天翼云媒体存储

常见问题与最佳实践

2025-12-18

天翼云科技有限公司

目录

1. 常见问题

1.1 一般性问题

1.1.1 什么是媒体存储

媒体存储（CT-XStor，原对象存储融合版）是天翼云基于分布式数据存储和媒体处理技术，为客户提供海量视频、图片及其他非结构化数据存取与处理的云服务，具有弹性灵活、安全可靠、高性价比等优点，支持多种存储协议及多档资源类型，深度匹配各类行业场景应用。

更多关于媒体存储介绍可参考：产品定义 。

1.1.2 媒体存储有哪些适用场景

媒体存储重点适用在视频监控、医疗影像、媒体资源管理等业务领域中海量非结构化数据上云需求。具体可参考：应用场景。

1.1.3 媒体存储的开放范围

在天翼云官网，媒体存储支持订购对象存储-标准型，计费方式为按需计费。具体开通方式可参考：订购指引。

其它能力（如：对象存储-低频型、文件存储、块存储等），暂不面向预付费客户开放，如需使用其他产品能力，如有需要请致电服务热线：400-810-9889。

1.1.4 媒体存储的持久性和可用性

天翼云媒体存储承诺标准型存储业务的可用性在 99.99%以上，低频访问型存储业务的可用性在 99.9%以上。

1.1.5 媒体存储的数据存储在哪里

在媒体存储上新建桶、块空间或文件空间时，您可以指定一个区域的资源池，您的数据将存储在该资源池的设备上。

您可以参考以下操作手册，在新建对应的存储空间时选择资源池：

新建桶可参考：新建 Bucket。

新建文件空间可参考：新建文件空间。

新建块空间可参考：新建块空间。

1.1.6 如何选择将数据存储在哪个区域

选择区域时，您可考虑以下的因素：

- 地理位置：一般情况下，我们推荐您遵循就近原则，选择靠近您或者您的目标用户的区域，这样可以减少网络时延，提高访问速度。
- 云服务之间的关系，如果多个天翼云的服务一起搭配使用，需要注意明确是否有内网互通的需求，不同区域的弹性云服务器、媒体存储服务内网不互通。

1.1.7 媒体存储有哪些产品类型

媒体存储支持对象存储、块存储与文件存储。

不同产品类型支持的使用方式及使用说明如下：

产品类型	使用方式	使用说明
对象存储	媒体存储控制台、API 接口（兼容标准 S3 协议）、其他第三方工具如 S3Browser、S3cmd 或 AWSCli 等。	对象存储
块存储	标准 iSCSI 协议挂载后使用。	块存储
文件存储	标准 NFS、CIFS 协议挂载后使用。	文件存储

1.1.8 对象存储的适用场景

媒体存储提供支持标准 S3 协议的对象存储服务，基于对象存储扁平化特性，可支持海量数据的读写，可用以存储非结构化数据，如网站与应用中的附件、图片、音视频、文本等格式文件等。

客户可通过专线、云上内网、互联网等多种方式接入。适合图片、视频等海量文件的存储、网页静态和动态资源分离、云端数据处理等。

您可参考用户指南，获取更多对象存储的使用介绍：用户指南。

1.1.9 块存储的适用场景与使用限制

媒体存储提供支持标准 iSCSI 协议的块存储服务，客户侧本地应用或云上应用均可对接，推荐通过专线接入或云上应用访问。

适用于低频、中频读写访问、业务弹性扩展、成本敏感型的存储场景。重点针对视频监控、医疗影像、媒资存储等场景推广，在此类场景中客户侧应用不支持改造或仅支持少量改造，且不具备条件对接文件存储时，可选择使用块存储。不适用于数据库、高性能计算等对 IO 吞吐要求较高的场景。

关于块存储的使用方式，您可参考：块存储。

1.1.10 文件存储的适用场景与使用限制

媒体存储提供的文件存储服务是基于 POSIX 文件接口，支持标准 NFS、CIFS 协议，客户侧本地应用或云上应用均可对接，可通过专线、云上内网方式接入，支持直连模式。

适用于低频、中频读写访问、业务弹性扩展、成本敏感型的文件共享存储场景。重点针对视频监控、医疗影像、媒资存储等场景推广。在此类场景中如客户侧应用不支持改造或仅支持少量改造时，优先推荐使用文件存储；不适合数据库、高性能计算等对 IO 吞吐要求较高的场景。

关于文件存储的使用方式，您可参考：文件存储。

1.1.11 媒体存储是否支持通过 HTTPS 访问

产品中的对象存储支持通过 HTTPS 访问。

可直接在浏览器中输入 https:+桶域名/自定义域名进行访问，或在使用对媒体存储控制台分配的域名进行访问时，直接在浏览器中将桶或对象的 URL 的协议头从 http 替换成 https 即可。

具体 HTTPS 访问地址的获取方式，可参考基础信息查看。

1.1.12 桶名和域名之间的关系

在媒体存储-对象存储服务中，您可在新建 Bucket 时指定一个存储桶名称，用以唯一标识该区域内的一个 Bucket。

域名，这里指 Endpoint，是媒体存储为每个存储区域提供一个区域节点，即为该区域的访问域名，用以进行公网访问，不同存储区域的访问域名不同。

您在某一存储区域创建存储桶后，可以通过 BucketName.Endpoint 的方式访问，其中 BucketName 为桶名称，Endpoint 为桶所在存储区域的访问域名。具体实例如下：

广东资源池 1 区的 Endpoint 为：gdoss.xstore.ctyun.cn；您创建了一个 Bucket，桶名为：doctest；则 Bucket 的访问域名为：doctest.gdoss.xstore.ctyun.cn 。

关于桶名和域名（Endpoint）的更多介绍，您可参考：访问规则。

关于如何获取 Bucket 的访问域名，您可参考：基础信息查看。

1.1.13 对象存储中的数据是否可以让其他用户访问

可以。

对于桶，您可以通过设置桶 ACL 和桶策略授予其他用户桶的数据读取权限，其他用户即可访问该桶。具体可参考：访问权限。

对于对象，您可以通过对象 ACL，对象策略和桶策略来授予其他用户对象的数据读取权限，或者通过临时 URL 方式，其他用户即可访问该对象。可参考：访问方式。

除此之外，您也可以通过 STS 角色，为临时用户颁发一个自定义时效和权限的访问凭证，使您区域粒度下的桶资源被更加安全地访问，参考：STS 角色管理。

1.1.14 产品中的对象存储，已删除的数据是否可以恢复

天翼云媒体存储无法恢复您主动删除、覆盖、配置规则自动删除或服务协议到期自动删除的数据，请您谨慎操作。

可能导致数据被删除或覆盖的场景：

- 通过控制台、API、SDK、XstorBrowser 方式删除对象。详情请参见删除对象。
- 通过控制台、API、SDK、XstorBrowser 方式上传同名文件到媒体存储，会导致桶内已有文件被覆盖。
- 若您在生命周期规则中配置了定期删除文件的规则，会根据生命周期的配置定期删除符合条件的文件。详情请参见生命周期。

- 若您配置了跨区域复制规则，且选择的是增/删/改同步，则对源存储空间（桶）进行文件修改或删除操作时，操作会同步到目标 Bucket。详情请参见存储桶复制。
- 没有正确的配置桶的访问权限，导致文件被他人恶意删除或覆盖。访问权限相关说明请参见访问权限-概述。
- 如果购买的存储资源到期后未及时续费，会为用户保留一段时间的数据。进入保留期后您在媒体存储中存储的数据会予以保留，对应资源会处于受限状态。保留期满仍未缴清欠款，存储在媒体存储中的数据将被销毁且无法恢复。详情请参见欠费说明。

您可通过以下方式防止误删除或误覆盖：

- 权限控制：正确使用媒体存储
- 提供的访问控制能力防止数据被删除或覆盖。详情请参见数据安全应用场景。
- 开启多版本控制：利用多版本控制，您可以在一个桶中保留多个版本的对象，使您更方便地检索和还原各个版本，在意外操作或应用程序故障时快速恢复数据。详情请参见版本控制。
- 跨区域复制到异地备份：您可以使用跨区域复制功能将数据复制到其他区域资源池进行备份。详情请参见存储桶复制。
- WORM 保护对象：您可以通过 WORM 功能保护对象，在保护期内阻止删除或覆盖对象。详情请参见合规保留。

1.1.15 产品中的对象存储，文件夹与文件系统的文件夹是否一样

不一样。对象存储服务中，并没有文件系统中的文件和文件夹概念。

为了使用户更方便进行管理数据，媒体存储提供了一种方式模拟文件夹：实际上是在对象的名称中增加“/”，并将该对象在媒体存储管理控制台上模拟成一个文件夹的形式进行展现。

1.1.16 媒体存储是否支持断点续传功能

目前控制台暂不支持主动的断点续传功能。

若您需要实现断点续传能力，请通过 SDK/API 进行文件分片上传，并在应用本地记录每个分片的上传情况作为上传的进度节点记录；当因网络波动等原因导致上传对象的过程被中断时，可在网络恢复后根据已上传的切片清单信息重新从此前的进度节点继续完成上传操作。

1.1.17 媒体存储是否提供图形化工具

媒体存储提供 XstorBrowser 图形化工具，本工具是对象存储图形化管理工具，提供类似 Windows 资源管理器的功能，支持完善桶管理和对象管理操作。

XstorBrowser 功能概述如下：

功能	描述
桶的基本操作	支持对象存储桶的基本操作，包括创建桶、查看桶的基本信息、碎片管理、删除桶等操作。
对象的基本操作	支持对象存储中对象的基本操作，包括新建文件夹、上传对象、列举对象、下载对象、删除对象等。
桶的 ACL 权限	支持修改桶的访问权限及添加其他用户对该桶的访问权限。
任务管理	主要是对执行的任务按状态进行展示，并提供暂停、删除、继续、重试等基本操作。

更多关于 XstorBrowser 的介绍，具体可参考：XstorBrowser。

1.1.18 产品中对象存储是否支持批量上传文件

媒体存储-对象存储服务的批量上传功能支持情况如下：

工具	批量上传
控制台	支持批量上传文件，单次最多支持 100 个文件同时上传，总大小建议不超过 2GB。详见：上传对象。
XstorBrowser	支持。详见：上传文件或文件夹。
SDK	不支持。
API	不支持。

1.1.19 产品中对象存储是否支持批量下载文件

媒体存储- 对象存储的批量下载功能支持情况如下：

工具	批量删除
控制台	不支持。
XstorBrowser	支持。详见：下载文件或文件夹。
SDK	不支持。
API	不支持。

1.1.20 产品中对象存储是否支持批量删除对象

媒体存储- 对象存储的批量删除功能支持情况如下：

工具	批量删除
控制台	目前只支持清空文件，即一键删除桶内所有对象。
XstorBrowser	支持。具体可参考：删除文件或文件夹。
SDK	支持。具体可参考：SDK 概览。
API	支持。具体可参考：删除对象。

1.1.21 为什么存储的数据丢失了

如您发现数据丢失，可参考以下原因进行排查：

- 请检查桶中是否设置了生命周期过期删除规则，符合规则的对象会被删除，具体可参考：生命周期 。
- 请检查桶是否授权了其他用户的写权限，被授权的用户都可以删除您的桶中的对象。若您开启了日志记录功能，可以通过日志记录查询到删除对象的用户，具体可参考：日志存储。

1.2 计费常见问题

1.2.1 媒体存储支持哪种计费方式

天翼云门户目前仅支持媒体存储-对象存储能力的按需计费开通，其它能力（如文件存储、块存储等）暂不面向线上预付费客户开放。

对象存储目前支持按需计费，即按实际使用的用量收费，以“小时”为周期统计资源使用量。

更多计费介绍，可参考：计费概述。

1.2.2 如何订购媒体存储

天翼云门户目前支持媒体存储中标准型对象存储能力的按需计费开通，具体可参考 订购指引 。

1.2.3 对象存储如何计费

对象存储标准型总计三个计费项：存储费用、公网流出流量费用、CDN 回源流出流量、请求费用，说明如下：

- 存储费用：即存储容量费用，按用户数据占用的存储空间量收取费用。
- 公网流出流量费用：按存储数据被调用或下载产生的公网流量收取费用。
- CDN 回源流出流量：数据从媒体存储传输到天翼云 CDN 所产生的回源流量。
- 请求费用：请求费用按照发送到对象存储的请求指令次数进行计算，实际上每调用一次 API 都计算一次请求次数。

更详细的计费项说明可参考 计费项 。

1.2.4 存储容量、流量的计算单位如何换算

存储容量

在媒体存储存储容量的计算过程中，进制为 1024，具体示例如下：

1EB = 1024PB，1PB=1024TB，1TB=1024GB。

流量

在媒体存储流量计费项包括：包括公网流出流量、内网流出流量、公网流入流量、内网流入流量、CDN 回源流量。

在流量的计算过程中，进制均为 1024，具体示例如下：

1EB = 1024PB，1PB=1024TB，1TB=1024GB。

1.2.5 存储桶内无对象，为什么还会产生存储费用

存储桶中的当前版本对象、历史版本对象和碎片均会占用存储空间，因而产生存储费用。

此问题可参考以下方式排查桶内是否还有占用存储空间的对象：

- 如果存储桶未启用过版本控制，请检查桶内是否还存在碎片。关于碎片管理的介绍，可参考：其他基础操作 章节中的【碎片管理】说明。
- 如果存储桶启用过版本控制，请检查桶内是否存在历史对象版本以及碎片。关于版本控制的介绍，可参考：版本控制。

1.2.6 存储桶内无对象，为什么还会产生公网流出流量费用

存储桶中如无对象，但账单体现公网流出流量费用，常见原因如下（包括但不限于）：

- 流量每小时结算一次，在这一个小时中您或被授权账号向桶中上传对象后又删除了该对象，结算时您看到桶内无对象，但实际已产生流量。
- 对桶的操作，如设置桶 ACL、设置生命周期，请求中均会携带消息体，会产生流量。
- 对媒体存储发送请求失败，返回 4xx 错误（403 除外），会同时返回消息体，该消息体会产生流量。

1.2.7 CDN 回源的流量为什么按照公网下行流量计费了

媒体存储 CDN 回源流出流量是指数据从媒体存储传输到天翼云 CDN 所产生的回源流量，这部分流量将按照 CDN 回源流出流量计费。具体可参考：计费项。

如您发现 CDN 回源流量按照公网流量计费，可参考一下原因并根据指引进行操作：

- 如您的 CDN 为其他云厂商的产品，则产生的回源流量会按照公网下行流量计算。

- 如您使用天翼云 CDN，需要在源站配置中，选择【媒体存储源站】，此配置下产生的回源流量会按照 CDN 回源流量进行计费。如您选择【IP 或域名】，产生的回源流量将按照公网下行流量计费。
- CDN 回源流量计费模式说明：
  - CDN 回源流量支持按需+资源抵扣包计费，如您为按需计费模式，且源站配置正确，则回源流量会按照 CDN 回源流量资费进行计费，如您需订购资源抵扣包，可联系您的客户经理或致电服务热线：400-810-9889。
  - 目前包周期（用量封顶模式）不支持 CDN 回源量流量计费，如您为包周期计费模式，则所有下行流量（包括 CDN 回源）均会按照公网下行流量计费，如您需转换计费方式，可联系您的客户经理或致电服务热线：400-810-9889。
  - 关于您服务具体的计费模式查询，可参考：订购管理。

1.2.8 欠费停服后，是否还能读取媒体存储中的文件

欠费后天翼云媒体存储服务会自动停止，您所占用的存储空间资源仍会继续扣费，因此欠费余额会累计。如果您在 15 天内充值补足欠款，服务会自动启用。

当欠费超过 15 天，将视为您主动放弃该服务，您保存在天翼云媒体存储系统的全部数据将会被销毁，销毁后数据不可恢复。因此请您及时关注账户余额并及时续费以保证您的服务不受到影响。

