判断源对象的恢复状态，只有当源对象处于已恢复状态时，才能拷贝成功。源对象未恢复或者正在恢复时，会拷贝失败，返回错误 403 Forbidden。异常描述为：

ErrorCode: InvalidObjectState

ErrorMessage: Operation is not valid for the source object's storage class

请求消息样式

```
PUT /ObjectName?partNumber=partNum&uploadId=UploadID HTTP/1.1
Host: bucketname.obs.region.example.com
```

```
Date: date
x-obs-copy-source: sourceobject
x-obs-copy-source-range:bytes=start-end
Authorization: authorization
Content-Length: length
```

请求消息参数

拷贝段需要在参数中指定目标段的段号和多段上传任务号，参数的具体意义如表 5-164 所示。

表 5-164 请求消息参数

参数名称	是否必选	参数类型	描述
partNumber	是	Integer	**参数解释：** 上传段的段号。 **约束限制：** 不涉及 **取值范围：** 从 1 到 10000 的整数。 **默认取值：** 不涉及
uploadId	是	String	**参数解释：** 多段上传任务 ID。 **约束限制：** 不涉及 **取值范围：** 长度大于 0 且不超过 32 的字符串。 **默认取值：** 不涉及

请求消息头

该请求的除了使用公共消息头外，还使用了两个扩展的消息头。公共消息头如表 5-165 所示。

表 5-165 请求消息头

参数名称	是否必选	参数类型	描述
x-obs-copy-source	是	String	**参数解释：** 拷贝的源对象。 示例：x-obs-copy-source：/SourceBucketName/SourceObjectNa

参数名称	是否必选	参数类型	描述
			me **约束限制：** 中文字符与%，需要进行 URL 编码。 **取值范围：** 不涉及 **默认取值：** 不涉及
x-obs-copy-source-range	否	Integer	**参数解释：** 源对象中待拷贝段的字节范围。 **约束限制：** 不涉及 **取值范围：** 取值范围为最小为 0，最大为对象长度减 1。 **默认取值：** 不涉及
x-obs-server-side-encryption-customer-algorithm	否。当使用 SSE-C 方式时，必选，需要与初始化上传段任务使用相同的算法。	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示加密目标段使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 需要和 x-obs-server-side-encryption-customer-key、 x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** AES256：使用 SSE-C 加密方式，且使用 AES256 加密算法。 **默认取值：** 不涉及
x-obs-server-side-encryption-customer-key	否。当使用 SSE-C 方式时，必选，需要与初始化上传段任务使用相同的密钥。	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示加密目标段使用的密钥。该密钥用于加密对象。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnN

参数名称	是否必选	参数类型	描述
			VaobncnLht/rCB2o/9Cw= **约束限制：** 该头域由 256-bit 的密钥经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm、x-obs-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 不涉及 **默认取值：** 不涉及
x-obs-server-side-encryption-customer-key-MD5	否。当使用 SSE-C 方式时，必选，需要与初始化上传段任务使用相同的 MD5 值。	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示加密目标段使用的密钥的 MD5 值。MD5 值用于确保密钥传输过程的数据完整性。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 该头域由密钥的 128-bit MD5 值经过 base64-encoded 得到，需要和 x-obs-server-side-encryption-customer-algorithm、x-obs-server-side-encryption-customer-key 一起使用。 **取值范围：** 密钥的 MD5 值。 **默认取值：** 不涉及
x-obs-copy-source-server-side-encryption-customer-algorithm	否。当拷贝源对象使用 SSE-C 方式时，必选。	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示源对象使用的加密算法。 示例：x-obs-copy-source-server-side-encryption-customer-algorithm：AES256 **约束限制：** 需要和 x-obs-copy-source-server-side-encryption-customer-key、x-obs-copy-source-server-side-encryption-customer-key-MD5 一起

参数名称	是否必选	参数类型	描述
			使用。 **取值范围：** AES256：使用 SSE-C 加密方式，且使用 AES256 加密算法。 **默认取值：** 不涉及
x-obs-copy-source-server-side-encryption-customer-key	否。当拷贝源对象使用 SSE-C 方式时，必选。	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示源对象使用的密钥。用于解密源对象。 示例：x-obs-copy-source-server-side-encryption-customer-key：K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** 该头域由 256bit 的密钥经过 base64-encoded 得到，需要和 x-obs-copy-source-server-side-encryption-customer-algorithm、x-obs-copy-source-server-side-encryption-customer-key-MD5 一起使用。 **取值范围：** 不涉及 **默认取值：** 不涉及
x-obs-copy-source-server-side-encryption-customer-key-MD5	否。当拷贝源对象使用 SSE-C 方式时，必选。	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示源对象使用的密钥的 MD5 值。MD5 值用于确保密钥传输过程的数据完整性。 示例：x-obs-copy-source-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 该头域由密钥的 128bit MD5 值经过 base64-encoded 得到，需要和 x-obs-copy-source-server-side-encryption-customer-algorithm、x-obs-copy-source-server-side-encryption-customer-key 一起使用。 **取值范围：**

参数名称	是否必选	参数类型	描述
			密钥的 MD5 值。 **默认取值：** 不涉及
x-obs-copy-source-if-match	否	String	**参数解释：** 只有当源对象的 Etag 与此参数指定的值相等时才进行复制对象操作，否则返回 412（前置条件不满足）。 示例：x-obs-copy-source-if-match: etag **约束限制：** 此参数可与 x-obs-copy-source-if-unmodified-since 一起使用，但不能与其它条件复制参数一起使用。 **取值范围：** 拷贝源对象的 ETag 值。 **默认取值：** 不涉及
x-obs-copy-source-if-none-match	否	String	**参数解释：** 只有当源对象的 Etag 与此参数指定的值不相等时才进行复制对象操作，否则返回 412（前置条件不满足）。 示例：x-obs-copy-source-if-none-match: etag **约束限制：** 此参数可与 x-obs-copy-source-if-modified-since 一起使用，但不能与其它条件复制参数一起使用。 **取值范围：** 不涉及 **默认取值：** 不涉及
x-obs-copy-source-if-unmodified-since	否	String	**参数解释：** 只有当源对象在此参数指定的时间之后没有修改过才进行复制对象操作，否则返回 412 错误码（未满足该参数设置的条件，返回前置条件不满足结果）。

参数名称	是否必选	参数类型	描述
			约束限制： • 此参数可与 x-obs-copy-source-if-match 一起使用，但不能与其它条件复制参数一起使用。 • 此参数指定的时间不能晚于当前的服务器时间（GMT 时间），否则参数不生效。 **取值范围：** 格式符合 RFC2616 规定格式的 HTTP 时间字符串。以下三种格式任选其一。 1. EEE, dd MMM yyyy HH:mm:ss z 2. EEEE, dd-MMM-yy HH:mm:ss z 3. EEE MMM dd HH:mm:ss yyyy 对应示例： 1. x-obs-copy-source-if-unmodified-since: Sun, 06 Nov 1994 08:49:37 GMT 2. x-obs-copy-source-if-unmodified-since: Sunday, 06-Nov-94 08:49:37 GMT 3. x-obs-copy-source-if-unmodified-since: Sun Nov 6 08:49:37 1994 **默认取值：** 不涉及
x-obs-copy-source-if-modified-since	否	String	**参数解释：** 只有当源对象在此参数指定的时间之后修改过才进行复制对象操作，否则返回 412 错误码（未满足该参数设置的条件，返回前置条件不满足结果）。 **约束限制：** • 此参数可与 x-obs-copy-source-if-none-match 一起使用，但不能与其它条件复制参数一起使用。 • 此参数指定的时间不能晚于当前的服务器时间（GMT 时间），否则参数不生效。 **取值范围：** 格式符合 RFC2616 规定格式的 HTTP 时间字符串。以下三种格式任选其一。

参数名称	是否必选	参数类型	描述
			1. EEE, dd MMM yyyy HH:mm:ss z 2. EEEE, dd-MMM-yy HH:mm:ss z 3. EEE MMM dd HH:mm:ss yyyy 对应示例： 1. x-obs-copy-source-if-unmodified-since: Sun, 06 Nov 1994 08:49:37 GMT 2. x-obs-copy-source-if-unmodified-since: Sunday, 06-Nov-94 08:49:37 GMT 3. x-obs-copy-source-if-unmodified-since: Sun Nov 6 08:49:37 1994 **默认取值：** 不涉及

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CopyPartResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <LastModified>modifiedDate</LastModified>
   <ETag>etag</ETag>

</CopyPartResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

表 5-166 附加响应消息头

参数名称	参数类型	描述
x-obs-server-side-encryption	String	**参数解释：** 该参数用于指定拷贝段副本的加密方式。 示例：x-obs-server-side-encryption：kms **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法

参数名称	参数类型	描述
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **取值范围：** 当您选择使用 kms 加密方式，但未设置此头域时，本头域为默认密钥的 ID。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的 KMS 主密钥所属的项目 ID（非企业项目 ID）。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **取值范围：** AES256：使用 SSE-C 加密方式，且使用 AES256 加密算法。
x-obs-server-side-encryption-customer-key-MD5	String	**参数解释：** 如果服务端加密是 SSE-C 方式，响应包含该头域，该头域表示加密使用的密钥的 MD5 值。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **取值范围：** 密钥的 MD5 值。

响应消息元素

该请求的响应消息使用消息元素来返回段拷贝的结果，元素的意义如表 5-167 所示。

表 5-167 响应消息元素

参数名称	参数类型	描述
LastModified	String	**参数解释：** 对象上次修改时间。时间格式为 ISO8601 的格式。

参数名称	参数类型	描述
		例如：2018-01-01T00:00:00.000Z。 **取值范围：** 不涉及
ETag	String	**参数解释：** 目标段的 ETag 值，是段内容的唯一标识，用于段合并时校验数据一致性。 **取值范围：** 长度为 32 的字符串。

错误响应消息

1. 如果 AccessKey 或签名无效，OBS 返回 403 Forbidden，错误码为 AccessDenied。
2. 查询源桶或目的桶是否存在，如果不存在，OBS 返回 404 Not Found，错误码为 NoSuchBucket。
3. 如果源对象不存在，OBS 返回 404 Not Found，错误码为 NoSuchKey。
4. 如果用户对指定对象没有读权限，OBS 返回 403 Forbidden，错误码为 AccessDenied。
5. 如果用户对目的桶没有写权限，OBS 返回 403 Forbidden，错误码为 AccessDenied。
6. 查询指定的任务不存在，OBS 返回 404 Not Found，错误码为 NoSuchUpload。
7. 如果用户不是多段上传任务的发起者，OBS 返回 403 Forbidden，错误码为 AccessDenied。
8. 当拷贝的单段超过 5G 时，OBS 返回 400 Bad Request。
9. 如果段序号超过范围[1,10000]，OBS 返回错误 400 Bad Request。

其他错误已包含在表 6-2 中。

请求示例

```
PUT /tobject02?partNumber=2&uploadId=00000163D40171ED8DF4050919BD02B8 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:16:32 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:dSnpnNpawDSsLg/xXxaqFzrAmMw=
x-obs-copy-source: /destbucket/object01
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D40ABBD20405D30B0542
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTIJpD2efLy5o8sTTComwBb2He0j11Ne
Content-Type: application/xml
Date: WED, 01 Jul 2015 05:16:32 GMT
Transfer-Encoding: chunked
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CopyPartResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <LastModified>2015-07-01T05:16:32.344Z</LastModified>
  <ETag>"3b46eaf02d3b6b1206078bb86a7b7013"</ETag>
</CopyPartResult>
```