若您确认不再使用天翼云媒体存储服务，请务必及时删除存储于媒体存储上的数据。

1.2.9 上传对象到媒体存储产生的流量是否收费

不收费。

媒体存储-对象存储服务中，流量项包括公网流出流量、内网流出流量、公网流入流量、内网流入流量、CDN 回源流量。

上传对象到媒体存储，这部分产生的流量为流入流量，如果您通过公网上传则为公网流入流量，如果您通过内网上传则为内网流入流量，以上的流入流量均不收费。

更多计费项说明，您可参考：计费项。

1.2.10 如何停用媒体存储服务或停止计费

媒体存储按需服务开通后无法手动关停，但可以将媒体存储中的所有数据（包括未完成上传的文件碎片、历史版本对象等）完全删除，且删除所有存储桶后，则不会再进行计费，无需注销账号。

关于碎片管理的介绍以及操作指引，可参考：其他基础操作 章节中的【碎片管理】说明。

关于版本控制的介绍以及操作指引，可参考：版本控制。

关于删除存储桶的介绍以及操作指引，可参考：其他基础操作 章节中的【删除存储桶】说明。

1.3 存储桶常见问题

1.3.1 创建存储桶时，存储桶的命名规则

媒体存储存储桶命名规则如下：

- 长度范围为 3~63 个字符。
- 支持小写字母、数字和短划线（-）。
- 必须以小写字母或数字作为开头和结尾。

1.3.2 创建存储桶后，是否可以修改存储区域

不可以。存储桶创建完成后，不可修改存储区域。

1.3.3 用自己的域名作为桶名，为什么通过 https 访问的时候弹出证书有问题

建议在控制台上建桶时规范一下桶名，不用域名来作为桶名。如需使用自定义域名访问存储桶，可在存储桶绑定自定义域名，可参考：自定义域名。

1.3.4 创建桶失败的原因

当用户碰到创建桶失败时常见失败原因如下，可对照逐一进行排查：

1. 所用的 AK/SK 无效，可能已被禁用或删除；

2. 使用的是子账号 AK/SK 密钥对，但尚未对子账号授予 CreateBucket 权限；
3. 用户由于容量包已用尽（包计费用户）或已欠费（按需计费用户）致使用户已被冻结，使得用户暂时不能进行任何操作；
4. 媒体存储用户在每个区域默认创建桶的上限为 1000 个，检查桶的个数是否已达上限；
5. 对应区域已存在同名桶，不可重复创建；
6. 创建的桶名超过 64 个字符或含有非法字符导致创建失败。

1.3.4 存储桶如何重命名

存储桶不支持重命名操作，如您需要使用新的命名，可重新创建另一个存储桶。

1.3.6 删除存储桶失败的原因

若用户删除桶过程中出现失败，常见失败原因如下，可逐一进行排查确认：

1. 用户已被冻结暂不能进行任何操作；
2. 所用的 AK/SK 无效或没有 DeleteBucket 权限；
3. 对应区域桶不存在；
4. 桶名错填成其他用户桶名；
5. 桶中存在对象不为空桶，直接删除桶导致失败。

1.3.7 存储桶标签最多可以设置多少对

一个存储桶最多可以在设置 10 对标签，且同个存储桶下不可以有相同的标签键。

标签键和标签值的配置也有字符限制，具体如下：

参数	参数说明
标签键	UTF-8 编码格式下，不超过 128 个字符；支持大小写字母（区分大小写）、数字、空格和+ - = . _ : /
标签值	UTF-8 编码格式下，不超过 128 个字符；支持大小写字母（区分大小写）、数字、空格和+ - = . _ : /

更多关于桶标签的介绍以及操作指引，您可参考：桶标签。

1.3.8 是否可以将存储桶 A 的数据迁移到存储桶 B

可以，但针对不同场景，我们建议您使用不同的迁移能力。

场景一：

如果您的桶 A 和桶 B 均存储在天翼云媒体存储，您可以通过【存储桶复制】功能进行数据复制，将桶 A 的数据迁移复制到桶 B，具体可参考：存储桶复制。

场景二：

如果您的桶 A 在第三方云厂商，桶 B 在天翼云媒体存储，您可通过媒体存储提供的数据迁移能力进行操作，将桶 A 的数据迁移到桶 B，具体可参考：数据迁移。

✧ **注意**

目前数据迁移服务暂不收取服务费用，但因迁移数据时会产生数据的上传下载以及 API 请求，所以会产生对应的下行流量费用以及 API 请求次数费用。

具体可参考：计费说明 。

1.3.9 是否可以将存储桶 A 的数据复制到存储桶 B

可以，可以通过以下两个操作实现：

- 通过 XstorBrowser 进行复制操作，具体可参考：复制文件或文件夹 。
- 通过存储桶复制进行复制操作，具体可参考：存储桶复制 。

1.3.10 为什么配置了跨域资源共享（CORS）仍然报错

- 首先请您检查跨域访问是否配置正确，请参考跨域资源共享 。
- 其次，浏览器会缓存历史旧数据，如果浏览器中缓存了设置允许跨域访问之前的请求头数据，那么在您配置过跨域访问之后，请求再次访问此 URL 时浏览器会读取缓存中未含有跨域头的 Response Header，从而产生 No Access-Control-Allow-Origin 的问题。

以上问题可以尝试下面方式解决：

- 缓存穿透，在请求的资源 URL 后添加任意参数。比如访问的资源为 abc.html，在其添加参数改为 abc.html?abc=1。

- 清除浏览器缓存，或者更换浏览器来访问资源。

1.3.11 为什么通过自定义域名访问桶，提示 NoSuchBucket，而通过存储桶访问域名可以访问

通过自定义域名访问存储桶，在完成自定义域名配置后，还需在域名提供商进行 CNAME 解析配置，请您检查是否有 CNAME 解析配置。

配置 CNAME 解析

以腾讯云和阿里云的 DNS 配置为例：

- 腾讯云 DNS 设置方法：

1. 腾讯云域名服务控制台。
2. 选择您需添加 CNAME 的域名，单击【解析】。
3. 进入指定域名的域名解析页，单击【添加记录】。
4. 在该新增列填写域名前缀为主机记录，选择记录类型为 CNAME，填写 CNAME 域名为记录值：
   - 记录类型：选择 CNAME。
   - 主机记录：填写子域名的前缀。若域名为 bucket.abc.com，则添加 bucket；若需要直接解析主域名 abc.com，则输入@；若需要解析泛域名，则输入*。
   - 解析路线：建议选择“默认。
   - 记录值：填写对象存储控制台【域名管理】域名对应的 CNAME 值。
   - TTL：建议填写 10 分钟。
   - 单击【保存】即可。

- 阿里云 DNS 设置方法：

1. 登录阿里云控制台，进入【云解析 DNS】->【域名解析】。
2. 选择您需添加 CNAME 的域名，单击【解析设置】。
3. 选择【添加记录】，在添加记录页进行如下设置：
   - 记录类型：选择 CNAME。

- 主机记录：填写子域名的前缀。若域名为 bucket.abc.com，则添加 bucket；

  若需要直接解析主域名 abc.com，则输入@；若需要解析泛域名，则输入*。

- 解析路线：建议选择默认。
- 记录值：填写对象存储控制台【域名管理】域名对应的 CNAME 值。
- TTL：建议填写 10 分钟。
- 单击【确定】即可。

验证 CNAME 配置

通过 nslookup 命令，如果被转向以下任一域名，则表示 CNAME 配置生效。

- bucketname.**oss.ctyunxs.cn
- bucketname.**oss-web.ctyunxs.cn
- bucketname.**oss-web.xstore.ctyun.cn
- bucketname.**oss.xstore.ctyun.cn

1.3.12 生命周期的适用场景

媒体存储支持基于对象的生命周期配置，您可通过控制台配置相关规则，实现定时删除指定的对象、碎片，管理对象的当前版本或历史版本等。

- 对于一些不再访问对象，可通过生命周期进行删除。
- 在开启版本控制后，如历史版本不再使用，可通过生命周期进行删除。

1.3.13 存储桶复制的适用场景

通过存储桶复制功能，可满足用户以下场景需求：

- 合规性要求：因合规性要求，数据需要在不同存储区域或存储桶保存一份副本。通过存储桶复制，可在媒体存储的存储区域之间复制数据，以满足业务要求。
- 数据迁移：因业务发展需要，将业务数据从一个存储桶复制到另一个存储桶，实现数据的迁移。
- 数据备份与容灾：因业务运行需要，希望所有写入媒体存储-对象存储服务的数据能够在另一存储区域进行备份，以预防在数据发生不可逆损毁时，有安全、可用的备份数据，可实时切换业务访问路径，保证业务不中断。

1.3.14 删除对象操作会同步复制到复制的桶中吗

删除对象操作是否同步，依赖于存储桶复制规则的具体配置。

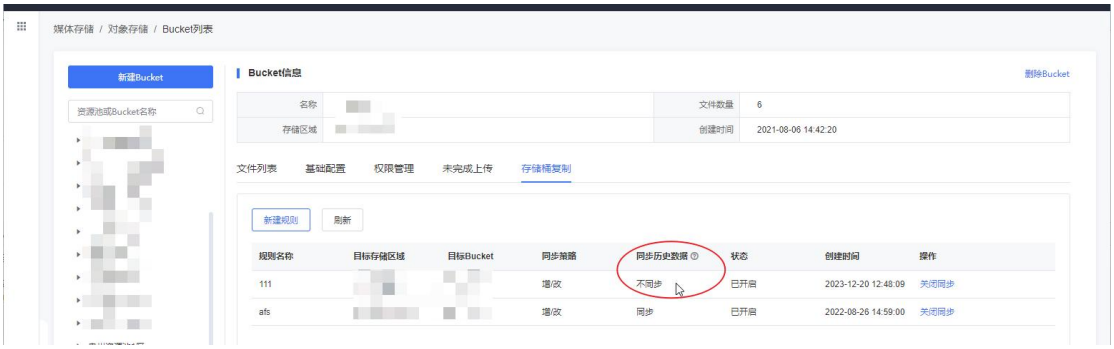
在存储桶复制的配置中，提供【同步策略】选项，您可根据业务需求进行配置：

- 增/改同步：如该 Bucket 内有对象新增和更新操作，将复制到目标 Bucket。
- 增/删/改同步：如该 Bucket 内有对象新增、更新、删除操作，将复制到目标 Bucket。

1.3.15 为什么有些对象没有复制到目标桶

如您发现有些对象没有复制到目的桶，建议参考以下原因：

- 存储桶复制为近实时的操作，可能会存在对象不会立刻复制到目标桶中的情况，请耐心等待。
- 请检查对应的存储桶复制规则，如您存储桶复制规则没有选择【同步历史数据】，则规则启用前已存在的对象不会被复制到目的桶。



1.4 对象常见问题

1.4.1 对象存储中，对于同名文件，是直接覆盖还是新增不同版本的文件

产品中的对象存储现已支持版本控制功能，当存储桶未启用版本控制功能，上传相同名称的文件至存储桶，会直接覆盖已存在的同名文件。

当存储桶开启了版本控制后，上传相同名称的文件至存储桶，会同时存在该对象的多个版本。

1.4.2 如何防止对象被未经授权下载

您可参考以下操作指引进行设置，防止对象未经授权的下载：

1. 可以将存储桶以及对象的 ACL 权限设置为私有，您成为桶以及对象资源的所有者，当其他账号需要对桶或对象进行读写时，您可以把读写权限开放给该用户。 具体操作可参考配置权限管理。
2. 可以通过防盗链设置白名单限制名单外的域名访问存储桶的默认访问地址。如用户根据 Referer 进行防盗链配置，媒体存储会根据浏览器访问请求附带的 Referer 与

用户配置的规则来判断允许或拒绝此请求，如果校验一致，则将允许该请求的访问；如果不一致，则将拒绝该请求的访问。 具体操作可参考配置防盗链。

1.4.3 如何进行批量下载

你可通过媒体存储提供的 XstorBrowser 进行批量下载操作。具体可参考：下载文件或文件夹 。

1.4.4 是否可以找回历史版本的对象

开启存储桶的版本控制功能后，可在文件列表中找到对象的多个版本，并且可以针对指定版本进行恢复、删除操作。

具体可参考 ： 版本控制 。

1.4.4 如何搜索对象存储中的对象

通过媒体存储控制台，可以对当前 Bucket 已上传的文件进行搜索，目前支持通过指定文件名前缀快速搜索。

具体参考： 搜索对象 。

1.4.6 为什么无法搜索到桶中对象

通过媒体存储控制台，可以对当前 Bucket 已上传的文件进行搜索，目前支持通过指定文件名前缀快速搜索。例如，您搜索“abc”，搜索的结果是以“abc”为对象名前缀的对象。

对象的搜索是基于前缀的精确匹配方式，若您输入的对象名称中不包含待搜索对象名称的前缀，则搜索不到相关的对象。例如，您待搜索对象名称为“test123”，您输入“123”搜索，则搜索不到“test123”对象，只能搜索到对象名前缀以“123”开头的对象。

1.4.7 已删除的数据在媒体存储中是否会有残留

用户选择清除数据之后，系统会保证完全删除数据，不会留下残留信息，无需担心信息泄露。

1.4.8 可以在线编辑对象吗

媒体存储一般情况下，不支持在线编辑对象内容。可以把对象下载到本地，修改后再重新上传至媒体存储。

特殊场景下，如对存储在媒体存储的图片进行图片处理，则可参考 图片处理 进行对应操作。

1.4.9 对象标签的使用限制

您可通过媒体存储控制台配置对象标签，一个对象最多支持 10 个标签。

具体可参考：对象标签 。

1.4.10 上传对象失败的原因

当用户碰到上传对象失败时常见失败原因如下，可对照逐一进行排查：

- 通过 API 或 SDK 上传对象时有 5GB 大小的限制，超过 5GB 的文件可以使用 XstorBrowser 或登录媒体存储控制台上传。也可以使用 API 或 SDK 的分片上传接口上传。
- 检查容量是否已超出已购买的容量包额度或账号是否已欠费。
  - 按包计费：
    - 登录媒体存储控制台。
    - 在控制台左侧点击【订购管理】查看资源包状态。
    - 点击对象资源包【详情】查看订购的资源包额度。
  - 按需计费：在控制台顶部导航栏单击【费用】进入费用中心。在【总览页】查看可用额度。
- 结合访问权限来检查该账号是否具有该桶的上传对象权限。
- 若以上并不能解决您的问题，请联系客户经理进一步解决。

1.4.11 下载对象失败的原因

若用户出现下载对象失败常见原因如下，可对照逐一进行检查：

1. 用户由于流量资源包已用尽（包计费用户）或已欠费（按需计费用户）冻结导致不能下载对象；
2. 下载对象链接不带预签名但对象为 private 权限或链接带预签名但已过期；
3. 用户在该对象所在桶上设置了防盗链，下载时未带对应 Referer 头导致下载失败；
4. 用户在该对象所在桶上设置了 Policy 限制了对象下载；
5. 该对象为归档存储对象，需取回后才能下载；
6. 对象不存在，导致 404 下载失败；
7. 用户开启了对象多版本，下载的对象版本为一个 DeleteMarker，实际未对应任何物理对象，导致下载失败；
8. 对象名中含有特殊字符，下载对象时对对象名重复编码导致 404 找不到对象，下载失败；
9. 对象被检测到违规已被屏蔽导致下载失败。
10. 若以上并不能解决您的问题，请联系客户经理进一步解决。

1.4.12 为什么无法在浏览器预览媒体存储中的对象

为进一步落实上级主管部门防范治理通信网络安全工作要求，天翼云媒体存储将于 2024 年 11 月 13 日起陆续在各资源池更新在线预览功能（含匿名访问功能），功能更新后存量桶、新增桶均不支持预览，即用户在访问任何格式的对象（包括但不限于图片、视频、静态网页）时将统一以下载对象方式处理。

具体可见产品公告：天翼云媒体存储关于调整在线预览功能公告。

资源池具体升级时间可见：为什么无法在浏览器预览媒体存储中的对象。

1.4.13 如何查看存储桶内的文件夹大小

可以使用 SDK，指定文件夹路径列举所有文件夹下对象，获取文件大小，再累加。

以 java SDK 举例说明，查询桶 bucketA 下文件夹 fileFolder 的总大小。

```
public static void main(String[] args) {
        String endpoint ="";
```

```
String ak = ""; //访问密钥 ID,可通过控制台-密钥管理获取
String sk = "";//私有访问密钥，可通过控制台-密钥管理获取
String bucketName = "bucketA";
String fileFolder = "fileFolder";
Long fileFolderSize = 0L;


AmazonS3ClientBuilder amazonS3ClientBuilder =AmazonS3ClientBuilder.st
andard();
AWSCredentials credentials = new BasicAWSCredentials(ak, sk);
ClientConfiguration clientConfiguration = new ClientConfiguration();
String signingRegion = "cn-north-1";
AwsClientBuilder.EndpointConfiguration endpointConfiguration = new Aw
sClientBuilder.EndpointConfiguration(endpoint, signingRegion);
AmazonS3 s3Client = amazonS3ClientBuilder.withCredentials(new AWSSta
ticCredentialsProvider(credentials))
.withClientConfiguration(clientConfiguration).withEndpointConfiguration(en
dpointConfiguration).build();
ListObjectsRequest req = new ListObjectsRequest();
req.setBucketName(bucketName);
req.setDelimiter("/");
req.setPrefix(fileFolder);
ObjectListing objectListing=s3Client.listObjects(req);
do{
    for (S3ObjectSummary objectSummary : objectListing.getObjectSum
maries()) {
        fileFolderSize += objectSummary.getSize();
```

```
                }
                    objectListing=s3Client.listNextBatchOfObjects(objectListing);
        }while (objectListing.isTruncated());
        System.out.println("文件夹大小为：" + fileFolderSize);
    }
```

1.4.14 如何获取对象访问路径

媒体存储对象访问路径为：https://桶名.域名/对象名 。例如：https://bucketname.sh4oss.ctyunxs.cn/objectname 。您可以按照以上方式拼接出来完整 URL，或者使用以下方式来获取对象访问路径 URL：

工具	对象 URL
管理控制台	单击对象的【详情】，进入对象详情页，从对象详情中的 URL 选择复制文件 URL，来获取到对象 URL 访问路径。注意：从控制台拷贝的私有对象 URL，其访问有效期限为 24 小时。
XstorBrowser	右键点击对象，从对象属性中 copy 获取到对象 URL 访问路径或者直接选择复制链接。
SDK	媒体存储 SDK 提供预签名接口可以生成预签名 URL。通过预签名 URL，移动端 APP 可以直接上传或者下载文件，具体可参考：使用预签名 URL 直传媒体存储。
API	只支持公共读权限的桶和对象，请参照上面的对象拼接访问方式。

1.5 文件碎片

1.5.1 为什么会产生文件碎片

文件碎片通常是由于数据上传失败而产生的。

在以下情况下通常会导致数据上传失败而产生碎片（以下情况仅列举部分常见场景）：

- 网络条件较差，与媒体存储的服务器之间的连接经常断开。
- 人为终止上传任务。

- 设备故障、断电等特殊情况。

✧ **注意**

由于文件碎片会占用存储空间，因而按照存储容量计费项进行计费，请及时清理不需要的文件碎片。

1.5.2 如何删除文件碎片

在分片上传对象过程中，如人为中断上传任务、本地设备断电等特殊情况导致数据上传失败，则会产生一些不完整的数据，这些不完整的数据即为碎片。

媒体存储支持通过控制台、SDK、XstorBrowser 等工具进行删除碎片的操作，具体指引如下：

操作途径	使用方式
控制台	可参考：碎片管理。
SDK	媒体存储支持多种语言 SDK，请从 SDK 概览页面选择进入对应的开发指南查阅。 1. 调用列举分片上传任务，列出存储桶中已经初始化但未 complete 的分片上传任务。 2. 调用取消分片上传任务，取消分片上传任务并删除碎片。
API（OpenAPI）	1.调用获取分片上传任务列表接口，列出存储桶中已经初始化但未 complete 的分片上传任务。 2.调用终止分片上传接口，取消分片上传任务并删除碎片。 目前 OpenAPI 仅支持西藏资源池调用，如其他区域需通过 API 访问调用，请联系媒体存储技术团队。
API（原生接口）	1.调用获取分片上传任务列表接口，列出存储桶中已经初始化但未 complete 的分片上传任务。 2.调用终止分片上传接口，取消分片上传任务并删除碎片。
XstorBrowser	可参考：碎片管理。

您也可以通过配置生命周期，定期清理文件碎片。具体可参考：生命周期。

1.5.3 文件碎片是否会产生存储费用

文件碎片会产生存储费用。

文件碎片会占用存储空间，因而按照存储容量计费项进行计费，请及时清理不需要的文件碎片。碎片管理具体操作可参考：其他基础操作-碎片管理。

1.6 数据迁移

1.6.1 数据迁移工具适用场景

媒体存储为客户提供在线迁移服务，客户可以通过在线迁移服务实现以下数据迁移场景：

- 将同个账号的媒体存储的某个 Bucket 数据迁移至另一个 Bucket。
- 跨不同的天翼云账号迁移媒体存储间的数据。
- 将第三方数据，如阿里云、AWS 等数据迁移到媒体存储。

目前在线迁移服务为内测能力，如有需要，可联系您的专属客户经理开通试用。

1.6.2 如何迁移其他云厂商对象存储数据到媒体存储

媒体存储提供数据迁移工具，您可轻松迁移数据至媒体存储。具体实践可参考：迁移其他云厂商数据到媒体存储。

1.6.3 使用数据迁移工具是否会产生费用

目前迁移服务暂不收取服务费用，但因迁移数据时会产生数据的上传下载以及 API 请求，所以会产生对应的下行流量费用以及 API 请求次数费用。

具体计费说明可参考：计费说明 。

1.6.4 迁移失败的对象是否可以重新迁移

对于迁移失败的对象，产品提供了重试迁移的功能，用户可以在任务详情页面点击【重试迁移】，产品将自动新建一个迁移任务进行重试。

需要注意的是，已开启迁移报告的迁移任务才可进行重试失败对象迁移。

具体可参考：管理迁移任务 。

1.7 版本控制

1.7.1 版本控制的适用场景

媒体存储支持版本控制，针对 Object 的覆盖和删除操作以历史版本的形式保存下来，可将 Object 恢复至任意的历史版本。

主要适用场景如下：

- 发生数据误删除：开启版本控制后，如果发生误删数据的情况，可通过对象多版本，恢复误删除的数据。（仅限于未指定版本号删除，若指定版本号删除，则无法恢复）。
- 文件被覆盖：对于同名对象文件被覆盖的情况，可以通过对象多版本，找回某个时间节点的对象版本。

1.7.2 是否可以上传同名文件

可以上传同名文件，但是可能会存在同名文件覆盖的情况，具体说明如下：

- 若存储桶未启用版本控制功能，上传相同名称的文件至存储桶，会直接覆盖已存在的同名文件。
- 若您的存储桶开启了版本控制，可上传相同名称的文件至存储桶，会同时存在该对象的多个版本。

1.7.3 版本控制是否会产生费用

版本控制功能本身不收取任何费用。

但对当前版本和所有历史版本的文件都会收取存储费用，因为版本控制开启后，针对对象文件的覆盖和删除操作将会以历史版本的形式保存下来，这部分历史版本文件会占用存储空间，所以会产生存储空间的收费。

具体可参考版本控制。

1.7.4 如何恢复误删除的对象

- 当存储桶未启用版本控制功能，则删除对象后无法找回。

- 当存储桶已启用版本控制，在未指定版本号删除的场景下，您可通过【Bucket 列表-文件列表】找回文件；如删除指定版本号对象，服务将彻底删除该版本对象，无法找回。
- 更多删除对象说明可参考：删除对象。

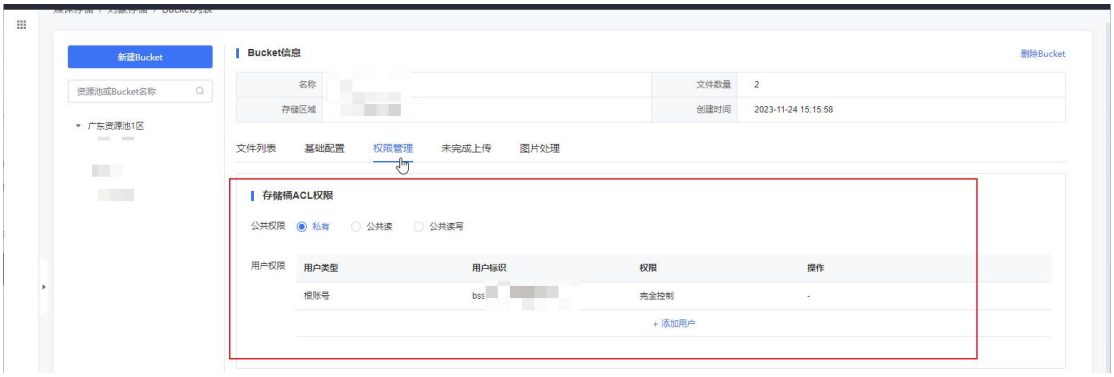
1.8 数据安全

1.8.1 后台工程师能否导出我存在媒体存储中的数据

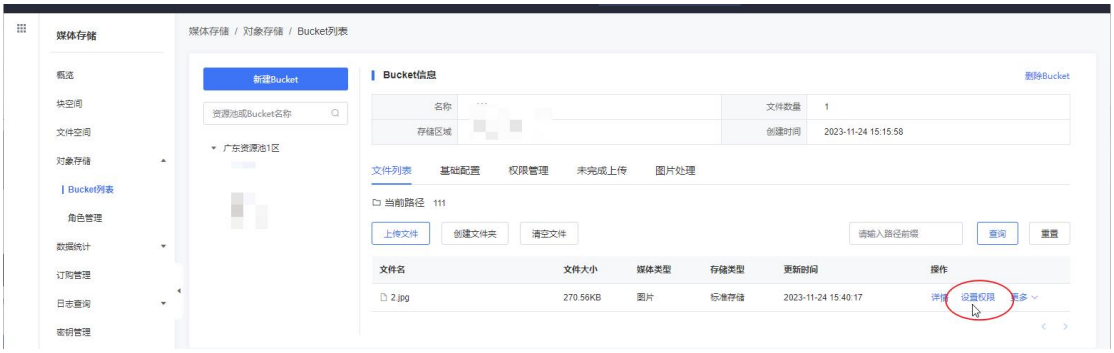
后台工程师无法导出用户数据。

访问桶或对象时，如果桶或对象权限为私有，则只有桶或对象的拥有者才能够访问，且访问时需要提供访问密钥（AK/SK）。

建议您对无需提供公开对外访问的存储桶和对象设置为私有权限。您可以登录媒体存储用户控制台，进入桶列表-权限管理进行相关操作，具体操作可参考权限管理。



对象的拥有者，可以登录对象存储用户控制台，选择目标桶，在对象列表找到需要操作的对象，选择【设置权限】。具体可参考：更改对象访问权限。



1.8.2 媒体存储如何保证我的数据不会被盗用

在存储桶和文件是私有权限的前提下，只有桶或对象的拥有者才能访问，访问时需要提供访问密钥（AK/SK），即使用（AK/SK）加密的方法来验证某个请求发送者身份，并可通过用户控制台进行对密钥的管理。

另外还有 ACL、桶策略、防盗链等多种访问控制机制保证数据的访问安全，所以不用担心您存储在媒体存储的数据会被盗用。

关于管理密钥可参考：密钥管理。

关于访问安全 ACL、桶策略可参考：访问权限。

关于设置防盗链可参考：防盗链。

1.8.3 在使用 AK 和 SK 访问媒体存储过程中，密钥 AK 和 SK 是否可以更换

主账号使用 AKSK 访问时可以替换。在使用过程中，密钥 AK 和 SK 可以随时更换。每个用户最多支持 5 对密钥。

子用户使用子用户 AKSK 可以替换，每个子用户拥有 1 对密钥。

使用临时凭证访问时可以替换，需要注意的是，临时凭证具有有效期，需核实该密钥是否在有效期内。

1.8.4 媒体存储是否支持对象加密上传

媒体存储支持服务端加密，当用户配置服务端加密后，服务将对收到的文件进行加密，再将加密文件持久化保存。当用户通过 GetObject 请求下载文件时，服务自动将加密文件解密后返回给用户，并在响应头中返回 x-amz-server-side-encryption，用于声明该文件进行了服务器端加密。服务端加密目前仅支持部分资源池，具体可参考：服务端加密。

访问方式	是否支持对象加密上传	参考文档
管理控制台	是	可参考：服务端加密。
XstorBroswer	否	暂不支持对象加密上传，但如果桶配置服务端加密功能，那向

访问方式	是否支持对象加密上传	参考文档
		该桶中上传的对象会自动实现加密存储。
API	是	可参考：服务端加密。
SDK	是	天翼云媒体存储兼容 AWS S3 接口，您可以通过 AWS S3 接口的 SetBucketEncryption 方法使用天翼云媒体存储的服务端加密功能。

1.8.5 如何访问或下载已加密的对象

使用媒体存储服务端生成和托管加密密钥的服务端加密（SSE-XOS），用户对整个加解密过程是无感知的，对象的访问和下载不受影响。

您可以使用加密对象分享的临时 URL 进行访问。加密对象分享后，使用分享的 URL 访问时服务端会自动解密。具体可参考：访问方式章节中“通过临时 URL 访问对象存储”的介绍。

1.8.6 我对存储在媒体存储上的数据加密时，可支持哪些加密技术

您在将数据上传到媒体存储前，可以事先对数据进行加密，以保证传输和保存的安全性。媒体存储不限定客户端加密的技术。

用户可根据需要对对象进行服务端加密，使对象更安全地存储，当启用服务端加密功能后，用户上传对象时，数据会在服务端加密成密文后存储。用户下载加密对象时，存储的密文会先在服务端解密为明文，再提供给用户。媒体存储支持完全由服务端生成和托管加密密钥的服务端加密（SSE-XOS）。具体可参考：服务端加密。

1.8.7 追加上传对象是否支持并发场景下的锁机制

媒体存储本身不支持，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。

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。

1.9 权限相关

1.9.1 如何对媒体存储进行访问权限控制

- 块存储：块设备在控制台新建时，需要设置 CHAP iqn 及用户密码，作为客户端连接块空间时的凭证。配置方式可参考：鉴权管理。
- 文件存储：用户在连接文件系统时，需要获取鉴权信息进行相应的鉴权操作。产品控制台提供鉴权管理能力，用户可通过控制台完成相关操作。配置方式可参考：鉴权管理。
- 对象存储：
  - 支持桶策略以及 ACL 进行访问权限管理，具体可参考：访问权限。
  - 支持主子账号配置，配置方式可参考：主子账号。