5.5.5 列举已上传未合并的段

功能介绍

用户可以通过本接口查询一个未合并任务所属的所有段信息。此接口列举的各个段大小和分段上传的各个段大小一致。

请求消息样式

```
GET /ObjectName?uploadId=uploadid&max-parts=max&part-number-marker=marker HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: auth
```

请求消息参数

该请求通过请求消息参数指定多段上传任务以及列出的段数量，参数的具体含义如表 5-168 所示。

表 5-168 请求消息参数

参数名称	是否必选	参数类型	描述
uploadId	是	String	**参数解释：** 多段上传任务的 ID。 **约束限制：** 不涉及 **取值范围：** 长度大于 0 且不超过 32 的字符串。 **默认取值：** 不涉及
max-parts	否	Integer	**参数解释：** 指定单次返回请求中包含段的最大数目。单位：个。 **约束限制：** 不涉及 **取值范围：** [1,1000]，当超出范围时，按照默认的 1000 进行处理。

参数名称	是否必选	参数类型	描述
			默认取值： 1000
part-number-marker	否	Integer	**参数解释：** 指定列举的起始位置，只有上传段的段号数目大于该参数的段会被列出。 **约束限制：** 不涉及 **取值范围：** 从 1 到 10000 的整数。 **默认取值：** 不涉及

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Bucket>BucketName</Bucket>
   <Key>object</Key>
   <UploadId>uploadid</UploadId>
   <Initiator>
      <ID>id</ID>
   </Initiator>
   <Owner>
      <ID>ownerid</ID>
   </Owner>
   <StorageClass>storageclass</StorageClass>
   <PartNumberMarker>partNumbermarker</PartNumberMarker>
   <NextPartNumberMarker>nextPartnumberMarker</NextPartNumberMarker>
   <MaxParts>maxParts</MaxParts>
   <IsTruncated>true</IsTruncated>
   <Part>
      <PartNumber>partNumber</PartNumber>
      <LastModified>modifiedDate</LastModified>
      <ETag>etag</ETag>
```

```
      <Size>size</Size>
   </Part>
</ListPartsResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应通过消息元素返回已上传了的段信息，元素的具体含义如表 5-169 所示。

表 5-169 响应消息元素

参数名称	参数类型	描述
ListPartsResult	Container	**参数解释：** 保存列举段请求结果的容器，ListPartsResult 是 Bucket、Key、UploadId、Initiator、Owner、StorageClass、PartNumberMarker、NextPartNumberMarker、MaxParts、IsTruncated、Part 的父节点。 **取值范围：** 请详见表 5-170。

表 5-170 ListPartsResult 响应消息元素说明

参数名称	参数类型	描述
Bucket	String	**参数解释：** 桶名称。 **取值范围：** 长度为 3~63 的字符串。
Key	String	**参数解释：** 对象名称。 **取值范围：** 长度大于 0 且不超过 1024 的字符串。
UploadId	String	**参数解释：** 上传任务 ID。 **取值范围：** 长度大于 0 且不超过 32 的字符串。
Initiator	Container	**参数解释：** 多段上传任务的创建者，Initiator 是 ID 的父节点。 **取值范围：**

参数名称	参数类型	描述
		请详见表 5-171。
Owner	Container	**参数解释：** 段的所有者，Owner 是 ID 的父节点。 **取值范围：** 请详见表 5-172。
StorageClass	String	**参数解释：** 存储类型。 **取值范围：** • STANDARD（标准存储） • WARM（低频访问存储） • COLD（归档存储）
PartNumberMarker	Integer	**参数解释：** 本次列举结果的上传段的段号起始位置。 **取值范围：** [1，10000]
NextPartNumberMarker	Integer	**参数解释：** 如果本次没有返回全部结果，响应请求中将包含 NextPartNumberMarker 元素，用于标明接下来请求的 PartNumberMarker 值。 **取值范围：** [1，10000]
MaxParts	Integer	**参数解释：** 单次返回请求中包含段的最大数目。单位：个。 **取值范围：** [1，10000]
IsTruncated	Boolean	**参数解释：** 标明是否本次返回的列举段结果列表被截断。 **取值范围：** true：表示本次没有返回全部段。 false：表示本次已经返回了全部段。
Part	Container	**参数解释：** 保存段信息的容器，Part 是 PartNumber、LastModified、ETag、Size 的父节点。 **取值范围：** 请详见表 5-173。

表 5-171 Initiator 响应消息元素说明

参数名称	参数类型	描述
ID	String	**参数解释：** 多段任务创建者的 urn。格式为 domainID/{domainId}:userID/{userId}。 **取值范围：** 不涉及

表 5-172 Owner 响应消息元素说明

参数名称	参数类型	描述
ID	String	**参数解释：** 多段任务创建者的 DomainId，格式为{domainId}。 **取值范围：** 不涉及

表 5-173 Part 响应消息元素说明

参数名称	参数类型	描述
PartNumber	Integer	**参数解释：** 已上传段的编号。 **取值范围：** [1，10000]
LastModified	Date	**参数解释：** 段上传的时间。时间格式为 ISO8601 的格式。 例如：2018-01-01T00:00:00.000Z。 **取值范围：** 不涉及
ETag	String	**参数解释：** 已上传段内容的 ETag，是段内容的唯一标识，用于段合并时校验数据一致性。 **取值范围：** 不涉及
Size	Integer	**参数解释：** 已上传段大小。 **取值范围：** 不涉及

错误响应消息

1. 如果 AccessKey 或签名无效，OBS 返回 403 Forbidden，错误码为 AccessDenied。
2. 如果请求的桶不存在，OBS 返回 404 Not Found，错误码为 NoSuchBucket。
3. 如果请求的多段上传任务不存在，OBS 返回 404 Not Found，错误码为 NoSuchUpload。
4. OBS 判断用户 DomainId 是否具有指定桶的读权限，如果没有权限，则 OBS 返回 403 Forbidden，错误码为 AccessDenied。

其他错误已经包含在表 6-2 中。

请求示例

```
GET /object02?uploadId=00000163D40171ED8DF4050919BD02B8 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:20:35 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:xkABdSrBPrz5yqzuZdJnK5oL/yU=
```

响应示例

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D40C099A04EF4DD1BDD9
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSK71fr+hDnzB0JBvQC1B9+S12AWxC41
Content-Type: application/xml
Date: WED, 01 Jul 2015 05:20:35 GMT
Content-Length: 888

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Bucket>test333</Bucket>
  <Key>obj2</Key>
  <UploadId>00000163D40171ED8DF4050919BD02B8</UploadId>
  <Initiator>

<ID>domainID/domainiddomainiddomainiddo000008:userID/useriduseriduseriduseridus0000
08</ID>
  </Initiator>
  <Owner>
    <ID>domainiddomainiddomainiddo000008</ID>
  </Owner>
  <StorageClass>STANDARD</StorageClass>
  <PartNumberMarker>0</PartNumberMarker>
  <NextPartNumberMarker>2</NextPartNumberMarker>
  <MaxParts>1000</MaxParts>
  <IsTruncated>false</IsTruncated>
  <Part>
    <PartNumber>1</PartNumber>
    <LastModified>2018-06-06T07:39:32.522Z</LastModified>
    <ETag>"b026324c6904b2a9cb4b88d6d61c81d1"</ETag>
    <Size>2058462721</Size>
  </Part>
```

```
  <Part>
    <PartNumber>2</PartNumber>
    <LastModified>2018-06-06T07:41:03.344Z</LastModified>
    <ETag>"3b46eaf02d3b6b1206078bb86a7b7013"</ETag>
    <Size>4572</Size>
  </Part>
</ListPartsResult>
```

5.5.6 合并段

功能介绍

如果用户上传完所有的段，就可以调用合并段接口，系统将在服务端将用户指定的段合并成一个完整的对象。在执行“合并段”操作以前，用户不能下载已经上传的数据。在合并段时需要将多段上传任务初始化时记录的附加消息头信息拷贝到对象元数据中，其处理过程和普通上传对象带这些消息头的处理过程相同。在并发合并段的情况下，仍然遵循 Last Write Win 策略，但“Last Write”的时间定义为段任务的初始化时间。

已经上传的段，只要没有取消对应的多段上传任务，都要占用用户的容量配额；对应的多段上传任务“合并段”操作完成后，只有指定的多段数据占用容量配额，用户上传的其他此多段任务对应的段数据如果没有包含在“合并段”操作指定的段列表中，“合并段”完成后删除多余的段数据，且同时释放容量配额。

合并完成的多段上传数据可以通过已有的下载对象接口，下载整个多段上传对象或者指定 Range 下载整个多段上传对象的某部分数据。

合并完成的多段上传数据可以通过已有的删除对象接口，删除整个多段上传对象的所有分段数据，删除后不可恢复。

合并完成的多段上传数据不记录整个对象的 MD5 作为 Etag，在下载多段数据或 List 桶内对象看到的多段数据其 Etag 的生成方式为：$MD5（M_1M_2……M_N）$-N，其中，M_n 表示第 n 段的 MD5 值， 如请求示例所示，有 3 个分段，每个分段都有对应的 MD5 值，合并段 ETag 的生成是先将 3 个分段的 MD5 合并起来再进行 MD5 计算得到一个新值，再拼接-N 作为合并段的 ETag 值，-N 表示总共有多少段，在该示例中即为-3。

合并段请求如果出现等待响应超时、服务端返回 500 或 503 报错时，建议客户端可以先尝试获取多段任务对应的对象元数据，并比较响应的 x-obs-uploadId 头域的值，是否与本次要合并的段任务 ID 相同，如果相同，说明服务端实际已经合并段成功，无需再进行重试。关于并发一致性说明，参见 8.4 并发一致性说明。

WORM

如果桶的 WORM 开关是开启的，合并段会为合并后生成的对象自动应用 WORM 保护。如果您在初始化时指定了 WORM 相关头域并配置了保护截止日期，则会使用该日期作为合并生成对象的保护截止日期。如果您在初始化时没有指定 WORM 相关头域，但配置了桶级默认保护策略，则会自动应用桶级默认保护策略的保护期限，实际的保护截止日期为合并时间+保护期限。在您合并后，您依旧可以正常为合并后的对象配置对象级 WORM 保护策略。

多版本

如果桶的多版本状态是开启的，则合并段后得到的对象生成一个唯一的版本号，并且会在响应报头 x-obs-version-id 返回该版本号。如果桶的多版本状态是暂停的，则合并段后得到的对象版本号为 **null**。关于桶的多版本状态，参见 5.2.11 设置桶的多版本状态。

须知

如果上传了 10 个段，但合并时只选择了 9 个段进行合并，那么未被合并的段将会被系统自动删除，未被合并的段删除后不能恢复。在进行合并之前请使用列出已上传的段接口进行查询，仔细核对所有段，确保没有段被遗漏。

请求消息样式

```
POST /ObjectName?uploadId=uploadID HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Length: length
Authorization: authorization
<CompleteMultipartUpload>
   <Part>
      <PartNumber>partNum</PartNumber>
      <ETag>etag</ETag>
   </Part>
   <Part>
      <PartNumber>partNum</PartNumber>
      <ETag>etag</ETag>
   </Part>
   <Part>
      <PartNumber>partNum</PartNumber>
      <ETag>etag</ETag>
   </Part>
</CompleteMultipartUpload>
```

请求消息参数

该请求在消息参数中指定多段上传任务号来标明它要合并哪一个上传段任务，参数意义如表 5-174 所示。

表 5-174 请求消息参数

参数名称	参数类型	是否必选	描述
uploadId	String	是	**参数解释：** 指明分段上传任务的 ID。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求消息元素

该请求需要在消息中带消息元素，指定要合并的段列表，元素的具体意义如表 5-175 中所示

表 5-175 请求消息元素

元素名称	元素类型	是否必选	描述
CompleteMultipartUpload	XML	是	**参数解释：** 合并的段列表。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
part	XML	是	**参数解释：** 分段信息的父节点，即 PartNumber 和 ETag 的父节点，表示要合并的一个段。 **约束限制：** 无 **取值范围：** 无 **默认取值：** 无
PartNumber	Integer	是	**参数解释：** 段号。 **约束限制：** 无 **取值范围：** [1,10000] **默认取值：** 无
ETag	String	是	**参数解释：** 上传段成功后返回的 ETag 值，是段

元素名称	元素类型	是否必选	描述
			内容的唯一标识。该值用于合并段时进行数据一致性校验。 **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

响应消息样式

```
HTTP/1.1 status_code
Date: date
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>http://example-Bucket.obs.region.example.com/example-Object</Location>
   <Bucket>bucketname</Bucket>
   <Key>ObjectName</Key>
   <ETag>ETag</ETag>
</CompleteMultipartUploadResult>
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

除公共响应消息头之外，还可能使用如表 5-176 中的消息头。

表 5-176 附加响应消息头

消息头名称	消息头类型	描述
x-obs-version-id	String	**参数解释：** 合并得到的对象的版本号。 **约束限制：** 无 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无
x-obs-server-side-encryption	String	**参数解释：** 该头域表示服务端的加密方式。 示例：x-obs-server-side-encryption：kms **约束限制：**

消息头名称	消息头类型	描述
		如果服务端加密是 SSE-KMS 方式，响应包含该头域。 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 无
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption 头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。
x-obs-sse-kms-key-project-id	String	**参数解释：** 加密方式为 SSE-KMS 且使用自定义 KMS 密钥加密时，返回 KMS 主密钥所属的项目 ID（非企业项目 ID）。 **取值范围：** x-obs-server-side-encryption-kms-key-id 指定的 KMS 主密钥所属的项目 ID（非企业项目 ID）。
x-obs-server-side-encryption-customer-algorithm	String	**参数解释：** 该头域表示加密使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm：AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** AES256 **默认取值：** 无

响应消息元素

该请求的响应消息中通过返回消息元素来返回合并段的结果，元素的具体意义如表 5-177 所示。

表 5-177 响应消息元素

元素名称	元素类型	描述
Location	String	**参数解释：** 合并后得到对象的路径。 **约束限制：** 格式： http://bucketName.obs.*region*.example.com/objectName **取值范围：** 无 **默认取值：** 无
Bucket	String	**参数解释：** 合并段所在的桶名。 **约束限制：** • 桶的名字需全局唯一，不能与已有的任何桶名称重复，包括其他用户创建的桶。 • 桶命名规则如下： – 3～63 个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。 – 禁止使用 IP 地址。 – 禁止以“-”或“.”开头及结尾。 – 禁止两个“.”相邻（如：“my..bucket”）。 – 禁止“.”和“-”相邻（如：“my-.bucket”和“my.-bucket”）。 • 同一用户在同一个区域多次创建同名桶不会报错，创建的桶属性以第一次请求为准。 **取值范围：** 无 **默认取值：** 无
Key	String	**参数解释：** 合并段后得到的对象名。 对象名是对象在存储桶中的唯一标识。对象名是

元素名称	元素类型	描述
		对象在桶中的完整路径，路径中不包含桶名。 **约束限制：** 无。 **取值范围：** 长度大于 0 且不超过 1024 的字符。 **默认取值：** 无
ETag	String	**参数解释：** 合并段后根据各个段的 ETag 值计算出的结果，是对象内容的唯一标识。 **约束限制：** 当对象是服务端加密的对象时，ETag 值不是对象的 MD5 值。 **取值范围：** 长度为 32 的字符串。 **默认取值：** 无

错误响应消息

1. 如果没有消息体，OBS 返回 400 Bad Request。
2. 如果消息体格式不正确，OBS 返回 400 Bad Request。
3. 消息体中如果段信息未按照段序号升序排列，OBS 返回 400 Bad Request，错误码为 InvalidPartOrder。
4. 如果 AccessKey 或签名无效，OBS 返回 403 Forbidden， 错误码为 AccessDenied。
5. 如果请求的桶不存在，OBS 返回 404 Not Found，错误码为 NoSuchBucket。
6. 如果请求的多段上传任务不存在，OBS 返回 404 Not Found，包含错误信息 NoSuchUpload。
7. 如果用户不是该任务的发起者（initiator），OBS 返回 403 Forbidden，错误码为 AccessDenied。
8. 在合并段时如果请求段列表中包含了不存在的段，OBS 返回 400 Bad Request，错误码为 InvalidPart。
9. 如果请求段列表中包含的段的 Etag 错误，OBS 返回 400 Bad Request，错误码为 InvalidPart。
10. 除最后一个段之外的其它段尺寸过小（小于 100KB），OBS 返回 400 Bad Request。
11. 对象在合并完成后总大小如果超过 48.8TB，OBS 返回 400 Bad Request。

其他错误已包含在表 6-2 中。

请求示例

```
POST /object02?uploadId=00000163D46218698DF407362295674C HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:23:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:dOfK9iILcKxo58tRp3fWeDoYzKA=
Content-Length: 422

<?xml version="1.0" encoding="utf-8"?>
<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
Server: OBS
x-obs-request-id: 8DF400000163D4625BE3075019BD02B8
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSN8D1AfQcIvyGBZ9+Ee+jU6zv1iYdO4
Content-Type: application/xml
Date: WED, 01 Jul 2015 05:23:46 GMT
Content-Length: 326

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-
30/">
   <Location>http://examplebucket.obs.region.example.com/object02</Location>
   <Bucket>examplebucket</Bucket>
   <Key>object02</Key>
   <ETag>"03f814825e5a691489b947a2e120b2d3-3"</ETag>
</CompleteMultipartUploadResult>
```

5.5.7 取消多段上传任务

功能介绍

如果用户希望取消一个任务，可以调用取消多段上传任务接口取消任务。合并段或取消任务接口被调用后，用户不能再对任务进行上传段和列举段的操作。

请求消息样式

```
DELETE /ObjectName?uploadId=uploadID HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: auth
```

请求消息参数

该请求通过消息参数，指定要取消的段任务的多段上传任务号，参数的意义如表 5-178 所示。

表 5-178 请求消息参数

参数名称	是否必选	参数类型	描述
uploadId	是	String	**参数解释：** 多段上传任务的 ID。 **约束限制：** 不涉及 **取值范围：** 长度大于 0 且不超过 32 的字符串。 **默认取值：** 不涉及

请求消息头

该请求使用公共消息头，具体请参考表 3-3。

请求消息元素

该请求消息中不使用消息元素。

响应消息样式

```
HTTP/1.1 status_code
Date: date
```

响应消息头

该请求的响应消息使用公共消息头，具体请参考表 3-19。

响应消息元素

该请求的响应消息中不带消息元素。

错误响应消息

1. 如果 AccessKey 或签名无效，OBS 返回 403 Forbidden，错误码为 AccessDenied。

2. 如果请求的桶不存在，OBS 返回 404 Not Found，错误码为 NoSuchBucket。
3. 用户执行取消多段上传任务操作时判断用户是否是任务初始化者或是桶的所有者，如果不是则 OBS 则返回 403 Forbidden。
4. 操作成功，OBS 向用户返回 204 No Content。

其他错误已包含在表 6-2 中。

请求示例

```
DELETE /object02?uploadId=00000163D46218698DF407362295674C HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:28:27 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:QmM2d1DBXZ/b8drqtEv1QJHPbM0=
```

响应示例

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 8DF400000163D463E02A07EC2295674C
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTp5YDlzn0UgqG3laRfkHLGyz7RpR9ON
Date: WED, 01 Jul 2015 05:28:27 GMT
```

5.6 服务端加密

5.6.1 服务端加密简介

用户可以使用普通方式上传、下载对象，也可以使用服务端加密方式进行上传、下载对象。

OBS 支持服务端加密功能，使加密的行为在服务端进行。

用户可以根据自身的需求，使用不同的密钥管理方式来使用服务端加密功能。当前支持的服务端加密方式： KMS 托管密钥的服务端加密(SSE-KMS)和客户提供加密密钥的服务端加密(SSE-C)。上述方式都采用行业标准的 AES256 加密算法。

SSE-KMS 方式，OBS 使用 KMS（Key Management Service）服务提供的密钥进行服务端加密。用户可以创建自定义密钥，用于 SSE-KMS 加密。

SSE-C 方式，OBS 使用用户提供的密钥和密钥的 MD5 值进行服务端加密。

使用服务端加密，返回的 ETag 值不是对象的 MD5 值。

当使用服务端加密时，建议您使用 HTTPS 对数据进行传输和接收。

5.6.2 服务端加密 SSE-KMS 方式

功能介绍

SSE-KMS 方式，OBS 使用 KMS（Key Management Service）服务提供的密钥进行服务端加密。用户可以创建自定义密钥，用于 SSE-KMS 加密。如果未指定，则用户首次向区域中的桶上传 SSE-KMS 加密的对象时，OBS 将自动为您创建一个默认密钥。自定义密钥或默认密钥用于加密和解密 KMS 生成的数据加解密密钥。

说明

使用非默认 IAM 项目下的自定义密钥对桶内对象进行 SSE-KMS 加密，只有密钥拥有者可以对加密后的对象进行上传下载类操作，非密钥拥有者不能对加密对象进行上传下载类操作。

使用默认密钥向区域中的桶上传 SSE-KMS 加密的对象时，该默认密钥归属于对象上传者，非密钥拥有者不能对使用默认密钥加密的对象进行上传下载类操作。

新增头域

SSE-KMS 方式新增加两个头域，用户可以通过配置表 5-179 中的头域来实现 SSE-KMS 加密。

您也可以通过配置桶的默认加密方式来对桶内的对象进行加密。在为桶配置默认加密后，对于任何不带加密头域的上传对象的请求，将使用桶的默认加密配置进行加密。有关桶的加密配置的更多信息请参考 5.2.31 设置桶的加密配置章节。

表 5-179 SSE-KMS 方式使用的头域

头域名称	头域类型	描述
x-obs-server-side-encryption	String	**参数解释：** 该参数用于指定加密方式。 示例：x-obs-server-side-encryption：AES256 **约束限制：** 不涉及 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 不涉及
x-obs-server-side-encryption-kms-key-id	String	**参数描述：** 当加密方式为 SSE-KMS 且使用指定密钥加密时，需输入密钥 ID。 **约束限制：** 当您设置了 x-obs-server-side-encryption

头域名称	头域类型	描述
		头域且赋值为“kms”，即选择 kms 加密方式时，才能使用该头域指定加密密钥。 **默认取值：** 当您选择使用 kms 加密方式，但未设置此头域时，默认的主密钥将会被使用。如果默认主密钥不存在，系统将默认创建并使用。

支持头域的接口

以下接口支持配置 SSE-KMS 相关头域：

- 5.4.1 PUT 上传
- 5.4.2 POST 上传（需要将 x-obs-server-side-encryption 和 x-obs-server-side-encryption-kms-key-id 放到表单中，而不是头域中）
- 5.4.3 复制对象 （新增的头域针对目标对象）
- 5.5.2 初始化上传段任务

您可以通过设置桶策略，来限制指定桶的请求头域，如果您要对桶中的所有对象执行服务端加密限制，则可通过设置桶策略达成。例如，如果要求本租户的上传对象请求不包含服务端加密 (SSE-KMS) 的相关头域 x-obs-server-side-encryption:"kms"，则可使用以下桶策略达成：