1.9.2 桶策略和对象策略之间的关系

桶策略是针对桶内不定个或所有对象设置的策略，而对象策略则是只针对桶内某一个对象。具体可参考：访问权限-概述。

1.9.3 桶策略和 ACL 的关系

桶 ACL 可以授权用户对桶进行读写的控制操作，而桶策略可以授权用户操作桶的更多高级设置。

对象 ACL 则是授权用户对桶内对象进行具体的读写操作。

如何选择？

- 以下情况推荐使用桶策略：
  - 不同的用户需要使用不同的权限时。
  - 用户需要使用桶的高级配置功能时。

- 以下情况推荐使用 ACL：
  - 需要对单独对象进行额外的授权时。
  - 需要开放某个对象给所有匿名用户访问时。
  - 仅对桶或对象需要基础的读写权限时。

1.9.4 如何确认存储桶目前的 ACL 权限是什么

您可通过控制台或 XstorBrowser 查看存储桶当前的 ACL 权限。

- 控制台：可参考权限管理。
- XstorBrowser：可参考配置桶 ACL 权限。

1.9.5 如何对存储桶的文件夹进行权限配置

您可通过存储桶 Policy 设置对指定文件夹进行权限配置，在添加策略弹窗输入对应的资源路径即可，如下图所示。

1.9.6 配置访问权限后，为什么还是返回 403AccessDenied

建议您参考以下思路进行排查：

- 账号是否欠费：由于欠费会导致停服，请您保持余额充足或者充值后再试。
- 访问的 AKSK 错误。
- 使用临时凭证或者临时 URL 访问时，凭证或 URL 已不在有效期内。
- 配置的权限错误，建议检查配置的资源、操作以及条件字段。具体可参考：桶策略。
- 由于在对象存储中，优先判断 deny 操作，建议您检查是否存在 deny 配置。

1.10 图片处理

1.10.1 什么是图片处理

针对媒体存储内存储的图片文件（Object），您可以在 GetObject 请求中携带图片处理参数对图片文件进行处理。例如添加图片水印、图片缩放、转换格式等。

支持直接使用一个或多个参数处理图片，存在多个图片处理参数时，将按照参数顺序对图片进行处理。

更多图片处理的介绍与操作指引，您可参考：图片处理-概述。

1.10.2 如何使用图片处理

媒体存储图片处理的使用方式有两种：

- 通过 URL 参数来访问图片处理，用户使用符合图片处理参数的 URL，即可在线获取处理后的图片文件。可参考：使用 URL 处理。
- 通过控制台样式应用，用户对图片文件选择对应的样式后，即可在控制台实时查看处理后的图片文件。可参考：使用控制台。

1.10.3 图片处理有哪些使用限制

图片处理通用的使用限制如下：

- 对原图大小没有限制。
- 总像素不能超过 2 亿 px（动态图片（例如 GIF 图片）的像素计算方式为宽 x 高 x 图片帧数；非动态图片（例如 PNG 图片）的像素计算方式为宽 x 高）。
- 图片格式只支持 JPG、PNG、BMP、GIF、WebP、TIFF、HEIC、AVIF。
- 目标缩放宽高不超过 4,096 px。

1.10.4 支持哪些图片处理操作

媒体存储支持图片处理操作，包括基础图片转换、水印功能、视频截帧、数据处理持久化保存等功能。

具体可参考：图片处理 。

1.10.5 图片处理是否会产生费用

目前媒体存储图片处理能力暂不收费，但通过图片处理后的文件，服务均会默认保存至当前存储桶，会产生对应的存储空间费用。

媒体存储收费的项目可参考计费概述。

更多关于图片处理的介绍以及注意事项，您可参考：图片处理-概述。

1.11 API 与 SDK 相关

1.11.1 OpenAPI 是什么

OpenAPI 是天翼云为客户提供的一种访问方式，用户可以通过 OpenAPI 提供接入节点进行访问。

✧ **注意**

目前媒体存储仅西藏资源池 1 区支持 OpenAPI 访问。具体可参考：终端节点 。

另外，媒体存储-对象存储兼容 AWS S3 协议，推荐您可通过 AWS S3 标准 API 访问媒体存储-对象存储，或使用媒体存储 SDK 进行访问，具体可参考：SDK 概览。