```
{
   "Statement": [
      {
         "Sid": "DenyUnEncryptedObjectUploads",
         "Effect": "Deny",
         "Principal": "*",
         "Action": "PutObject",
         "Resource": "YourBucket/*",
         "Condition": {
            "StringNotEquals": {
               "x-obs-server-side-encryption": "kms"
            }
         }
      }
   ]
}
```

请求示例：使用默认密钥对上传的对象进行加密

```
PUT /encryp1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:08:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:f3/7eS6MFbW3JO4+7I5AtyAQENU=
x-obs-server-side-encryption:kms
```

```
Content-Length: 5242
Expect: 100-continue

[5242 Byte object contents]
```

响应示例：使用默认密钥对上传的对象进行加密

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D45AA81D038B6AE4C482
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: kms
x-obs-server-side-encryption-kms-key-id:
region:783fc6652cf246c096ea836694f71855:key/522d6070-5ad3-4765-9737-9312ddc72cdb
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTv7cHmAnGfBAGXUHeibUsiETTNqlCqC
Date: Wed, 06 Jun 2018 09:08:21 GMT
Content-Length: 0
```

请求示例：使用指定密钥对上传的对象进行加密

```
PUT /encryp1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:08:50 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:f3/PWjkXYTYGs5lPOctTNEI2QENU=
x-obs-server-side-encryption:kms
x-obs-server-side-encryption-kms-key-id: 522d6070-5ad3-4765-43a7-a7d1-ab21f498482d
Content-Length: 5242
Expect: 100-continue

[5242 Byte object contents]
```

响应示例：使用指定密钥对上传的对象进行加密

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D45AA81D038B6AE4C482
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: kms
x-obs-server-side-encryption-kms-key-id:
region:783fc6652cf246c096ea836694f71855:key/522d6070-5ad3-4765-43a7-a7d1-
ab21f498482d
x-obs-id-2: 32AAAUJAIAABAdiAEAABA09AEAABCTv7cHmAn12BAG83ibUsiET5eqlCqg
Date: Wed, 06 Jun 2018 09:08:50 GMT
Content-Length: 0
```

请求示例：将普通对象拷贝为加密对象，且指定加密密钥

```
PUT /destobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
x-obs-server-side-encryption:kms
x-obs-server-side-encryption-kms-key-id:
region:783fc6652cf246c096ea836694f71855:key/522d6070-5ad3-4765-9737-9312ddc72cdb
Accept: */*
```

```
Date: Wed, 06 Jun 2018 09:10:29 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:SH3uTrElaGWarVI1uTq325kTVCI=
x-obs-copy-source: /bucket/srcobject1
```

响应示例：将普通对象拷贝为加密对象，且指定加密密钥

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB78000001648480AF3900CED7F15155
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: kms
x-obs-server-side-encryption-kms-key-id:
region:783fc6652cf246c096ea836694f71855:key/522d6070-5ad3-4765-9737-9312ddc72cdb
x-obs-id-2: oRAXhgwdaLc9wKVHqTLSmQB7I35D+32AAAUJAIAABAAAQAAEAABAAAQAAEAABCS
Date: Wed, 06 Jun 2018 09:10:29 GMT
Content-Length: 0
```

请求示例：在 URL 中携带签名并上传加密对象

```
PUT /destobject?AccessKeyId=UI3SN1SRUQE14OYBKTZB&Expires=1534152518&x-obs-server-
side-encryption=kms&Signature=chvmG7%2FDA%2FDCQmTRJu3xngldJpg%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:10:29 GMT
```

响应示例：在 URL 中携带签名并上传加密对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB78000001648480AF3900CED7F15155
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: kms
x-obs-server-side-encryption-kms-key-id:
region:783fc6652cf246c096ea836694f71855:key/522d6070-5ad3-4765-9737-9312ddc72cdb
x-obs-id-2: oRAXhgwdaLc9wKVHqTLSmQB7I35D+32AAAUJAIAABAAAQAAEAABAAAQAAEAABCS
Date: Wed, 06 Jun 2018 09:10:29 GMT
Content-Length: 0
```

5.6.3 服务端加密 SSE-OBS 方式

功能介绍

SSE-OBS 方式，OBS 使用服务自身提供的密钥进行服务端加密。与 SSE-KMS 的区别在于 SSE-OBS 是 OBS 管理密钥，而非 KMS。

新增头域

SSE-OBS 方式用户可以通过配置表 5-180 中的头域来实现 SSE-OBS 加密。

您也可以通过配置桶的默认加密方式来对桶内的对象进行加密。在为桶配置默认加密后，对于任何不带加密头域的上传对象的请求，将使用桶的默认加密配置进行加密。有关桶的加密配置的更多信息请参考 5.2.31 设置桶的加密配置章节。

表 5-180 SSE-OBS 方式使用的头域

参数名称	是否必选	参数类型	描述
x-obs-server-side-encryption	是	String	**参数解释：** 该参数用于指定加密方式。 示例：x-obs-server-side-encryption：AES256 **约束限制：** 不涉及 **取值范围：** • kms：使用 SSE-KMS 加密方式 • AES256：使用 SSE-OBS 加密方式，且使用 AES256 算法 **默认取值：** 不涉及

支持头域的接口

以下接口支持配置 SSE-OBS 相关头域：

- 5.4.1 PUT 上传
- 5.4.2 POST 上传（需要将 x-obs-server-side-encryption 放到表单中，而不是头域中）
- 5.4.3 复制对象 （新增的头域针对目标对象）
- 5.5.2 初始化上传段任务

您可以通过设置桶策略，来限制指定桶的请求头域，如果您要对桶中的所有对象执行服务端加密限制，则可通过设置桶策略达成。例如，如果要求本租户的上传对象请求不包含服务端加密 (SSE-OBS) 的相关头域 x-obs-server-side-encryption:"AES256"，则可使用以下桶策略达成：

```
{
   "Statement": [
      {
         "Sid": "DenyUnEncryptedObjectUploads",
         "Effect": "Deny",
         "Principal": "*",
         "Action": "PutObject",
         "Resource": "YourBucket/*",
         "Condition": {
            "StringNotEquals": {
               "x-obs-server-side-encryption": "AES256"
            }
```

```
            }
        }
    ]
}
```

请求示例：使用默认密钥对上传的对象进行加密

```
PUT /encryp1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:08:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:f3/7eS6MFbW3JO4+7I5AtyAQENU=
x-obs-server-side-encryption:AES256
Content-Length: 5242
Expect: 100-continue

[5242 Byte object contents]
```

响应示例：使用默认密钥对上传的对象进行加密

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D45AA81D038B6AE4C482
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: AES256
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTv7cHmAnGfBAGXUHeibUsiETTNqlCqC
Date: Wed, 06 Jun 2018 09:08:21 GMT
Content-Length: 0
```

请求示例：将普通对象拷贝为加密对象

```
PUT /destobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
x-obs-server-side-encryption:AES256
Accept: */*
Date: Wed, 06 Jun 2018 09:10:29 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:SH3uTrElaGWarVI1uTq325kTVCI=
x-obs-copy-source: /bucket/srcobject1
```

响应示例：将普通对象拷贝为加密对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB78000001648480AF3900CED7F15155
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: AES256
x-obs-id-2: oRAXhgwdaLc9wKVHqTLSmQB7I35D+32AAAUJAIAABAAAQAAEAABAAAQAAEAABCS
Date: Wed, 06 Jun 2018 09:10:29 GMT
Content-Length: 0
```

请求示例：在 URL 中携带签名并上传加密对象

```
PUT /destobject?AccessKeyId=UI3SN1SRUQE14OYBKTZB&Expires=1534152518&x-obs-server-side-encryption=AES256&Signature=chvmG7%2FDA%2FDCQmTRJu3xngldJpg%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:10:29 GMT
```

响应示例：在 URL 中携带签名并上传加密对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB78000001648480AF3900CED7F15155
ETag: "d8bffdfbab5345d91ac05141789d2477"
x-obs-server-side-encryption: AES256
x-obs-id-2: oRAXhgwdaLc9wKVHqTLSmQB7I35D+32AAAUJAIAABAAAQAAEAABAAAQAAEAABCS
Date: Wed, 06 Jun 2018 09:10:29 GMT
Content-Length: 0
```

5.6.4 服务端加密 SSE-C 方式

功能介绍

SSE-C 方式，OBS 使用用户提供的密钥和密钥的 MD5 值进行服务端加密。

新增头域

OBS 不存储您提供的加密密钥，如果您丢失加密密钥，则会无法获取该对象。SSE-C 方式新增加六个头域来支持 SSE-C 加密。

使用 SSE-C 方式加密对象，您必须使用下面的三个头域。

表 5-181 SSE-C 方式加密对象使用的头域

参数名称	是否必选	参数类型	描述
x-obs-server-side-encryption-customer-algorithm	是	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示加密对象使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** 如果服务端加密是 SSE-C 方式，响应包含该头域。 **取值范围：** AES256：使用 SSE-C 加密方式，且使用 AES256 加密算法。

参数名称	是否必选	参数类型	描述
			默认取值： 不涉及
x-obs-server-side-encryption-customer-key	是	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示加密对象使用的密钥，头域值是 256bit 密钥的 base64 编码。 示例：x-obs-server-side-encryption-customer-key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** 仅在 SSE-C 加密方式下使用该头域。 **取值范围：** 不涉及 **默认取值：** 不涉及
x-obs-server-side-encryption-customer-key-MD5	是	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示加密对象使用的密钥的 MD5 值，头域值是加密密钥 MD5 值的 base64 编码。MD5 值用于确保密钥传输过程的数据完整性。 示例：x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 仅在 SSE-C 加密方式下使用该头域。 **取值范围：** 密钥的 MD5 值。 **默认取值：** 不涉及

该新增的三个头域可以应用于如下接口：

- 5.4.1 PUT 上传
- 5.4.2 POST 上传
- 5.4.3 复制对象（新增的头域针对目标对象）
- 5.4.5 获取对象元数据
- 5.4.4 下载对象
- 5.5.2 初始化上传段任务
- 5.5.3 上传段
- 5.5.4 拷贝段（新增的头域针对目标段）

针对复制对象和拷贝段，另外增加三个头域支持源对象是 SSE-C 加密的场景。

表 5-182 源对象是 SSE-C 加密的头域

参数名称	是否必选	参数类型	描述
x-obs-copy-source-server-side-encryption-customer-algorithm	是	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示解密源对象使用的算法。 示例：x-obs-server-side-encryption-customer-algorithm: AES256 **约束限制：** 仅在 SSE-C 加密方式下使用该头域。 **取值范围：** AES256：使用 SSE-C 加密方式，且使用 AES256 加密算法。 **默认取值：** 不涉及
x-obs-copy-source-server-side-encryption-customer-key	是	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示解密源对象使用的密钥的 base64 值。 示例：x-obs-copy-source-server-side-encryption-customer-algorithm: K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw= **约束限制：** 仅在 SSE-C 加密方式下使用该头域。 **取值范围：** 不涉及 **默认取值：** 不涉及
x-obs-copy-source-server-side-encryption-customer-key-MD5	是	String	**参数解释：** SSE-C 方式下使用该头域，该头域表示解密源对象使用的密钥的 MD5 值。MD5 值用于确保密钥传输过程的数据完整性。 示例：x-obs-copy-source-server-side-encryption-customer-key:4XvB3tbNTN+tIEVa0/fGaQ== **约束限制：** 仅在 SSE-C 加密方式下使用该头域。 **取值范围：**