1.11.2 PUT 上传和 POST 上传有什么区别

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
```

两种方式单次上传对象大小范围均为[0, 5GB]，如果需要上传超过 5GB 的大文件，需要通过分片上传实现。

1.11.3 使用媒体存储 SDK 上传超过 5GB 的大文件失败

媒体存储服务端上传对象有最大限制，单次上传最大限制为 5GB。

如需上传超过 5GB 的大对象时，需要通过 SDK 进行分片上传，具体可参照 SDK 概览，根据具体语言选择对应的章节查看。

1.12 工具相关

1.12.1 什么是 XstorBrowser

XstorBrowser 是媒体存储提供的对象存储图形化管理工具，提供类似 Windows 资源管理器的功能，支持完善桶管理和对象管理操作。您可通过 XstorBrowser 操作产品中的对象存储资源。

XstorBrowser 对 PC 机要求如下表所示。

规格项	规格要求	备注
操作系统	Windows 7 Pro SP1 64-bit Windows 10 Pro 64-bit Windows Server 2008 R2 Enterprise 64-bit Windows Server 2016 standard 64-bit Mac OS	暂时不支持 Linux 平台，Windows 用户建议使用 Windows7 及以上版本。

针对不同操作系统，XstorBrowser 下载地址如下表所示。

支持平台	下载地址
Windows x64	XstorBrowser-win 下载
Mac OS	XstorBrowser-Mac 下载

1.12.2 如何获取 AK/SK

您可通过媒体存储控制台获取访问密钥（AK/SK），具体可参考：密钥管理。

1.12.3 如何获取 EndPoint

在 XstorBrowser 登录界面中，EndPoint 信息填写您具体需要访问的存储桶所在区域，您可参考以下方法获取：

方法一：

您可通过媒体存储控制台存储桶的基础配置标签页查询，具体可参考：基础信息查看。

方法二：

根据您开通的存储区域，参考资源池与区域节点 页面，获取对应的 EndPoint 信息。

1.12.4 XstorBrowser 支持什么操作

XstorBrowser 支持的功能如下表所示：

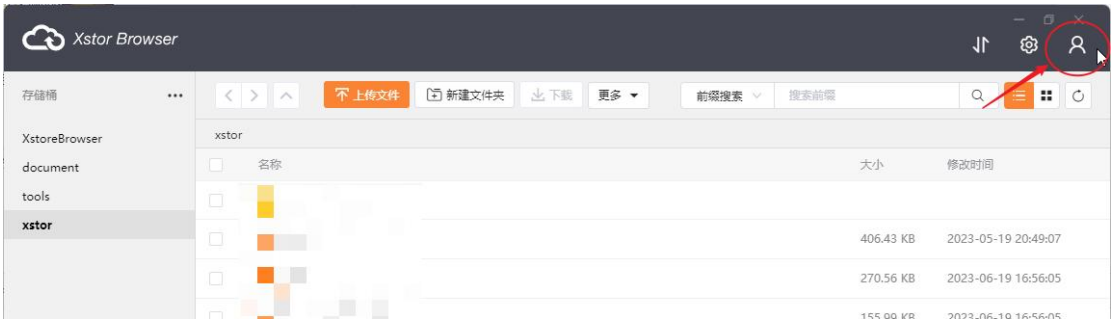
功能	描述	操作指引
桶的基本操作	支持对象存储桶的基本操作，包括创建桶、查看桶的基本信息、碎片管理、删除桶等操作。	存储桶操作
对象的基本操作	支持对象存储中对象的基本操作，包括新建文件夹、上传对象、列举对象、下载对象、删除对象等。	对象操作
桶的 ACL 权限	支持修改桶的访问权限及添加其他用户对该桶的访问权限。	配置桶 ACL 权限
任务管理	主要是对执行的任务按状态进行展示，并提供暂停、删除、继续、重试等基本操作。	任务管理

1.12.5 如何通过 XstorBrowser 同时操作两个存储区域的资源

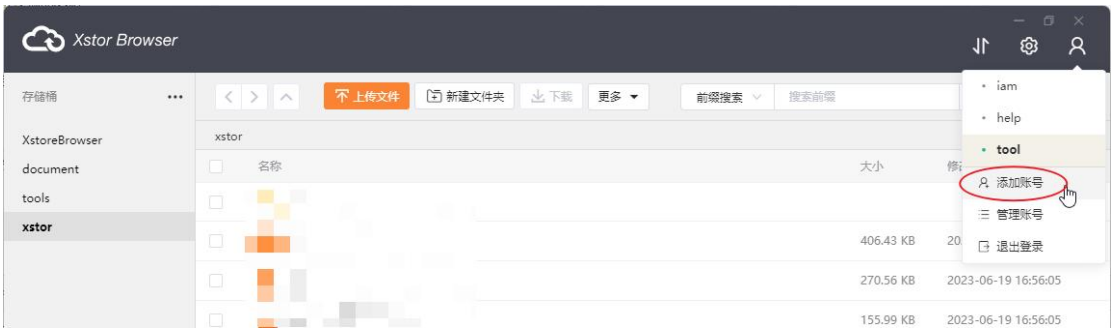
XstorBrowser 中，一个操作账号仅支持操作一个存储区域的资源，如您需有多个存储区域，可添加多个操作账号进行操作。

如您需切换存储区域，可参考以下操作：

1. 点击菜单栏的账号图标。



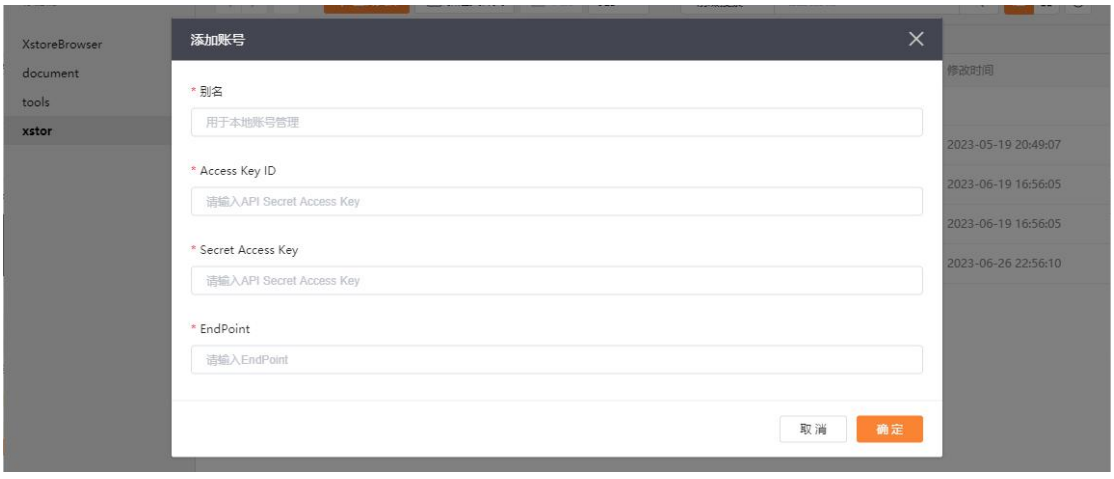
2. 点击【添加账号】。



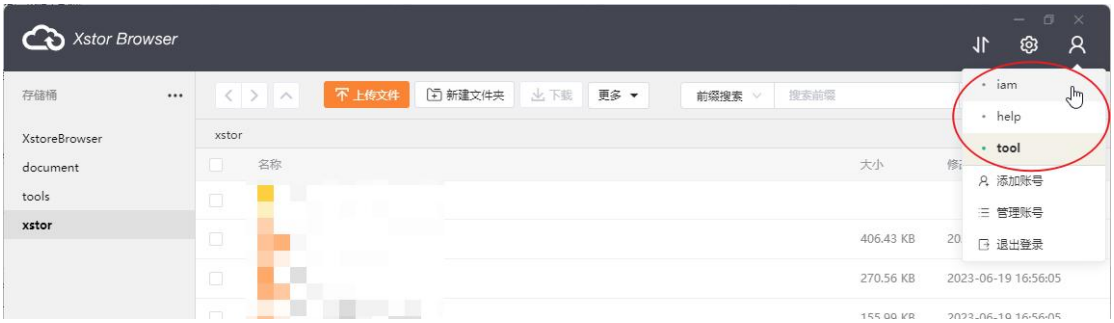
3. 在添加账号弹窗填写别名、AK(Access Key ID)、SK(Secret Access Key)、EndPoint信息，点击【确定】。 别名仅用于操作账号的本地管理，不参与鉴权，您可根据需要自行命名。

   Access Key ID、Secret Access Key 您可通过控制台的密钥管理菜单查询，具体参考：获取访问密钥。

   EndPoint 信息填写您具体需要访问的存储桶所在区域，可通过控制台存储桶的基础配置标签页查询，具体参考：基础信息查看。



4. 再次点击菜单栏的账号图标，即可选择对应的操作账号操作其他存储区域的资源。



1.12.6 如何通过 XstorBrowser 分享文件

您可以通过 XstorBrowser 对象 URL（对象共享）功能，可实现匿名用户通过对象共享链接地址，直接访问对象数据。

登录工具后，选择目标桶，在对象列表选择需要分享的文件或文件夹，点击对应的【...】按钮，下拉选择【分享】，并且在分享链接弹窗自定义分享链接的有效时间。其他用户即可通过临时 URL 访问该对象，超过有效期后则无法继续通过该链接访问。

关于分享文件更多说明可参考：用户指南-对象-其他基础操作。

2. 最佳实践

2.1 概览

最佳实践部分将根据对象存储、块存储、文件存储分别介绍最佳实践。

对象存储

对象存储通过快速使用、数据安全、数据迁移与备份、操作使用分别阐述各自的最佳实践。

快速使用

通过 XstorBrowser 访问对象存储

数据安全

数据安全应用场景

对子用户进行桶级别的权限隔离

校验上传对象的数据一致性

使用服务端加密进行数据保护

数据迁移与备份

迁移其他云厂商数据到媒体存储

通过镜像回源迁移数据到媒体存储

备份存储桶

操作使用

通过生命周期管理对象

性能优化实践

WEB 端直传媒体存储流程优化实践

移动应用使用临时凭证直传

使用预签名 URL 直传媒体存储

块存储

Linux 主机挂载

Windows 主机挂载

文件存储

NFS 协议挂载

CIFS 协议挂载

SMB 协议挂载

Windows 主机自动挂载 CIFS

Linux 主机自动挂载 CIFS/NFS

2.2 对象存储

2.2.1 快速使用

2.2.1.1 通过 XstorBrowser 访问对象存储

XstorBrowser 是媒体存储提供的对象存储图形化管理工具，提供类似 Windows 资源管理器的功能。

用户通过 XstorBrowser 客户端，可完成创建存储桶、上传下载对象等基础操作。本文将为您介绍通过 XstorBrowser 使用媒体存储的实践步骤。

操作流程

步骤一：下载 XstorBrowser。

步骤二：登录 XstorBrowser。

步骤三：创建存储桶。

步骤四：上传/下载对象。

下载 XstorBrowser

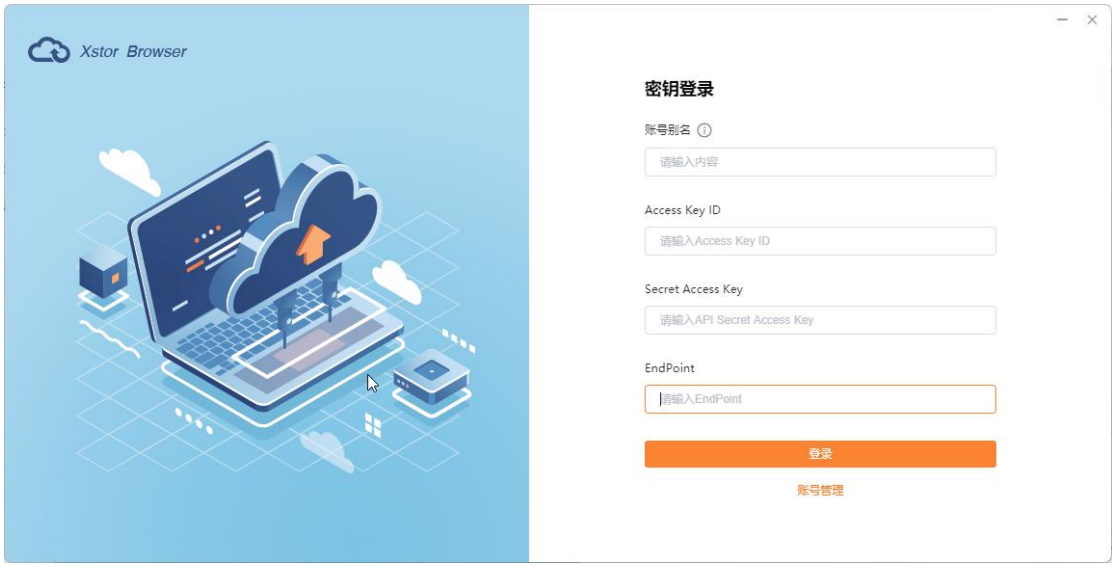
您可通过媒体存储提供下载地址，提前下载 XstorBrowser 安装包。

具体下载地址可见：下载和安装 XstorBrowser 。

登录 XstorBrowser

1. 安装图像化管理工具 XstorBrowser 后，打开登陆界面界面，填写账号别名，Access Key ID、Secret Access Key 以及 EndPoint 信息。
2. Access Key ID、Secret Access Key 您可通过媒体存储控制台的密钥管理菜单查询，具体参考：密钥管理。
3. EndPoint 信息填写您具体需要访问的存储桶所在区域，可通过媒体存储控制台存储桶的基础配置标签页查询，具体参考：基础信息查看。

操作界面如下：

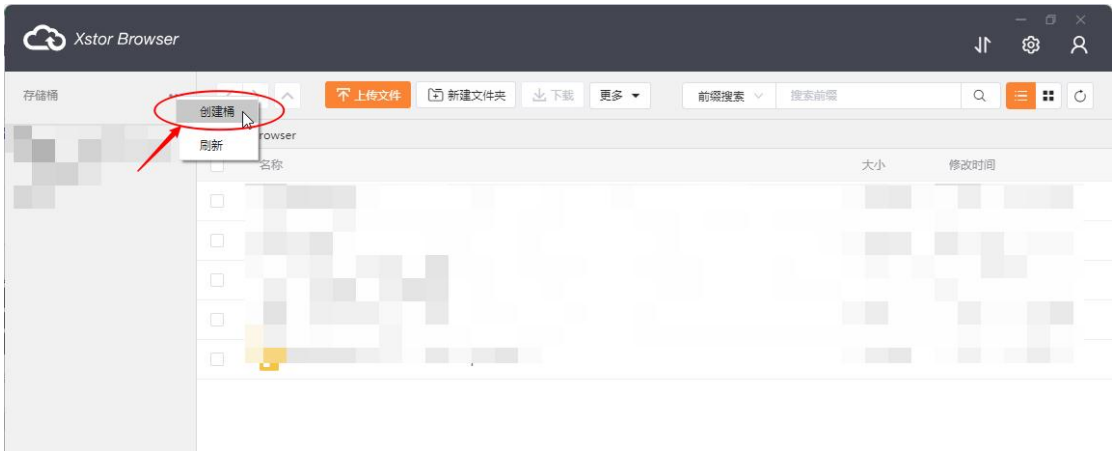


创建存储桶

1. 完成登录后，进入该用户的对象及桶列表页面。按照下方示例，点击左上角的【创建桶】。
2. 在弹窗填写桶名称，点击【确定】完成创建。

更多关于存储桶的概述介绍，可参考：什么是存储桶 。

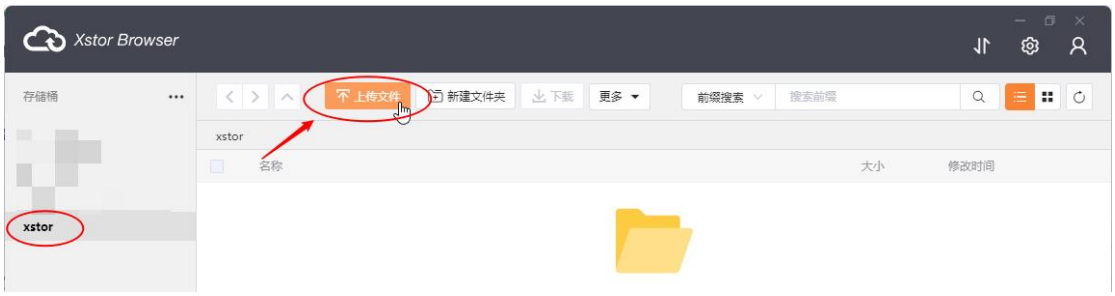
操作界面如下：



上传对象

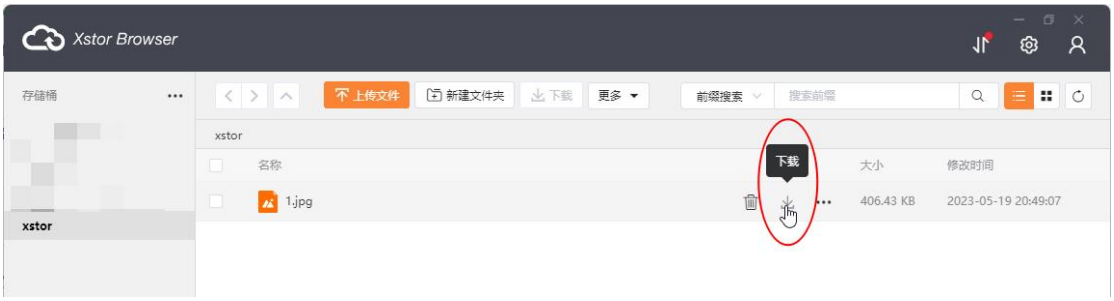
1. 选择具体的存储桶后，点击【上传文件】。
2. 在弹窗中点击【选择文件】，选择需要上传的本地文件后点击【确定】进行上传操作。

操作界面如下：



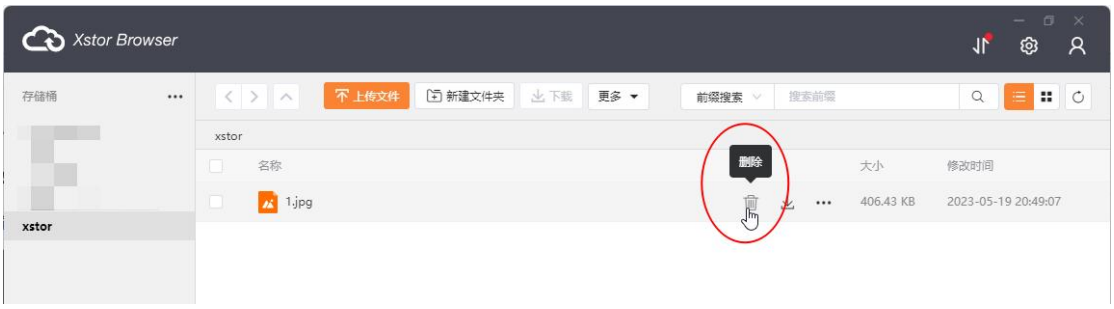
下载对象

选中待下载的对象，点击下载按钮，即可完成下载。



删除对象

选中要删除的对象，点击删除按钮，即可删除对象。



相关文档

更多 XstorBrowser 操作，可参考：工具指南-XstorBrowser。

2.2.2 数据安全

2.2.2.1 数据安全应用场景

媒体存储深知数据安全对用户的重要性。在媒体存储中，我们致力于为您提供可靠、安全的数据存储解决方案，并积极推动数据安全的最佳实践。

我们将通过本文为您介绍媒体存储数据安全的最佳实践，帮助您保护和管理存储在我们平台上的数据。我们将分享一系列有效的措施，旨在确保数据的机密性、完整性和可用性。

我们将从访问控制，数据加密，数据备份和灾难恢复，数据完整性检查，安全审计和监控等几个方面，介绍天翼云媒体存储的数据安全实践。

访问控制

正确使用天翼云媒体存储为您提供的访问控制能力，通过细粒度的访问控制策略，可以有效保护您的数据不被泄露和破坏。

- 建议不同管理员分别使用不同子账号，通过子账号来控制不同管理员的资源访问和管理权限，避免单一账号访问权限过高，导致数据泄露和误操作而产生的风险。

主账号管理员可以结合自己的业务和实际需求，分别创建不同的子用户，然后对子用户授权不同的权限。这样做可以更好的对不同管理员进行权限隔离和管理。详情参考：主子账号。

- 对临时用户，建议使用 STS 发放最小权限的临时访问凭证，让您的资源访问更加安全。临时访问凭证无需透露您的长期密钥，具有一定的时间有效期限，所以针对某些临时需要访问您数据的业务场景或者临时用户，推荐您使用 STS 发放最小权限临时访问凭证，降低潜在的攻击面和数据泄露风险。同时定期审查 STS 角色的权限设置，确保角色权限与用户或服务的实际需求保持一致。如果用户或服务不再需要特定的权限，应立即撤销相应的角色访问权限。对于特权角色，进行更频繁的审查和严格的权限管理。详情参考：STS 角色管理。