参数名称	是否必选	参数类型	描述
			密钥的 MD5 值。 **默认取值：** 不涉及

请求示例：上传 SSE-C 加密对象

```
PUT /encryp2 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:12:00 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mZSfafoM+llApk0HGOThlqeccu0=
x-obs-server-side-encryption-customer-algorithm:AES256
x-obs-server-side-encryption-customer-
key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw=
x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ==
Content-Length: 5242

[5242 Byte object contents]
```

响应示例：上传 SSE-C 加密对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D45E0017055619BD02B8
ETag: "0f91242c7f3d86f98ae572a686d0696e"
x-obs-server-side-encryption-customer-algorithm: AES256
x-obs-server-side-encryption-customer-key-MD5: 4XvB3tbNTN+tIEVa0/fGaQ==
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSSAJ8bTNJV0X+Ote1PtuWecqyMh6zBJ
Date: Wed, 06 Jun 2018 09:12:00 GMT
Content-Length: 0
```

请求示例：将 SSE-C 加密对象拷贝为 KMS 加密对象

```
PUT /kmsobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 06 Jun 2018 09:20:10 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mZSfafoM+llApk0HGOThlqeccu0=
x-obs-copy-source-server-side-encryption-customer-algorithm:AES256
x-obs-copy-source-server-side-encryption-customer-
key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw=
x-obs-copy-source-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ==
x-obs-server-side-encryption: kms
x-obs-copy-source: /examplebucket/encryp2
Content-Length: 5242

[5242 Byte object contents]
```

响应示例：将 SSE-C 加密对象拷贝为 KMS 加密对象

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB7800000164848E0FC70528B9D92C41
ETag: "1072e1b96b47d7ec859710068aa70d57"
x-obs-server-side-encryption: kms
x-obs-server-side-encryption-kms-key-id:
region:783fc6652cf246c096ea836694f71855:key/522d6070-5ad3-4765-9737-9312ddc72cdb
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTkkRzQXs9ECzZcavVRncBqqYNkoAEsr
Date: Wed, 06 Jun 2018 09:20:10 GMT
Content-Length: 0
```

请求示例：在 URL 中携带签名并上传 SSE-C 加密对象

```
PUT
/encrypobject?AccessKeyId=H4IPJX0TQTHTHEBQQCEC&Expires=1532688887&Signature=EQmDuOh
aLUrzrzRNZxwS72CXeXM%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
x-obs-server-side-encryption-customer-algorithm: AES256
x-obs-server-side-encryption-customer-
key:K7QkYpBkM5+hca27fsNkUnNVaobncnLht/rCB2o/9Cw=
x-obs-server-side-encryption-customer-key-MD5:4XvB3tbNTN+tIEVa0/fGaQ==
Content-Length: 5242
Expect: 100-continue

[5242 Byte object contents]
```

响应示例：在 URL 中携带签名并上传 SSE-C 加密对象

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "1072e1b96b47d7ec859710068aa70d57"
x-obs-server-side-encryption-customer-algorithm: AES256
x-obs-server-side-encryption-customer-key-MD5: 4XvB3tbNTN+tIEVa0/fGaQ==
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Content-Length: 0
```

5.6.5 与服务端加密相关的接口

本节介绍与服务端加密相关的接口，以及使用该接口时服务端加密对传输协议使用要求。

与服务端加密相关的接口对使用传输协议的具体要求，详情如表所示。

表 5-183 与 SSE-C 相关的接口对传输协议的使用要求

接口	传输协议

接口	传输协议
PUT 上传对象	HTTPS
POST 上传对象	HTTPS
初始化上传段任务	HTTPS
获取对象元数据	HTTPS
获取对象内容	HTTPS
上传段	HTTPS
合并段	HTTP or HTTPS

表 5-184 与 SSE-KMS 相关的接口对传输协议的使用要求

接口	传输协议
PUT 上传对象	HTTPS
POST 上传对象	HTTPS
初始化上传段任务	HTTPS
获取对象元数据	HTTP or HTTPS
获取对象内容	HTTPS
上传段	HTTPS
合并段	HTTP or HTTPS

表 5-185 复制对象接口对传输协议的使用要求

源对象	目标对象	传输协议
非加密对象	SSE-KMS 加密对象	HTTPS
SSE-KMS 加密对象	SSE-KMS 加密对象	HTTPS
SSE-C 加密对象	SSE-KMS 加密对象	HTTPS
非加密对象	SSE-C 加密对象	HTTPS
SSE-KMS 加密对象	SSE-C 加密对象	HTTPS
SSE-C 加密对象	SSE-C 加密对象	HTTPS
非加密对象	非加密对象	HTTP or HTTPS
SSE-KMS 加密对象	非加密对象	HTTP or HTTPS
SSE-C 加密对象	非加密对象	HTTP or HTTPS

表 5-186 拷贝段接口对传输协议的使用要求

源对象	目标段	传输协议
非加密对象	SSE-KMS 加密段	HTTP or HTTPS
SSE-KMS 加密对象	SSE-KMS 加密段	HTTP or HTTPS
SSE-C 加密对象	SSE-KMS 加密段	HTTP or HTTPS
非加密对象	SSE-C 加密段	HTTPS
SSE-KMS 加密对象	SSE-C 加密段	HTTPS
SSE-C 加密对象	SSE-C 加密段	HTTPS
非加密对象	非加密段	HTTP or HTTPS
SSE-KMS 加密对象	非加密段	HTTP or HTTPS
SSE-C 加密对象	非加密段	HTTP or HTTPS

6 错误码

调用接口出错后，将不会返回结果数据。调用方可根据每个接口对应的错误码来定位错误原因。 当调用出错时，HTTP 请求返回一个 3xx，4xx 或 5xx 的 HTTP 状态码。返回的消息体中是具体的错误代码及错误信息。

错误响应消息格式

当错误发生时，响应消息头中都会包含：

- Content-Type: application/xml
- 错误对应的 3xx，4xx 或 5xx 的 HTTP 状态码。

响应消息体中同样会包含对错误的描述信息。下面的错误响应消息体示例展示了所有 REST 错误响应中公共的元素。

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
<Code>NoSuchKey</Code>
<Message>The resource you requested does not exist</Message>
<Resource>/example-bucket/object</Resource>
<RequestId>001B21A61C6C0000013402C4616D5285</RequestId>
<HostId>RkRCRDJENDc5MzdGQkQ4OUY3MTI4NTQ3NDk2Mjg0M0FB
QUFBQUFBYmJiYmJiYmJD</HostId>
</Error>
```

各元素的具体含义如表 6-1 所示。

表 6-1 错误响应消息元素

元素名称	描述
Error	错误响应消息体 XML 结构中描述错误信息的根节点元素 。
Code	错误响应消息体 XML 中错误响应对应的 HTTP 消息返回码，具体的错误码请参见表 6-2。
Message	错误响应消息体 XML 中具体错误更全面、详细的英文解释，具体的错误消息请参见表 6-2。

元素名称	描述
RequestId	本次错误请求的请求 ID，用于错误定位。
HostId	返回该消息的服务端 ID。
Resource	该错误相关的桶或对象资源。

说明

许多错误响应包含其他的更丰富的错误信息， 建议将所有错误信息记入日志，方便程序员在诊断程序错误时阅读和理解。

错误码说明

在向 OBS 系统发出请求后，如果遇到错误，会在响应中包含响应的错误码描述错误信息。对象存储访问服务的错误码如表 6-2 所示。

表 6-2 错误码

状态码	错误码	错误信息	处理措施
301 Moved Permanently	PermanentRedirect	尝试访问的桶必须使用指定的地址，请将以后的请求发送到这个地址。	按照返回的重定向地址发送请求。
301 Moved Permanently	WebsiteRedirect	Website 请求缺少 bucketName。	携带桶名后重试。
307 Moved Temporarily	TemporaryRedirect	临时重定向，当 DNS 更新时，请求将被重定向到桶。	会自动重定向，也可以将请求发送到重定向地址。
400 Bad Request	BadDigest	客户端指定的对象内容的 MD5 值与系统接收到的内容 MD5 值不一致。	检查头域中携带的 MD5 与消息体计算出来的 MD5 是否一致。
400 Bad Request	BadDomainName	域名不合法。	使用合法的域名。
400 Bad Request	BadRequest	请求参数不合法。	根据返回的错误消息体提示进行修改。
400 Bad Request	CustomDomainAlreadyExist	配置了已存在的域。	已经配置过了，不需要再配置。
400 Bad Request	CustomDomainNotExist	删除不存在的域。	未配置或已经删除，无需删除。
400 Bad Request	EntityTooLarge	• 通过 SDK 或 API 的 PUT 上传、POST 上传和追加写超过 5GB 的文件 • 上传段超过 5GB • 设置的桶配置超过 20KB • 超过 post 表单中 policy 限	修改上传的 policy 中的条件或者减少对象大小。

状态码	错误码	错误信息	处理措施
		制的文件大小（用户可自定义） • 通过 SDK 或 API 的多段上传以及 SDK 的断点续传超过 48.8TB 的文件	
400 Bad Request	EntityTooSmall	• 上传段除最后一段小于 100KB • 小于 post 表单中 policy 限制的文件大小（用户可自定义）	修改上传的 policy 中的条件或者增加对象大小。
400 Bad Request	IllegalLocationConstraintException	用户未带 Location 在非默认 Region 创建桶。	请求发往默认 Region 创建桶或带非默认 Region 的 Location 创建桶。
400 Bad Request	IncompleteBody	由于网络原因或其他问题导致请求体未接收完整。	重新上传对象。
400 Bad Request	IncorrectNumberOfFilesInPostRequest	每个 POST 请求都需要带一个上传的文件。	带上一个上传文件。
400 Bad Request	InvalidArgument	无效的参数。	根据返回的错误消息体提示进行修改。
400 Bad Request	InvalidBucket	请求访问的桶已不存在。	更换桶名。
400 Bad Request	InvalidBucketName	请求中指定的桶名无效，超长或带不允许的特殊字符。	更换桶名。
400 Bad Request	InvalidContentLength	Content-Length 头域内容有误。	请检查封装头域或联系技术支持。
400 Bad Request	InvalidDefaultStorageClass	存储类别不可用。	请确认能够使用的存储类别。
400 Bad Request	InvalidEncryptionAlgorithmError	错误的加密算法。下载 SSE-C 加密的对象，携带的加密头域错误，导致不能解密。	携带正确的加密头域下载对象。
400 Bad Request	InvalidLocationConstraint	创建桶时，指定的 Location 不合法或不存在。	指定正确的 Location 创建桶。
400 Bad Request	InvalidPart	一个或多个指定的段无法找到。这些段可能没有上传，或者指定的 entity tag 与段的 entity tag 不一致。	按照正确的段和 entity tag 合并段。
400 Bad Request	InvalidPartOrder	段列表的顺序不是升序，段列表必须按段号升序排列。	按段号升序排列后重新合并。