- 利用好桶的权限配置功能，有效保护您的数据不被异常访问和操作。天翼云媒体存储提供了存储空间访问策略（Bucket Policy）、存储空间访问控制列表（Bucket ACL）

和对象访问控制列表（Object ACL）等方式，通过它，您可以限制其他用户访问您数据的操作权限，避免您的数据被破坏和泄露。详情参考：访问权限-概述。

- 利用好防盗链配置，通过可以设置黑白名单，限制资源的可见性，能有效避免资源被盗用的风险。天翼云媒体存储提供了防盗链机制，可以防止其他网站或未经授权的第三方使用您的存储资源，防止资源盗链和滥用。只有经过授权的来源网站才能访问和显示您的资源，提高资源的安全性和保密性。详情参考：防盗链。
- 建议将需要公共读写的对象和私有资源对象进行分桶存储，以便简化您的访问控制策略。推荐将公开访问的对象数据使用公有桶存储，私有资源对象采用私有桶存储。同时请切勿将敏感数据放入公共桶存储，将私有桶错授权为公共读写权限，这样容易造成数据泄露风险。详情参考：访问权限-ACL。
- 数据对象临时的访问，建议使用临时 URL 访问数据对象。为了有效避免数据泄露的风险性，针对数据对象的临时访问场景，建议您生成一个临时的 URL 给访问者，同时在生成临时 URL 的时候，设置 URL 的有效期，过期自动失效。详情参考：访问方式。
- 利用好合规保留功能，可以降低您数据被误删和篡改的可能。对于一些私密敏感数据，一经上传，就需要对其进行锁定，防止被他人误删除或者篡改。天翼云媒体存储提供了合规保留功能，开启合规保留后，处于合规保留期的对象均“不可删除、不可篡改”。详情参考：合规保留。

数据加密

- 推荐使用 HTTPS 协议访问天翼云媒体存储，可以有效提高数据传输过程中的安全性。HTTPS 是一种互联网通讯协议，它可以有效防止潜在攻击者使用中间人攻击或类似攻击来窃听或操纵网络流量。我们推荐访问天翼云媒体存储的时候，使用 HTTPS 进行数据传输。
- 敏感或者静态数据，推荐您对数据进行加密存储，能避免数据泄露风险。天翼云媒体存储支持服务端加密，当用户配置服务端加密后，将对收到的文件进行加密，再

将加密文件持久化保存。当用户通过请求下载文件时，天翼云媒体存储自动将加密文件先解密再返回给用户。详情参考：服务端加密。

数据备份和灾难恢复

建立完善的数据备份机制和灾难恢复计划，以应对意外事件，确保数据的可用性，完整性和安全性。

- 建议启用版本控制，可以有效应对对象数据误删除，误覆盖的场景。版本控制功能是一种在相同的桶中保留对象的多个版本的策略。当发生数据被误删除或者被误覆盖的时候，可以通过对象多版本，快速恢复误删除/误覆盖的对象数据。详情参考：版本控制。
- 使用桶复制实现跨区域拷贝，构建数据异地备份，实现异地数据容灾。天翼云媒体存储提供了桶复制功能，他可以帮助您自动进行异地数据备份，实现异地数据容灾的作用。详情参考：存储桶复制。

数据完整性检查

- 采用完整性检查机制，确保存储在媒体存储中的数据在传输和存储过程中没有被篡改或损坏。天翼云媒体存储提供了数据一致性校验功能，可以避免因为网络劫持、数据缓存等原因导致的数据不一致问题。详情参考：校验上传对象的数据一致性。
- 建立安全审计和监控机制，通过及时检测和响应潜在的安全威胁来保护数据的安全性。天翼云媒体存储提供事件通知能力，当对象存储资源发生变动（如新对象上传、删除对象）时，可通过事件通知配置及时收到通知消息。另外，我们还提供用量统计，您可以实时查询和掌握桶中所产生的存储空间、流量与请求次数用量信息。详情参考：事件通知和用量统计。

安全审计和监控

- 建立安全审计和监控机制，通过及时检测和响应潜在的安全威胁来保护数据的安全性。

- 天翼云媒体存储提供事件通知能力，当对象存储资源发生变动（如新对象上传、删除对象）时，可通过事件通知配置及时收到通知消息。另外，我们还提供用量统计，您可以实时查询和掌握桶中所产生的存储空间、流量与请求次数用量信息。详情参考：事件通知和用量统计。
- 天翼云媒体存储还提供了日志存储和告警管理功能来为您的数据安全性保驾护航。其中日志存储功能可以帮您把对桶的各类访问请求记录日志进行存储，方便对桶请求的分析和审计。告警管理功能可以对使用过程中产生的响应状态码，读写请求，流量等进行监控和告警，当这些监控数据达到你自定义的告警阀值，就会发送告警信息。详情参考：日志存储和告警管理。

2.2.2.2 对子用户进行桶级别的权限隔离

适用时间

媒体存储自 2024 年 11 月 13 日对天翼云统一身份认证 IAM 的主子账号权限体系对接互通，该实践方案生效于 2024 年 11 月 13 日后。

适用场景

通过对子用户进行桶级别的权限隔离的实践方案，可实现以下场景需求：

- 子用户级别的权限隔离，每个存储桶仅能由指定的子用户操作。
- 对于 ACL 为“私有”的存储桶或对象，则只有该 Bucket 的主账号或被授权者可以对相应资源进行读写操作。通过子用户授权隔离，可向指定的子用户授予相应资源的读或写权限。

操作步骤

对子用户进行桶级别的权限授权操作步骤如下：

登录媒体存储控制台

登录媒体存储控制台，具体可参考：登录控制台 。

创建存储桶

主账号通过媒体存储控制台，完成存储桶的创建，具体可参考：新建 Bucket 。

授权子用户

主账号通过媒体存储控制台，完成子用户授权，具体可参考：子用户授权 。

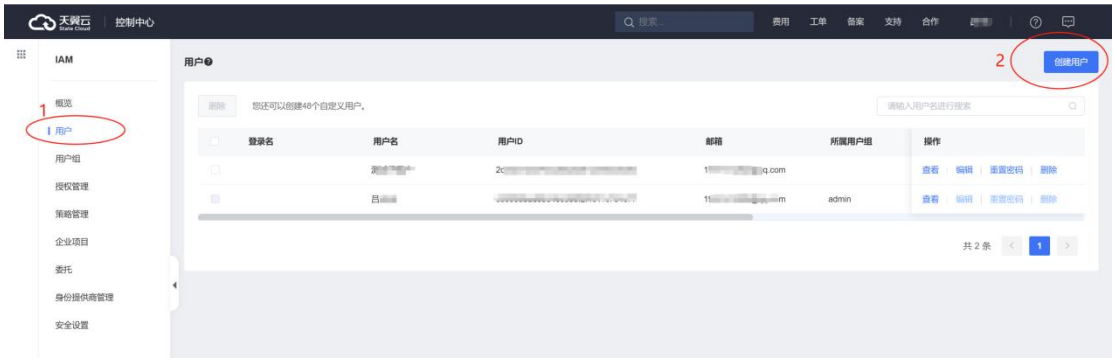
登录 IAM 控制台

主账号使用天翼云账号登录 IAM 控制台。

新建子用户

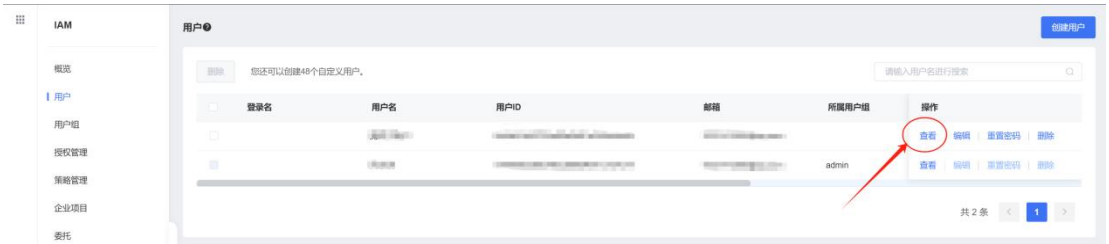
主账号登录 IAM 控制台，选择【用户】页签，再右上角点击【创建用户】，根据页面表单信息填写，点击【确认】即可完成子用户创建。具体可参考：新建子用户 。

操作界面如下：

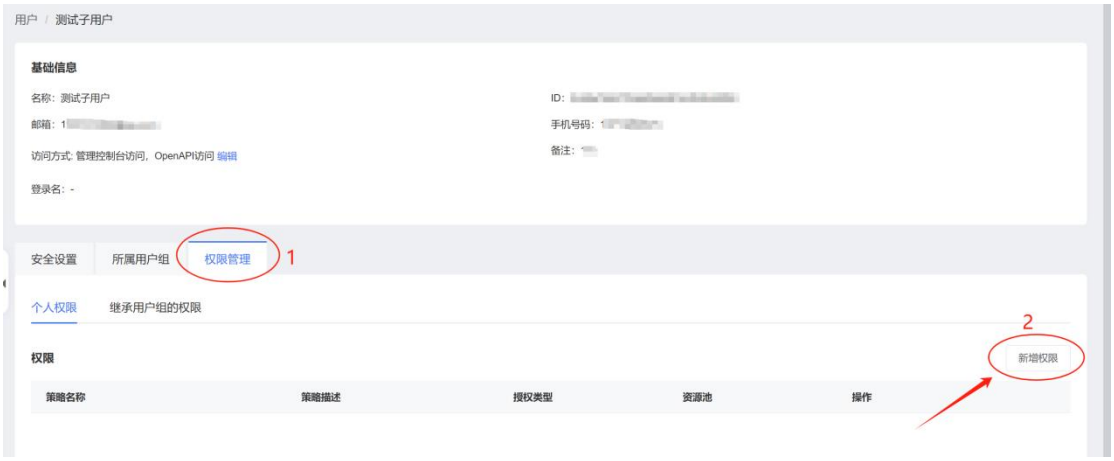


授权子用户

1. 主账号使用天翼云账号登录天翼云统一身份认证控制台，点击左侧导航窗格的【用户】页签，在用户列表选择目标子用户对应的操作栏中点击【查看】按钮，进入用户详情页面。



2. 进入用户详情后，点击【权限管理】，在【个人权限】页签，点击【新增权限】。



3. 在【新增授权】的页面，利用搜索框输入【媒体存储-产品侧授权】，点击搜索图标，可以检索出一条名为【媒体-产品侧授权】的系统全局策略，如下图所示。

4. 勾选【媒体存储-产品侧授权】策略，并点击【下一步】，在【设置最小授权范围】这一页无需做任何配置，直接点击【确定】，完成本次权限配置。至此，当前子用户具备了使用媒体存储产品控制台的权限。

   注意：媒体存储尚未实现与 CTIAM 自定义策略对接。因此，请勿使用 CTIAM 权限管理的【新增权限】选择您【自定义策略】去授权，或将作用范围配置为特定资源池。

5. 上述操作只是给该子用户配置了拥有登录媒体存储产品控制台的权限，如需给子用户配置具体的自定义细化策略，需主账号登录媒体存储控制台，进入【子用户授权】菜单，列表展示已授权对象存储相关权限的子用户信息。

6. 进入【子用户授权】列表，点击对应子用户的【管理】按钮，进入子用户详情页面。

7. 在【策略管理】标签页，点击【新增策略授权】按钮，选择需要新增的权限，点击保存，即可完成授权操作。

获取子用户 AK/SK

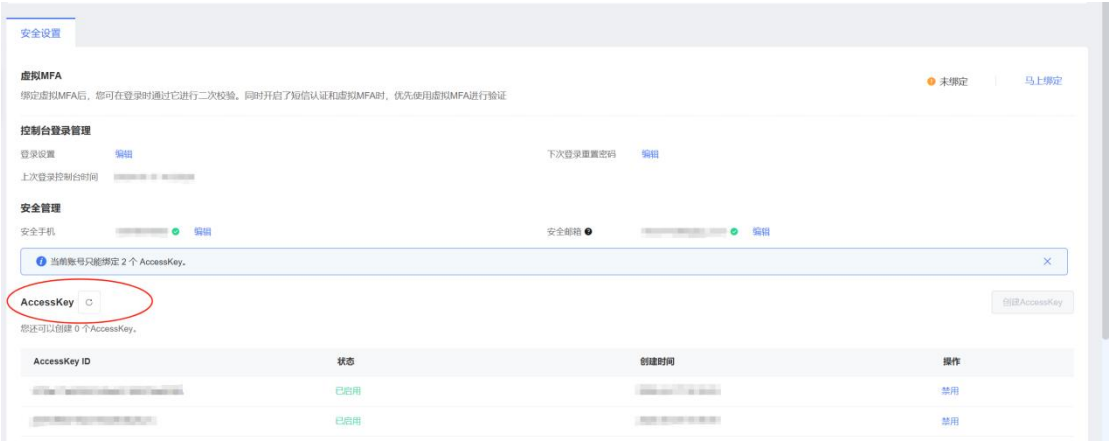
方法一：媒体存储控制台，可查看您原在媒体存储控制台创建的 AK/SK 和从 CTIAM 新建的 AK/SK。

1. 登录媒体存储控制台，进入子用户授权菜单，点击对应子用户的【管理】按钮，进入子用户详情页面。
2. 在【密钥管理】标签页可查看其 AK/SK 信息。

方法二：CTIAM 控制台，只能查看在 CTIAM 新建的 AK/SK。

1. 登录天翼云统一身份认证控制台。

2. 点击左侧导航窗格的【用户】，在用户名对应的操作选项中点击【查看】按钮，进入用户详情页面。

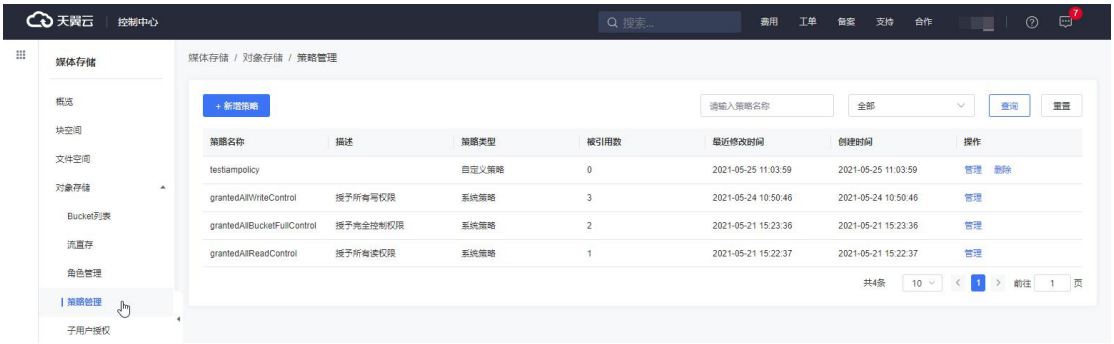


相关问题

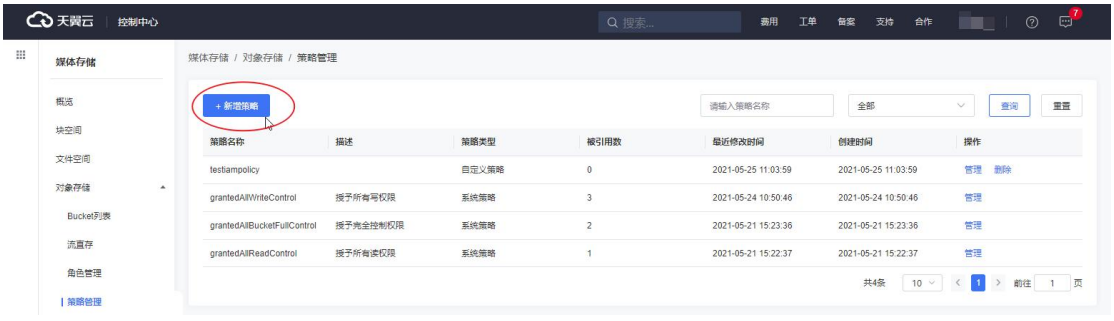
在子用户权限隔离场景中，您可能会遇到需要新增自定义策略，或取消授权等问题，可参考以下步骤操作。

新增策略

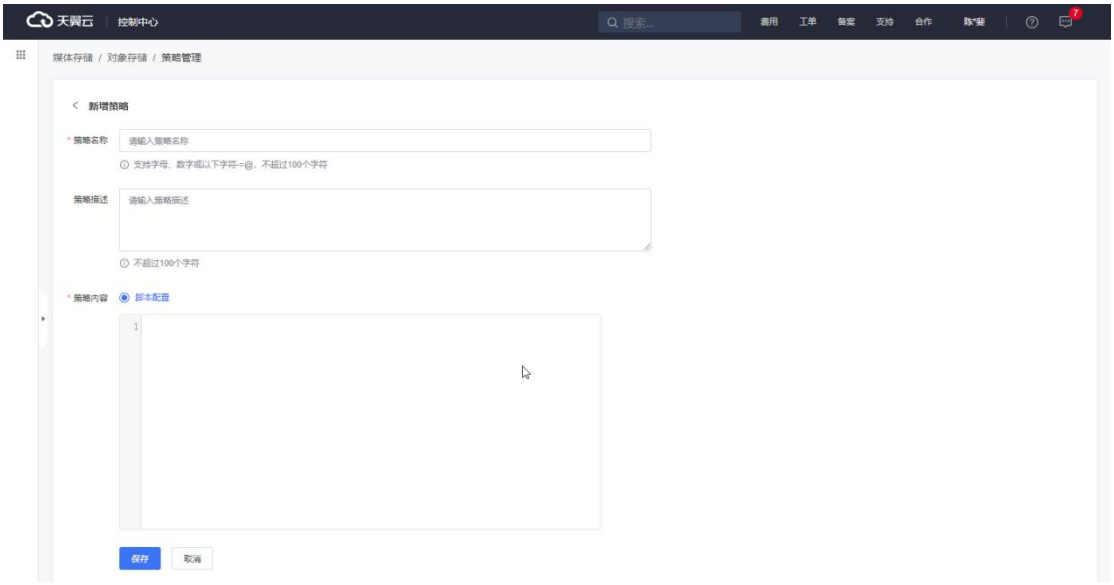
1. 进入媒体存储控制台，进入策略管理。



2. 系统已预置部分策略，如需自定义策略，请点击【新增策略】进行操作。



3. 在新增策略页面中，填写策略名称与策略语句。



策略示例

- 示例一：

以下策略语句为允许子账号对指定 bucket 执行所有操作，子用户可通过子用户控制台、API、SDK 进行相关操作。

其中 BUCKET-NAME 需替换为实际的桶名称。

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Action": "*",
  "Resource": ["arn:aws:s3:::BUCKET-NAME","arn:aws:s3:::BUCKET-NAME/*"],
  "Effect": "Allow",
  "Sid": "BUCKET-NAME"
 }, {
  "Action": "s3:ListAllMyBuckets",
  "Resource": "*",
  "Effect": "Allow",
  "Sid": "BUCKET-NAME"
```

```
  }, {
    "Action": "s3:ListBucket",
    "NotResource": "arn:aws:s3:::BUCKET-NAME",
    "Effect": "Deny",
    "Sid": "BUCKET-NAME"
  }]
}
```