状态码	错误码	错误信息	处理措施
400 Bad Request	InvalidPolicyDocument	表单中的内容与策略文档中指定的条件不一致。	根据返回的错误消息体提示修改构造表单的 policy 重试。
400 Bad Request	InvalidRedirectLocation	无效的重定向地址。	指定正确的地址。
400 Bad Request	InvalidRequest	无效请求。	根据返回的错误消息体提示进行修改。
400 Bad Request	InvalidRequestBody	请求体无效，需要消息体的请求没有上传消息体。	按照正确的格式上传消息体。
400 Bad Request	InvalidTargetBucketForLogging	delivery group 对目标桶无 ACL 权限。	对目标桶配置 ACL 权限后重试。
400 Bad Request	KeyTooLongError	提供的 Key 过长。	使用较短的 Key。
400 Bad Request	KMS.DisabledException	SSE-KMS 加密方式下，主密钥被禁用。	更换密钥后重试，或联系技术支持。
400 Bad Request	KMS.NotFoundException	SSE-KMS 加密方式下，主密钥不存在。	携带正确的主密钥重试。
400 Bad Request	MalformedACLError	提供的 XML 格式错误，或者不符合要求的格式。	使用正确的 XML 格式重试。
400 Bad Request	MalformedError	请求中携带的 XML 格式不正确。	使用正确的 XML 格式重试。
400 Bad Request	MalformedLoggingStatus	Logging 的 XML 格式不正确。	使用正确的 XML 格式重试。
400 Bad Request	MalformedPolicy	Bucket policy 检查不通过。	根据返回的错误消息体提示结合桶 policy 的要求进行修改。
400 Bad Request	MalformedQuotaError	Quota 的 XML 格式不正确。	使用正确的 XML 格式重试。
400 Bad Request	MalformedXML	当用户发送了一个配置项的错误格式的 XML 会出现这样的错误。	使用正确的 XML 格式重试。
400 Bad Request	MaxMessageLengthExceeded	拷贝对象，带请求消息体。	拷贝对象不带消息体重试。
400 Bad Request	MetadataTooLarge	元数据消息头超过了允许的最大元数据大小。	减少元数据消息头。
400 Bad Request	MissingRegion	请求中缺少 Region 信息，且系统无默认 Region。	请求中携带 Region 信息。
400 Bad Request	MissingRequestBodyError	当用户发送一个空的 XML 文	提供正确的 XML 文档。

状态码	错误码	错误信息	处理措施
		档作为请求时会发生。	
400 Bad Request	MissingRequiredHeader	请求中缺少必要的头域。	提供必要的头域。
400 Bad Request	MissingSecurityHeader	请求缺少一个必须的头。	提供必要的头域。
400 Bad Request	MultipleContentLengths	多个 Content-Length 头域。	请检查封装头域或者联系技术支持。
400 Bad Request	TooManyBuckets	用户拥有的桶的数量达到了系统的上限，并且请求试图创建一个新桶。	删除部分桶后重试。
400 Bad Request	TooManyCustomDomains	配置了过多的用户域名。	删除部分用户域名后重试。
400 Bad Request	TooManyWrongSignature	因高频错误请求被拒绝服务。	更换正确的 Access Key 后重试。
400 Bad Request	UnexpectedContent	该请求需要消息体而客户端没带，或该请求不需要消息体而客户端带了。	根据说明重试。
400 Bad Request	UserKeyMustBeSpecified	该操作只有特殊用户可使用。	请联系技术支持。
400 Bad Request	FileGatewayBucket	并行文件系统不支持配置桶清单。	请使用对象桶。
400 Bad Request	ForbidAnonReqOverwriteRespHead	匿名用户调用下载对象接口时，不能重写响应头。	请删除重写响应头参数，或者携带签名调用下载对象接口。
403 Forbidden	AccessDenied	拒绝访问，请求没有携带日期头域或者头域格式错误。	请求携带正确的日期头域。
403 Forbidden	AccessDenied	试图修改或删除受 WORM 保护的对象。	等待 WORM 保护过期后再修改或删除。
403 Forbidden	AccessForbidden	权限不足，桶未配置 CORS 或者 CORS 规则不匹配。	修改桶的 CORS 配置，或者根据桶的 CORS 配置发送匹配的 OPTIONS 请求。
403 Forbidden	AllAccessDisabled	用户无权限执行某操作。桶名为禁用关键字。	更换桶名。
403 Forbidden	DeregisterUserId	用户已经注销。	充值或重新开户。
403 Forbidden	InArrearOrInsufficientBalance	用户欠费或余额不足而没有权限进行某种操作。	充值。
403 Forbidden	InsufficientStorag	存储空间不足。	超过配额限制，增加配额或删

状态码	错误码	错误信息	处理措施
	eSpace		除部分对象。
403 Forbidden	InvalidAccessKeyId	系统记录中不存在客户提供的 Access Key Id。	携带正确的 Access Key Id。
403 Forbidden	InvalidObjectState	• 归档存储对象需要先进行恢复才能下载、复制、修改元数据。 • 恢复的对象不是归档存储对象。	• 先对对象进行恢复操作。 • 确认操作对象是归档存储对象。
403 Forbidden	NotSignedUp	您的账户还没有在系统中注册，必须先在系统中注册了才能使用该账户。	先注册 OBS 服务。
403 Forbidden	RequestTimeTooSkewed	客户端发起请求的时间与 OBS 服务端的时间相差太大。 出于安全目的，OBS 会校验客户端与 OBS 服务端的时间差，当该时间差大于 15 分钟时，OBS 服务端会拒绝您的请求，从而出现此报错。	请检查客户端时间是否与当前 OBS 服务端时间相差太大。请根据本地 UTC 时间调整客户端时间后再访问。
403 Forbidden	SignatureDoesNotMatch	请求中带的签名与系统计算得到的签名不一致。	检查您的 Secret Access Key 和签名计算方法。
403 Forbidden	VirtualHostDomainRequired	未使用虚拟主机访问域名。	Host 使用 3.1 构造请求。
403 Forbidden	Unauthorized	用户未实名认证。	请实名认证后重试。
404 Not Found	NoSuchBucket	指定的桶不存在。	先创建桶再操作。
404 Not Found	NoSuchBucketPolicy	桶 policy 不存在。	先配置桶 policy。
404 Not Found	NoSuchCORSConfiguration	CORS 配置不存在。	先配置 CORS。
404 Not Found	NoSuchCustomDomain	请求的用户域不存在。	先设置用户域。
404 Not Found	NoSuchKey	指定的 Key 不存在。	先上传对象。
404 Not Found	NoSuchLifecycleConfiguration	请求的 LifeCycle 不存在。	先配置 LifeCycle。
404 Not Found	NoSuchUpload	指定的多段上传不存在。Upload ID 不存在，或者多段上传已经终止或完成。	使用存在的段或重新初始化段。
404 Not Found	NoSuchVersion	请求中指定的 version ID 与现	使用正确的 version ID。

状态码	错误码	错误信息	处理措施
		存的所有版本都不匹配。	
404 Not Found	NoSuchWebsiteConfiguration	请求的 Website 不存在。	先配置 Website。
405 Method Not Allowed	MethodNotAllowed	指定的方法不允许操作在请求的资源上。 对应返回的 Message 为：Specified method is not supported.	方法不允许。
405 Method Not Allowed	FsNotSupport	并行文件系统不支持该 API。	方法不允许。
408 Request Timeout	RequestTimeout	用户与 Server 之间的 socket 连接在超时时间内没有进行读写操作。	检查网络后重试，或联系技术支持。
409 Conflict	BucketAlreadyExists	请求的桶名已经存在。桶的命名空间是系统中所有用户共用的，选择一个不同的桶名再重试一次。	更换桶名。
409 Conflict	BucketAlreadyOwnedByYou	发起该请求的用户已经创建过了这个名字的桶，并拥有这个桶。	不需要再创建桶了。
409 Conflict	BucketNotEmpty	用户尝试删除的桶不为空。	先删除桶中对象，然后再删桶。
409 Conflict	InvalidBucketState	无效的桶状态，配置跨 Region 复制后不允许关闭桶多版本。	不关闭桶的多版本或取消跨 Region 复制。
409 Conflict	OperationAborted	另外一个冲突的操作当前正作用在这个资源上，请重试。	等待一段时间后重试。
409 Conflict	ServiceNotSupported	请求的方法服务端不支持。	服务端不支持，请联系技术支持。
409 Conflict	FsObjectConflict	并行文件系统中，文件和目录不支持互相覆盖，或者重命名的目标文件已经存在。	请确认待覆盖的目标类型（是文件还是目录），请确认重命名的目标文件是否存在。
409 Conflict	FsRenameConflict	重命名操作时，另外一个冲突的操作当前正作用在这个对象上	排查业务逻辑，避免对同一个对象做并发操作导致操作结果不被预期
409 Conflict	DirectoryNotEmpty	非空目录对象不能被直接删除	先清空此目录对象下的全部对象，再删除这个目录对象
409 ObjectNotAppendable	ObjectNotAppendable	该对象不支持追加上传	请确认桶类型，并行文件系统不支持追加上传。请确认对象

状态码	错误码	错误信息	处理措施
			类型，归档存储对象不支持追加上传。
411 Length Required	MissingContentLength	必须要提供 HTTP 消息头中的 Content-Length 字段。	提供 Content-Length 消息头。
412 Precondition Failed	PreconditionFailed	用户指定的先决条件中至少有一项没有包含。	根据返回消息体中的 Condition 提示进行修改。
414 URI Too Long	Request-URI Too Large	请求使用的 URI 过长	请减少 URI 的长度。
416 Client Requested Range Not Satisfiable	InvalidRange	请求的 range 不可获得。	携带正确的 range 重试。
429 Too Many Requests	TooManyRequests	短时间内请求量过多。 例如针对同一个对象/桶的元数据并发请求量过多，包括并发修改对象元数据、上传对象、修改写对象等接口。	请减少请求量或并发数。
500 Internal Server Error	InternalError	系统遇到内部错误，请重试。	请联系技术支持。
501 Not Implemented	ServiceNotImplemented	请求的方法服务端没有实现。	当前不支持，请联系技术支持。
503 Service Unavailable	ServiceUnavailable	服务器过载或者内部错误异常。	等待一段时间后重试，或联系技术支持。
503 Service Unavailable	SlowDown	高并发请求导致访问异常	请降低请求频率。

7 IAM 策略和授权项

7.1 IAM 策略及授权项说明

如果您需要对您所拥有的对象存储服务（OBS）进行精细的权限管理，您可以使用统一身份认证服务（Identity and Access Management，简称 IAM），如果当前的账号已经能满足您的要求，不需要创建独立的 IAM 用户，您可以跳过本章节，不影响您使用 OBS 的其它功能。

默认情况下，新建的 IAM 用户没有任何权限，您需要将其加入用户组，并给用户组授予 IAM 策略，才能使用户组中的用户获得 IAM 策略定义的权限，这一过程称为授权。授权后，用户就可以基于 IAM 策略对云服务进行操作。

关于 IAM 中和 OBS 相关的用户策略介绍，请参见《对象存储服务用户指南》的“产品介绍 > 权限管理”章节。关于 IAM 策略的语法结构及示例，请参见《对象存储服务用户指南》中的“控制台指南 > 权限控制 > 权限控制方式介绍 > IAM 策略”章节。

策略根据授权精度分为细粒度策略和 RBAC 策略。RBAC 策略是将服务作为一个整体进行授权，授权后，用户可以拥有这个服务的所有权限。细粒度策略以 API 接口为粒度进行权限拆分，授权更加精细，可以精确到某个操作。

说明

- 如果您要允许或是禁止某个接口的操作权限，请使用细粒度策略。
- 由于缓存的存在，对用户、用户组以及企业项目授予 OBS 相关的 RBAC 策略后，大概需要等待 10~15 分钟 RBAC 策略才能生效；授予 OBS 相关的细粒度策略后，大概需要等待 5 分钟细粒度策略才能生效。

账号具备所有接口的调用权限，如果使用账号下的 IAM 用户发起 API 请求时，该 IAM 用户必须具备调用该接口所需的权限，否则，API 请求将调用失败。每个接口所需要的权限，与各个接口所对应的授权项相对应，只有发起请求的用户被授予授权项所对应的策略，该用户才能成功调用该接口。例如，用户要调用接口来创建桶，那么这个 IAM 用户被授予的策略中必须包含允许“obs:bucket:CreateBucket”的授权项，该接口才能调用成功。

支持的授权项

细粒度策略支持的操作与 API 相对应，授权项列表说明如下：

- 权限：自定义策略中授权项定义的内容即为权限。
- 对应 API 接口：自定义策略实际调用的 API 接口。
- 授权项：自定义策略中支持的 Action，在自定义策略中的 Action 中写入授权项，可以实现授权项对应的权限功能。
- IAM 项目（Project）/企业项目（Enterprise Project）：自定义策略的授权范围，包括 IAM 项目与企业项目。授权范围如果同时支持 IAM 项目和企业项目，表示此授权项对应的自定义策略可以在 IAM 和企业管理两个服务中给用户组授权并生效。如果仅支持 IAM 项目，不支持企业项目，表示仅能在 IAM 中给用户组授权并生效，如果在企业管理中授权，则该自定义策略不生效。

说明

“√”表示支持，“x”表示暂不支持。

OBS 支持的自定义策略授权项如下所示：

- 7.2 桶相关授权项：包括 OBS 所有面向桶的接口所对应的授权项，如列举全部桶、创建桶、删除桶、设置桶策略、设置桶的日志记录、设置桶的事件通知、设置桶的跨区域复制配置等接口。
- 7.3 对象相关授权项：包括上传对象、下载对象、删除对象等接口。

7.2 桶相关授权项

表 7-1 桶相关授权项列表

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
列举全部桶	5.1.1 获取桶列表	obs:bucket:ListAllMyBuckets	√	√
创建桶	5.1.2 创建桶	obs:bucket:CreateBucket	√	√
列举桶内对象	5.1.3 列举桶内对象	obs:bucket:ListBucket	√	√
列举桶内多版本对象	5.1.3 列举桶内对象	obs:bucket:ListBucketVersions	√	√
判断桶是否存在并	5.1.4 获取桶元数	obs:bucket:HeadBuc	√	√

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
获取桶的元数据	据	ket		
获取桶位置	5.1.5 获取桶区域位置	obs:bucket:GetBucketLocation	√	√
删除桶	5.1.6 删除桶	obs:bucket:DeleteBucket	√	√
设置桶策略	5.2.1 设置桶策略	obs:bucket:PutBucketPolicy	√	√
获取桶策略配置的相关信息	5.2.2 获取桶策略	obs:bucket:GetBucketPolicy	√	√
删除桶策略	5.2.3 删除桶策略	obs:bucket:DeleteBucketPolicy	√	√
设置桶 ACL	5.2.4 设置桶 ACL	obs:bucket:PutBucketAcl	√	√
获取桶 ACL 的相关信息	5.2.5 获取桶 ACL	obs:bucket:GetBucketAcl	√	√
设置桶日志记录	5.2.6 设置桶日志管理配置	obs:bucket:PutBucketLogging	√	√
获取桶日志记录的相关信息	5.2.7 获取桶日志管理配置	obs:bucket:GetBucketLogging	√	√
设置和删除桶生命周期规则	5.2.8 设置桶的生命周期配置 5.2.10 删除桶的生命周期配置	obs:bucket:PutLifecycleConfiguration	√	√
获取桶生命周期规则	5.2.9 获取桶的生命周期配置	obs:bucket:GetLifecycleConfiguration	√	√
设置多版本	5.2.11 设置桶的多版本状态	obs:bucket:PutBucketVersioning	√	√
获取桶多版本的相关信息	5.2.12 获取桶的多版本状态	obs:bucket:GetBucketVersioning	√	√
设置桶默认存储类型	5.2.13 设置桶默认存储类型	obs:bucket:PutBucketStoragePolicy	√	√
获取桶默认存储类	5.2.14 获取桶默认	obs:bucket:GetBuck	√	√

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
型	存储类型	etStoragePolicy		
设置桶的跨区域复制配置	5.2.15 设置桶的跨区域复制配置	obs:bucket:PutReplicationConfiguration	√	√
获取桶的跨区域复制配置	5.2.16 获取桶的跨区域复制配置	obs:bucket:GetReplicationConfiguration	√	√
删除桶的跨区域复制配置	5.2.17 删除桶的跨区域复制配置	obs:bucket:DeleteReplicationConfiguration	√	√
设置桶标签	5.2.18 设置桶标签	obs:bucket:PutBucketTagging	√	√
获取桶标签	5.2.19 获取桶标签	obs:bucket:GetBucketTagging	√	√
删除桶标签	5.2.20 删除桶标签	obs:bucket:DeleteBucketTagging	√	√
设置桶配额	5.2.21 设置桶配额	obs:bucket:PutBucketQuota	√	√
获取桶配额	5.2.22 获取桶配额	obs:bucket:GetBucketQuota	√	√
获取桶存量信息	5.2.23 获取桶存量信息	obs:bucket:GetBucketStorage	√	√
设置桶清单	5.2.24 设置桶清单	obs:bucket:PutBucketInventoryConfiguration	√	√
获取和列举桶清单	5.2.25 获取桶清单 5.2.26 列举桶清单	obs:bucket:GetBucketInventoryConfiguration	√	√
删除桶清单	5.2.27 删除桶清单	obs:bucket:DeleteBucketInventoryConfiguration	√	√
设置桶的自定义域名	5.2.28 设置桶的自定义域名	obs:bucket:PutBucketCustomDomainConfiguration	√	√
获取桶的自定义域名	5.2.29 获取桶的自定义域名	obs:bucket:GetBucketCustomDomainConfiguration	√	√

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
删除桶的自定义域名	5.2.30 删除桶的自定义域名	obs:bucket:DeleteBucketCustomDomainConfiguration	√	√
设置和删除桶的加密配置	5.2.31 设置桶的加密配置 5.2.33 删除桶的加密配置	obs:bucket:PutEncryptionConfiguration	√	√
获取桶的加密配置	5.2.32 获取桶的加密配置	obs:bucket:GetEncryptionConfiguration	√	√
设置桶归档存储对象直读策略	5.2.34 设置桶归档存储对象直读策略	obs:bucket:PutDirectColdAccessConfiguration	√	√
获取桶归档存储对象直读策略	5.2.35 获取桶归档存储对象直读策略	obs:bucket:GetDirectColdAccessConfiguration	√	√
删除桶归档存储对象直读策略	5.2.36 删除桶归档存储对象直读策略	obs:bucket:DeleteDirectColdAccessConfiguration	√	√
设置桶的静态网站托管	5.3.1 设置桶的网站配置	obs:bucket:PutBucketWebsite	√	√
获取桶的静态网站配置的相关信息	5.3.2 获取桶的网站配置	obs:bucket:GetBucketWebsite	√	√
删除桶的静态网站托管配置	5.3.3 删除桶的网站配置	obs:bucket:DeleteBucketWebsite	√	√
设置和删除桶 CORS	5.3.4 设置桶的 CORS 配置 5.3.6 删除桶的 CORS 配置	obs:bucket:PutBucketCORS	√	√
获取桶 CORS 配置的相关信息	5.3.5 获取桶的 CORS 配置	obs:bucket:GetBucketCORS	√	√
配置桶级默认 WORM 策略	5.2.37 配置桶级默认 WORM 策略	obs:bucket:PutBucketObjectLockConfiguration	√	√
获取桶级默认	5.2.38 获取桶级默	obs:bucket:GetBuck	√	√

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
WORM 策略	认 WORM 策略	etObjectLockConfiguration		
列举桶中已初始化多段任务	5.5.1 列举桶中已初始化多段任务	obs:bucket:ListBucketMultipartUploads	√	√

7.3 对象相关授权项

表 7-2 对象相关授权项列表

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
可用作于 PUT 上传对象，POST 上传对象，追加写对象，初始化上传段任务，上传段，拷贝段，合并段，修改写对象，截断对象，重命名对象	5.4.1 PUT 上传 5.4.2 POST 上传 5.4.9 追加写对象 5.5.2 初始化上传段任务 5.5.3 上传段 5.5.6 合并段 5.4.13 修改写对象 5.4.14 截断对象 5.4.15 重命名对象	obs:object:PutObject	√	√
复制对象	5.4.3 复制对象	obs:object:GetObject obs:object:PutObject	√	√
获取对象内容，对象元数据	5.4.4 下载对象 5.4.5 获取对象元数据	obs:object:GetObject	√	√
获取指定版本对象	5.4.4 下载对象	obs:object:GetObjec	√	√

权限	对应 API 接口	授权项（Action）	IAM 项目（Project）	企业项目（Enterprise Project）
内容和对象元数据	5.4.5 获取对象元数据	tVersion		
单个删除和批量删除对象	5.4.6 删除对象 5.4.7 批量删除对象	obs:object:DeleteObject	√	√
单个删除和批量删除指定版本对象	5.4.6 删除对象 5.4.7 批量删除对象	obs:object:DeleteObjectVersion	√	√
恢复归档存储对象	5.4.8 恢复归档存储对象	obs:object:RestoreObject	√	√
设置对象 ACL	5.4.10 设置对象 ACL	obs:object:PutObjectAcl	√	√
设置指定版本对象 ACL	5.4.10 设置对象 ACL	obs:object:PutObjectVersionAcl	√	√
获取对象 ACL 的相关信息	5.4.11 获取对象 ACL	obs:object:GetObjectAcl	√	√
获取指定版本对象 ACL 的相关信息	5.4.11 获取对象 ACL	obs:object:GetObjectVersionAcl	√	√
修改对象元数据	5.4.12 修改对象元数据	obs:object:ModifyObjectMetaData	√	√
列举已上传段	5.5.5 列举已上传未合并的段	obs:object:ListMultipartUploadParts	√	√
取消多段上传任务	5.5.7 取消多段上传任务	obs:object:AbortMultipartUpload	√	√
配置对象级 WORM 保护策略	5.4.16 配置对象级 WORM 保护策略	obs:object:PutObjectRetention	√	√
获取对象级 WORM 保护策略	5.4.5 获取对象元数据	obs:object:GetObject obs:object:GetObjectRetention	√	√

8 附录

8.1 状态码

服务器向用户返回的状态码和提示信息如表 8-1 所示：

表 8-1 状态码

状态码	说明
2xx	服务器成功返回用户请求的数据。
4xx	客户端发出的请求有错误，服务器没有进行新建或修改数据的操作。
5xx	服务器发生错误，用户将无法判断发出的请求是否成功。