- 示例二：

以下策略语句为允许子账号读取指定 Bucket 中的所有 Object，子用户仅可通过 API、SDK 进行相关操作。如您需创建一个子用户用以 CDN 回源私有存储桶的场景，可参考此示例。

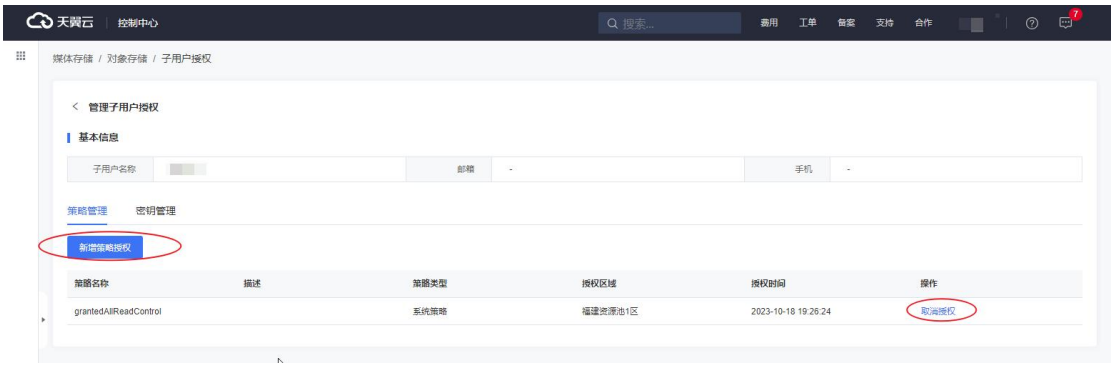
其中 BUCKET-NAME 需替换为实际的桶名称。

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Action": "s3:GetObject",
    "Resource": ["arn:aws:s3:::BUCKET1-NAME/*","arn:aws:s3:::BUCKET2-NAME/*"],
    "Effect": "Allow",
    "Sid": "BUCKET-NAME"
  }]
}
```

取消子用户的某个权限策略

1. 点击对应子用户的【管理】按钮，进入子用户详情页面。

2. 在【策略管理】标签页，点击【新增策略授权】按钮，选择需要新增的权限。如需取消授权策略，则点击对应策略的【取消授权】。



2.2.2.3 校验上传对象的数据一致性

应用场景

媒体存储提供了数据一致性校验功能，可以避免因为网络劫持、数据缓存等原因导致的数据不一致问题。

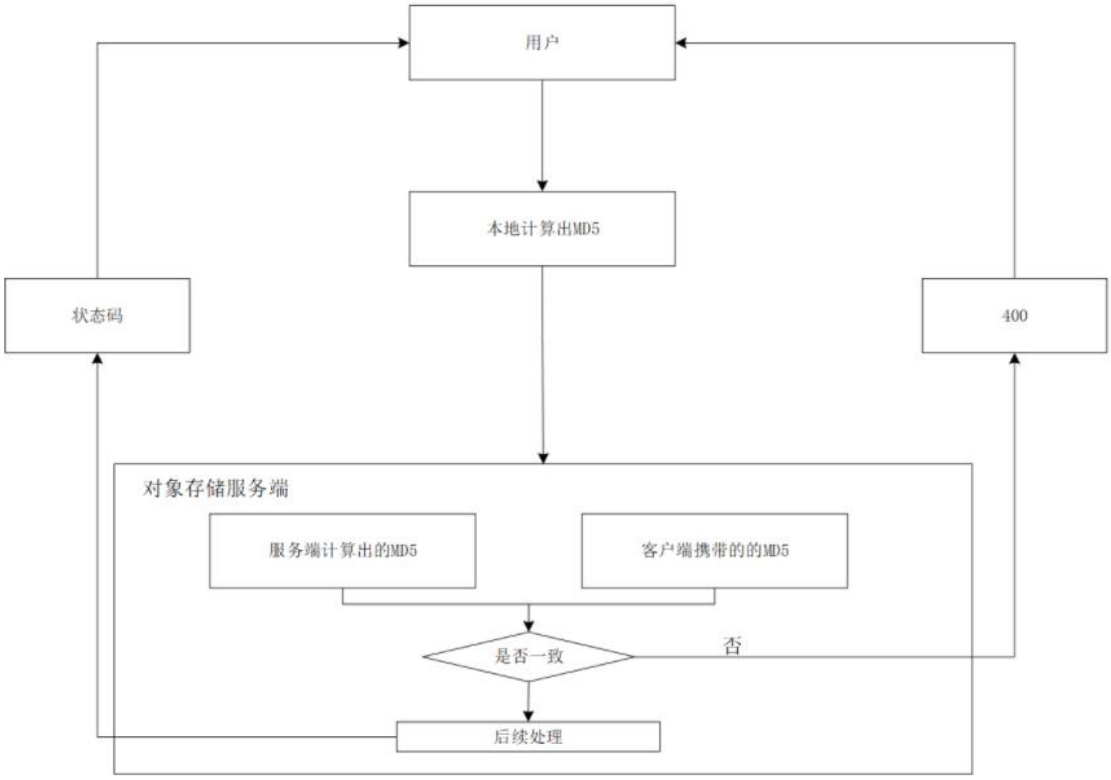
能力概述

媒体存储提供通过计算 MD5 值的方式对上传的数据进行一致性校验。默认情况下，服务不会进行一致性校验。

上传对象时，客户端需要先计算出对象的 MD5 值然后携带上传至媒体存储，媒体存储服务端再根据上传的对象内容计算出 MD5 值，最终与携带上传的 MD5 值进行对比。

如果对比结果一致，对象上传成功，否则上传失败。

具体校验逻辑如下图所示：



校验方法

- 本地对象计算出的 MD5 值作为请求头。
- 上传时设置请求消息参数 Content-MD5 为本地计算得出的校验值，具体示例如下：
  - 请求示例：

```
PUT  /v1/testbucket/test  HTTP/1.1
```

  - 请求头 header：

```
Host:gdoss.xstore.ctyun.cn

Date:Wed,  28  Oct  2023  09:32:00  GMT

Authorization:authorization  string

Content-Type:text/plain

Content-Length:1145

Content-MD5:gnzLDu**********hOew==
```

  - 请求体 body： 实际文件数据。

- MD5 计算方式为：openssl dgst -md5 -binary 上传的对象（example.txt） | base64 。

2.2.2.4 使用服务端加密进行数据保护

背景概述

媒体存储服务的服务端加密功能可以帮助客户实现数据的安全保护，为客户提供更加可靠和安全的数据存储服务。

SSE-XOS（Server Side Encryption XOS）是完全由媒体存储托管加密的服务端加密特性，客户无需管理密钥，服务端会为每个对象使用不同的密钥进行加密，并且会有一个定期轮换的密钥来加密密钥本身，该方式适用于批量数据加解密。

适用区域

本功能目前仅部分资源池支持，具体可参考：资源池与区域节点。

如需使用，可联系客户经理或提交工单申请。

应用效果

服务端加密功能是一种数据安全保障措施，它可以在数据上传至对象存储服务时就对其进行加密，以防止数据被未经授权的访问，降低数据泄露的风险。

通过使用服务端加密，您可以将数据传输至媒体存储服务，服务端会在接收到数据后立即对其进行加密处理，然后再将加密后的数据存储在云端。

未经授权的人访问了存储在云端的数据，他们也无法读取被加密的数据内容。

批量数据加密

当企业需要对大量的数据统一进行加密，您可以使用媒体存储的 SSE-XOS 设置桶级别的加密配置，使得上传到该桶的对象数据自动被加密从而达到批量数据加密，实现批量数据加密的效果。

操作步骤如下：

调用设置存储桶加密配置接口，为存储桶指定一种加密算法：

```
PUT/{bucket}?encryption  HTTP/1.1
Host:cname.domain.com
Content-MD5:ContentMD5
<ServerSideEncryptionConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Rule>
    <ApplyServerSideEncryptionByDefault>
      <SSEAlgorithm>AES256</SSEAlgorithm>
    </ApplyServerSideEncryptionByDefault>
  </Rule>
</ServerSideEncryptionConfiguration>
```

设置了桶的默认加密属性之后，用户往该桶上传对象成功后会在响应中返回以下 header 表示对象数据经过加密：

header	值
x-amz-server-side-encryption	AES256

单独对象加密

当企业需要灵活指定被加密的对象，用户可以在上传对象请求中通过指定以下 header 来完成一次上传对象数据的加密操作。

header	值
x-amz-server-side-encryption	AES256

上传对象成功后在响应中返回以下 header 表示对象数据经过加密：

header	值
x-amz-server-side-encryption	AES256

2.2.3 数据迁移与备份

2.2.3.1 迁移其他云厂商数据到媒体存储

部分用户有大量数据在第三方云厂商对象存储上，如果他们需要将他们的数据迁移到对媒体存储，需要先将第三方云厂商上的对象数据下载到本地，再通过控制台、客户端等工具上传，整个过程耗时又耗力，并且很容易存在漏传、误传等问题。

针对迁移第三方云厂商的对象数据至媒体存储的场景，媒体存储提供在线迁移服务。通过迁移服务，用户只需在控制台配置简单的连接参数以及迁移任务，即可把数据从第三方云厂商简单、平滑地迁移至媒体存储。

下面以阿里云 OSS 数据迁移到媒体存储为例。

前提条件

- 已在天翼云注册账号，并完成实名认证。
- 已开通媒体存储。
- 已在媒体存储中创建桶用来接收数据。
- 创建源端阿里云和目的端天翼云对象存储的访问密钥（AK/SK）。
- 源端桶对应账户需要的权限：只读访问阿里云对象存储服务（OSS）的权限。

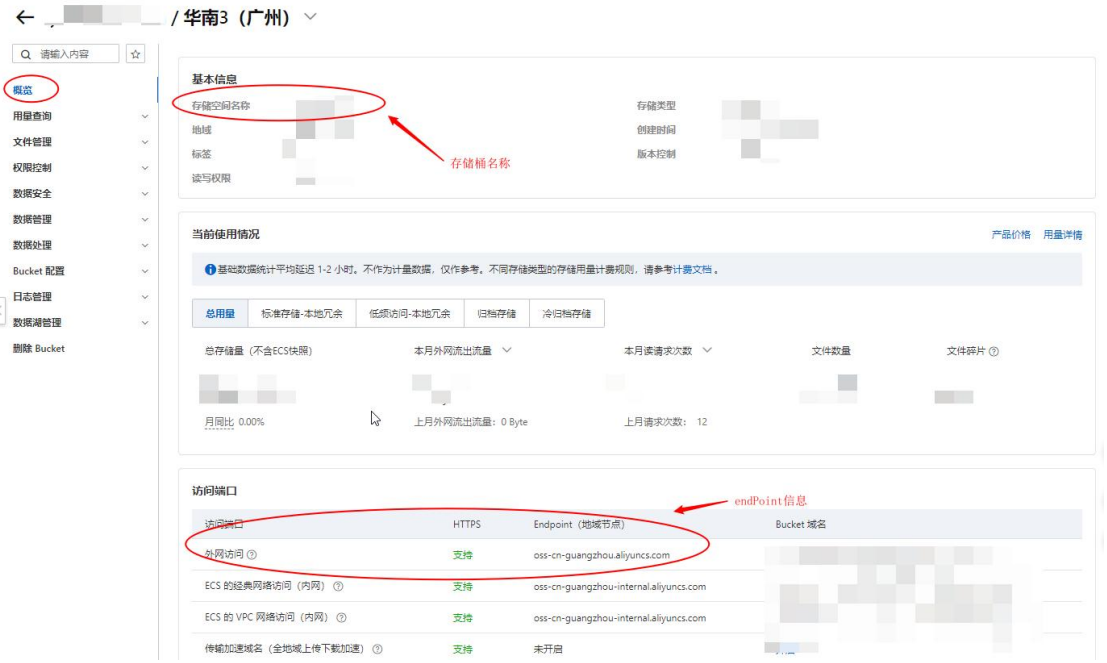
注意： 若您源端有归档状态的数据需要迁移，则应该先解冻，待解冻完成后再进行迁移，否则这些数据会迁移失败。

源端操作

源端阿里云：检查阿里云账号是否拥有 AK/SK 以及只读访问对象存储服务（OSS）的权限，如果没有，参考以下步骤生成 AK/SK 并添加权限。

1. 登录阿里云 RAM 控制台。
2. 在左侧导航栏，选择身份管理 > 用户。
3. 在用户页面，单击用户名称，进入用户详情页面。
4. 在用户 AccessKey 区域，单击“创建 AccessKey”，生成 AccessKey ID 和 AccessKey Secret。

5. 添加权限：在用户页面，单击 RAM 用户操作列的添加权限，授予 RAM 用户只读访问对象存储服务（OSS）的权限。
6. 记录需要迁移的桶名以及 EndPoint 信息，具体获取如下图：



目的端操作

在目的端创建用于存放迁移数据的桶，创建方式可参考 新建 Bucket 。

迁移操作

1. 登录媒体存储控制台。
2. 单击“在线迁移”，进入在线迁移管理页面。
3. 单击“创建迁移任务”，并设置任务相关参数，完成迁移任务的创建。
4. 设置“选择源端”区域的参数：

参数	设置
数据源	选择 oss。
访问密钥	填写源端阿里云的访问密钥（AK）。
私有访问密钥	填写源端阿里云的私有访问密钥（SK）。

参数	设置
Endpoint	填写源端阿里云的 Endpoint。
BucketName	填写源端阿里云需要迁移的数据所在的桶名。

5. 设置“选择目的端”区域的参数。

参数	设置
存储区域	填写目的端天翼云媒体存储的存储区域。
BucketName	选择迁移的目的桶。

6. 填写完成后，请单击“测试链接”，则会校验是否可正常链接到源端，链接正常则可进入下一步。
7. 单击“下一步”，进入“配置任务参数”页面。完成相关配置。
8. 创建完成后，任务进入初始化，初始化完成后，您可在任务列表查看迁移进度。

相关文档

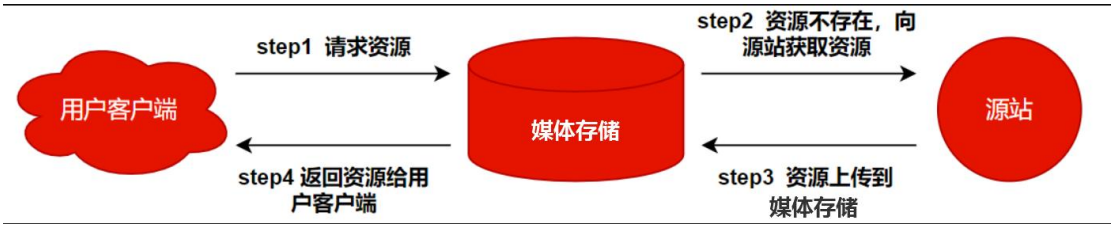
- 更多关于数据迁移的说明可参考：数据迁移-概述。
- 更多关于创建迁移任务参数说明可参考：创建迁移任务。

2.2.3.2 通过镜像回源迁移数据到媒体存储

正常情况下，当客户端访问媒体存储中的资源时，若资源不存在，则服务端会返回 404 错误。媒体存储提供回源功能，配置回源规则后，当请求者访问的对象在存储桶中不存在时，可以根据回源规则从指定的源站获取对象。

在回源配置中，可开启 3xx 跟随，媒体存储会同时将数据保存到存储桶中，整个过程不中断业务，实现客户源站数据热迁移的需求。

镜像回源流程如下图所示：



本功能目前仅部分资源池支持，具体可参考：资源池与区域节点。如需使用，可联系客户经理或提交工单申请。

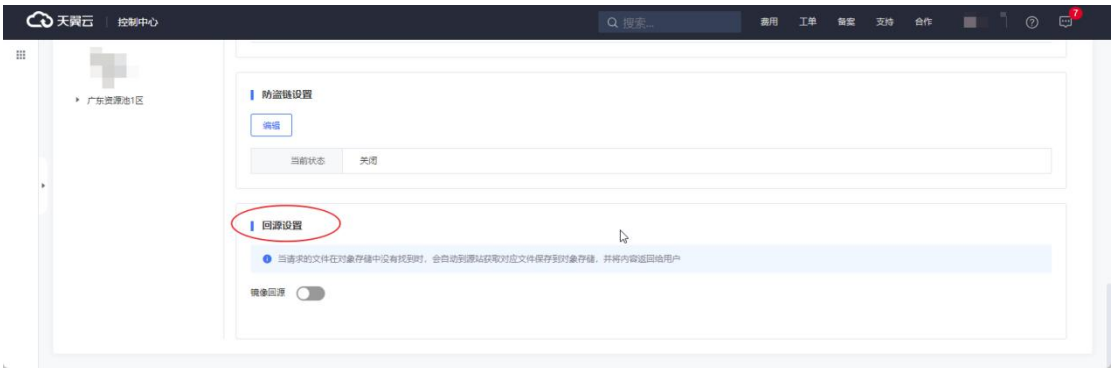
配置方法

本实践将通过控制台操作介绍具体配置方法。

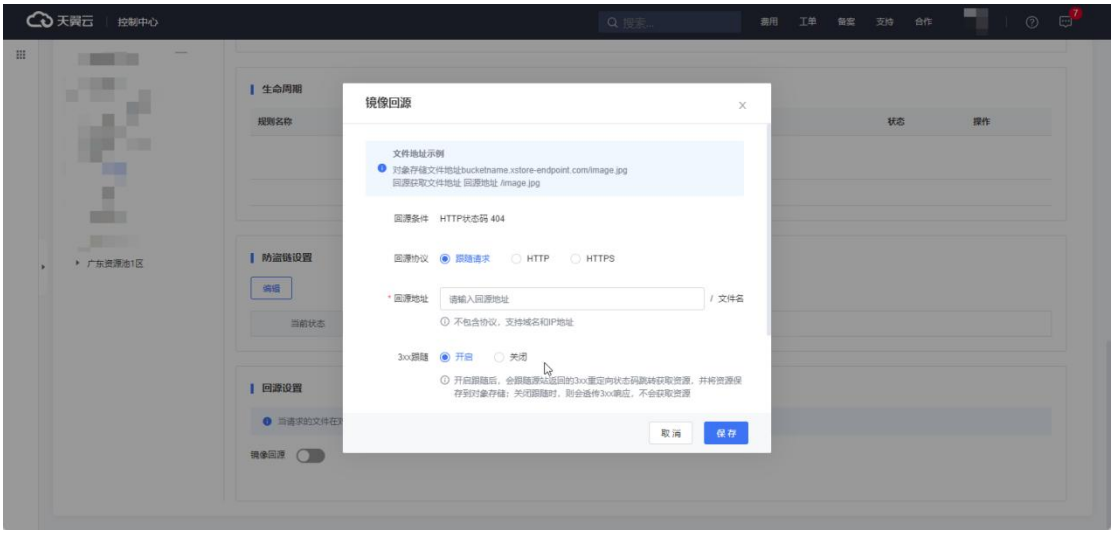
1. 登录媒体存储控制台，进入【对象存储-Bucket 列表】菜单。

2. 选择需要配置镜像回源的存储桶，并点击【基础配置】页签。

3. 在【回源设置】模块，打开回源设置。



4. 在弹窗填写相关信息，点击【保存】完成操作。



配置参数说明

参数	参数说明
回源条件	触发回源规则的条件，默认 HTTP 状态码 404。
回源协议	对象存储访问源站时的 HTTP 协议：选择跟随请求协议，以请求对象存储所使用的协议访问源站；选择 HTTP 或 HTTPS，则以对应选择的协议访问源站。
回源地址	设置回源的源站地址，填写时不需包含协议，支持域名或 IP 地址填写。
3xx 跟随	开启跟随后，会跟随源站返回的 3xx 重定向状态码跳转获取资源，并将资源保存到对象存储；关闭跟随时，则会透传 3xx 响应，不会获取资源。
回源超时	设置回源超时时间，超时后直接返回 404 状态码，最大 300 秒。
新增回源 header	支持设置回源 header，设置完成后，可携带指定的新增头部访问源站，当前最多支持新增 10 个回源 header。

2.2.3.3 备份存储桶

对于存储在媒体存储-对象存储的数据，本产品提供了数据迁移以及存储桶复制的备份能力。用户可根据不同场景的备份需求，选择对应的备份方式。

通过数据迁移备份

媒体存储提供在线迁移服务，用户可通过在线迁移服务实现以下数据迁移场景：

- 将同个账号的媒体存储的某个 bucket 数据迁移至另一个 bucket。
- 跨不同的天翼云账号迁移媒体存储间的数据。
- 将第三方数据，如阿里云、AWS 等数据迁移到媒体存储。

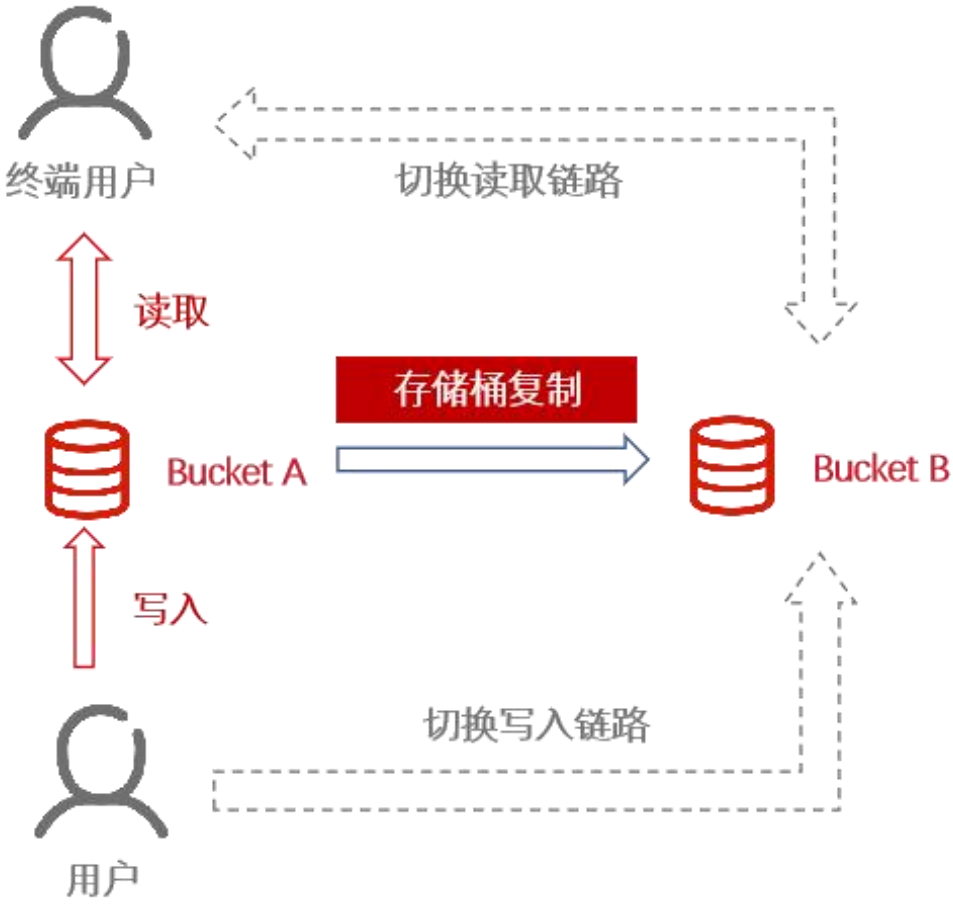
数据迁移会产生备份存储空间、请求次数等费用，具体可参考：计费说明。

支持通过控制台进行数据迁移的配置，具体配置方法可参考：创建迁移任务。

通过存储桶复制备份

存储桶复制是跨不同存储区域或同一存储区域的 Bucket 自动、异步（近实时）复制对象（object），可根据配置，将源桶 Object 的创建、更新和删除等操作复制到目标 Bucket。具体介绍及配置方法可参考：存储桶复制。

存储桶复制流程如下图：用户配置 BucketA 复制到 BucketB 的增改同步规则，当 BucketA 的创建对象时，会自动复制一份到 BucketB，满足备份需求。用户可根据业务规划，切换写入或读取链路，保证业务不受影响。



2.2.4 操作使用

2.2.4.1 通过生命周期管理对象

适用场景

生命周期是指对象从更新到被删除的周期时间，媒体存储支持基于对象的生命周期配置，您可通过控制台配置相关规则，实现定时删除指定的对象、碎片或管理历史版本文件。

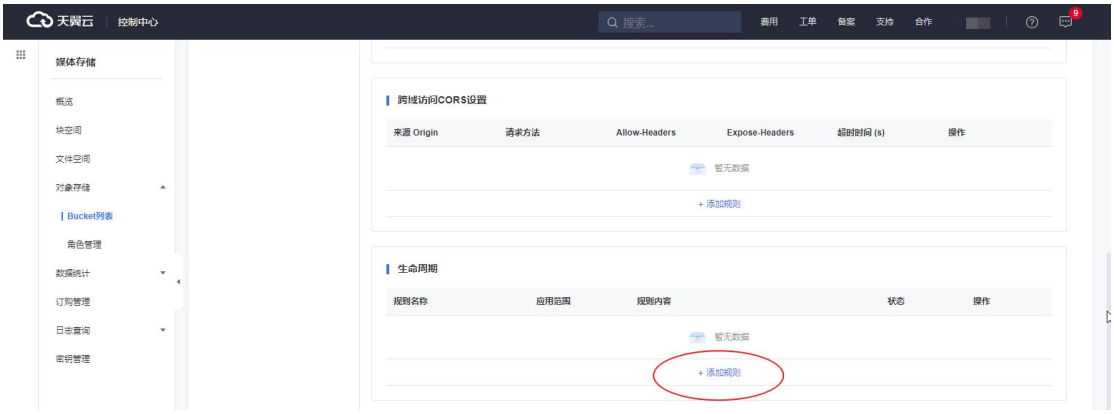
用户通过设置存储桶的生命周期规则，按照设定的生效条件，将与生命周期规则匹配的对象进行删除，从而无需逐一或者批量删除对象，降低用户的操作难度，并可进行成本控制。

操作步骤

1. 登录媒体存储控制台，进入【对象存储-Bucket 列表】菜单。

2. 选择需要配置生命周期的存储桶，并点击【基础配置】页签。

3. 在【生命周期】模块点击【添加规则】。



4. 根据弹窗指引填写相关的规则信息，包括规则状态、规则名称、应用范围、管理文件时间、删除碎片时间。点击【确定】完成操作。

5. 需要注意的是，如果一个对象同时命中多条生命周期规则，对象存储会以最短过期时间为准执行。

2.2.4.2 性能优化实践

媒体存储按照对象名的 UTF-8 编码范围来进行自动分区管理，对系统进行水平扩展与动态负载均衡。如果您在上传大量文件时，在对象命名规则上使用了顺序前缀（如时间戳或字母递增顺序），有可能导致大量对象的请求访问集中于某个特定分区，造成访问热点。从而导致热点分区上的请求速率受限，出现访问时延上升的问题。

针对此类问题，我们建议您在为对象命名时使用随机前缀，让对象均匀分布在多个分区上。

例如：

您可能上传的对象文件名格式如下：