说明

注：请使用符合 https://www.ietf.org/rfc/rfc2616.txt 规定的 HTTP/HTTPS 请求格式发送 API 请求。

8.2 获取访问密钥（AK/SK）

在调用接口的时候，需要使用 AK/SK 进行签名验证。AK/SK 获取步骤如下：

步骤 1 登录控制台。

步骤 2 鼠标指向界面右上角的登录用户名，在下拉列表中单击“我的凭证”。

步骤 3 在左侧导航栏单击“访问密钥”。

步骤 4 单击“新增访问密钥”，进入“新增访问密钥”页面。

步骤 5 输入访问密钥描述信息（非必填），单击“确定”。

步骤 6 通过邮箱或者虚拟 MFA 进行验证，输入对应的验证码，单击“确定”。

步骤 7 单击“立即下载”，浏览器下载访问密钥。

说明

为防止访问密钥泄露，建议您将其保存到安全的位置。

----结束

8.3 获取账号、IAM 用户、项目、用户组、区域、委托的名称和 ID

获取账号、IAM 用户、项目的名称和 ID

- **从控制台获取账号名、账号 ID、用户名、用户 ID、项目名称、项目 ID**
  a. 在系统首页右上角，单击“控制台”。
  b. 在右上角的用户名中选择“我的凭证”。
  c. 在“我的凭证”界面，API 凭证页签中，查看账号名、账号 ID、用户名、用户 ID、项目名称、项目 ID。

     每个区域的项目 ID 有所不同，需要根据业务所在的区域获取对应的项目 ID。
- **调用 API 获取用户 ID、项目 ID**
  - 获取用户 ID 请参考《统一身份认证服务接口参考》的“管理员查询 IAM 用户列表”章节的内容。
  - 获取项目 ID 请参考《统一身份认证服务接口参考》的“查询指定条件下的项目列表”章节的内容。

获取用户组名称和 ID

步骤 1 登录系统，进入 IAM 控制台，选择“用户组”页签。

步骤 2 单击需要查询的用户组前的下拉框，即可查询用户组名称、用户组 ID。

----结束

获取区域 ID

步骤 1 登录系统，进入 IAM 控制台，选择“项目”页签。

步骤 2 “项目”列的内容即为所属区域对应的 ID。

----结束

获取委托名称和 ID

步骤 1 登录系统，进入 IAM 控制台，选择“委托”页签。

步骤 2 鼠标移动到需要查询名称和 ID 的委托上，黑色框中出现的第一行为委托名称，第二行为委托 ID。

----结束

8.4 并发一致性说明

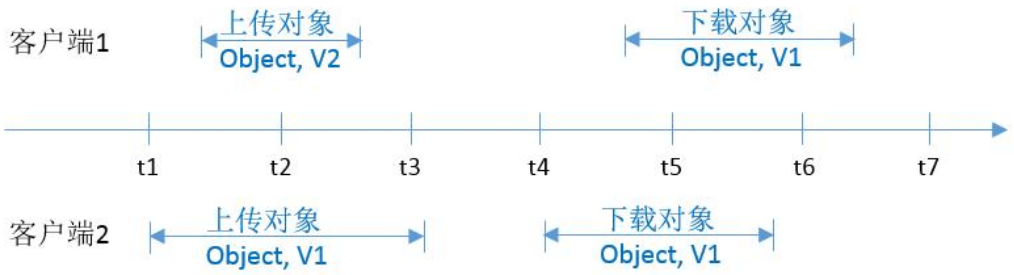
当客户端发起的写/删除请求返回成功之后，客户端可以获取到最新数据。当写操作客户端等待超时、服务端返回 500 或者 503 的 HTTP 响应错误码时，之后的读取操作有可能成功读取到数据，也有可能读不到数据。建议客户端在出现上述错误时，查询数据是否已经上传成功，如果不成功则重新上传。

针对同一个对象或桶的操作，比如多个客户端对同一个对象并行上传、查询和删除时，具体操作结果依赖于操作到达系统的时间和系统内部处理的时延，可能返回不一致的结果。比如，当多个客户端并行上传同一个对象时，系统最后收到的上传请求会覆盖前一个上传的对象。如果需要避免同一个对象被并行访问，需要在上层应用中增加对象的锁机制。

并发操作举例

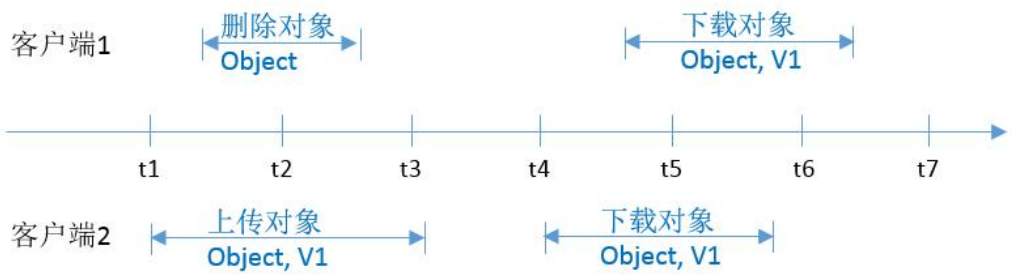
1. 当客户端 2 正在上传一个对象 v1 时，客户端 1 同时上传一个同名的对象 v2 成功后，不管是客户端 1 还是客户端 2 都能够读取最新的对象数据 v1，如图 8-1 所示。

图 8-2 并发成功上传同一个对象



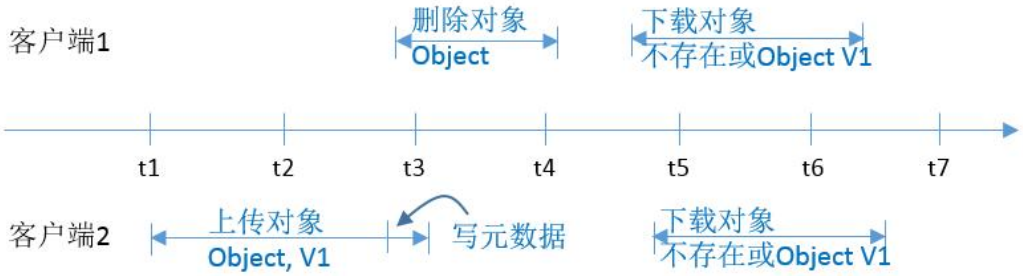
2. 当客户端 2 上传一个对象 v1 的时候，如果在对象数据上传且还没有写入对象元数据的过程中，客户端 1 删除同名的对象成功后，客户端 2 的上传对象操作仍然返回成功，并且不论客户端 1 还是客户端 2 都能读取到对象数据 v1，如图 8-2 所示。

图 8-3 并发上传和删除同一个对象（1）



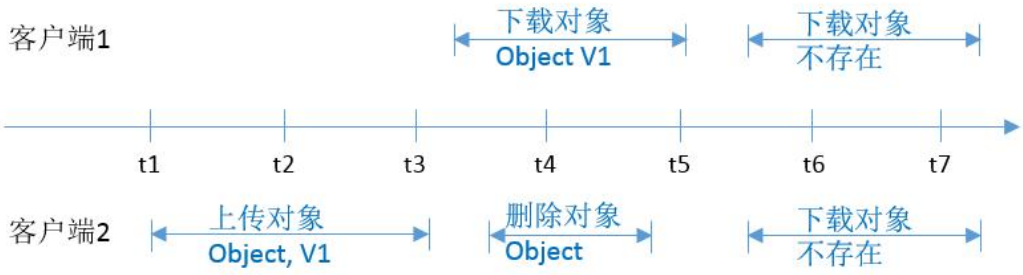
3. 当客户端 2 上传一个对象 v1 的时候，如果在对象数据上传完成，系统写入对象元数据的短暂过程中，客户端 1 发起删除同名的对象成功后，客户端 2 的上传对象操作仍然返回成功，但是客户端 1 和客户端 2 下载对象 Object1 时，有可能读到对象数据 v1，也有可能返回对象不存在，如图 8-3 所示。

图 8-4 并发上传和删除同一个对象（2）



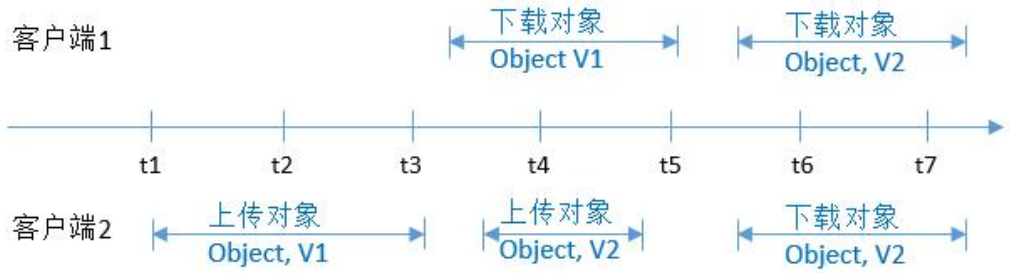
4. 当客户端 1 下载一个对象的过程中，客户端 2 发起删除同名对象操作，此时客户端 1 可能下载到完整的对象数据，也有可能只能下载到对象的一部分数据。当客户端 2 的删除操作返回成功后，再发起下载对象请求，则返回对象不存在，如图 8-4 所示。

图 8-5 并发下载和删除同一个对象



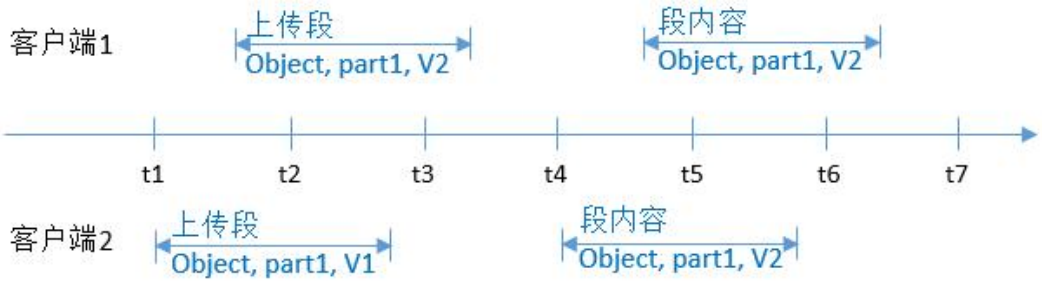
5. 当客户端 1 下载一个对象的过程中，客户端 2 发起更新同名对象操作，此时客户端 1 可能下载到完整的对象数据，也有可能只能下载到对象的一部分数据。当客户端 2 的更新操作返回成功后，再发起下载对象请求，则返回最新的对象数据，如图 8-5 所示。

图 8-6 并发下载和更新同一个对象



6. 当客户端 2 正在上传一个对象的段 v1 时，客户端 1 同时上传同一个对象的相同段号的段 v2 成功后，不管是客户端 1 还是客户端 2 列举段时都能够列举 etag 为 v2 的段信息，如图 8-6 所示。

图 8-7 并发上传同名对象的同名段



A 修订记录

发布日期	修订记录
2026-02-24	第七次正式发布。 本次更新说明如下： 新增 5.4.9 追加写对象。
2025-05-15	第六次正式发布。 本次更新说明如下： 更新：1.1 概述，新增使用 TLS1.0/TLS1.1 协议版本时的风险提示。
2024-04-11	第五次正式发布。 本次更新说明如下： • 新增存储类别相关接口。 • 新增归档对象直读策略相关接口。 • 新增跨区域复制相关接口。 • 新增桶标签相关接口。 • 新增桶清单相关接口。 • 新增服务端加密相关接口。 • 新增 WORM 策略相关接口。
2023-06-21	第四次正式发布。 本次更新说明如下： • 新增静态网站托管相关接口。 • 新增桶配额相关接口。
2022-08-11	第三次正式发布。 • 新增自定义域名相关接口。
2021-10-21	第二次正式发布。

发布日期	修订记录
	• 新增并行文件系统相关接口。 • 新增图片处理相关说明。
2020-06-15	第一次正式发布。