```
bucket_name/test-20230613/1.log
bucket_name/test-20230613/2.log
bucket_name/test-20230613/3.log
bucket_name/test-20230613/4.log
bucket_name/test-20230613/1.log
...
```

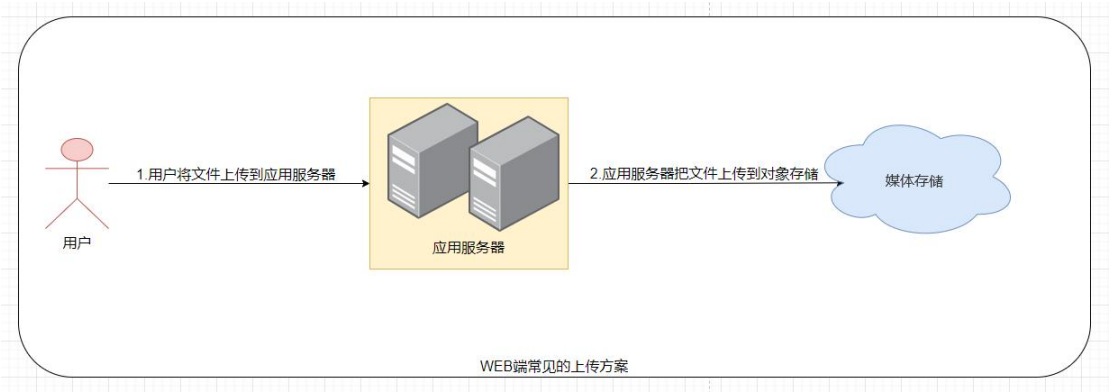
此时，我们建议您将对象计算 hash 值(即对象的 md5)，然后取 md5 中的前 3-4 个字符作为对象名的前缀。我们这里以 3 个字符为例，可以修改成如下形式：

```
bucket_name/d3b/test-20230613/1.log
bucket_name/fe2/test-20230613/2.log
bucket_name/94d/test-20230613/3.log
bucket_name/fa3/test-20230613/4.log
bucket_name/e3a/test-20230613/1.log
...
```

2.2.4.3 WEB 端直传媒体存储流程优化实践

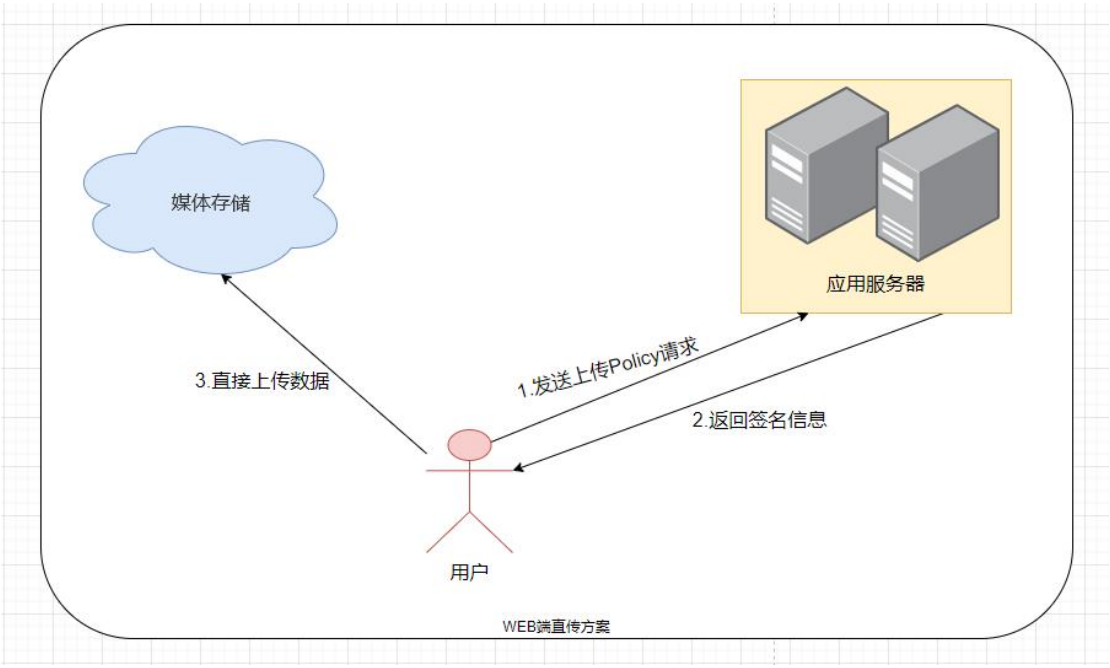
优化实践

在 WEB 端需要将用户的数据上传到媒体存储，常见的方法通常是让用户通过浏览器先上传文件到用户的应用服务器，然后应用服务器再上传到媒体存储。这种方案上传的中转数据需要经过用户应用服务器，传输效率低，同时在多任务上传的过程，无疑会增加应用服务器的压力。具体的上传流程如下图：



本文将介绍另外一种方案，通过在 WEB 端直接调用 PostObject 接口，将用户的文件对象直传给天翼云媒体存储。这种方案，具有传输效率高，降低用户应用服务器压力等优点。

WEB 端直传媒体存储流程如下图：



在 WEB 端直传，我们主要基于媒体存储的 post 接口，用表单上传的方式，将对象上传到指定的桶里面，所以在上传之前，我们需要先创建桶。同时上传的对象文件，最大不能超过 5GB。

利用 post 接口进行表单直传的详细具体步骤：

1.将 post 上传接口中基于表单上传需要发送的 Policy 信息，发送给应用服务器。我们假设发送给应用服务器的 Policy 信息，如下：

```
{"expiration":"2023-12-28T00:00:00Z","conditions":[{"bucket":"openapi-hp-test"},{"
key":"post_dog.png"}]}
```

2.应用服务器收到 Policy 信息后，对其进行签名，然后返回给用户。

应用服务器，我们可以采用 Java,Python,GoLang 等语言进行开发，计算 post 上传的签名信息。

我们假设后端应用服务器是 python 开发的，使用 v2 签名，示例代码如下：

```
import base64
import hmac
import hashlib
import binascii


def sign2(key, msg):
    return hmac.new(bytes(key, encoding='utf-8'), msg.encode("utf-8"), hashlib.sha
1).digest()



def getSignatureKey2(key, encodepolicy):
    signaturebyte = sign2(key, encodepolicy)
    return binascii.b2a_base64(signaturebyte)


if __name__ == "__main__":
    SK = 'xxxxx'
    bucketName = 'xx'
    objectKey = 'xx'
    expirationTime = "2023-12-28T00:00:00Z"
```

```
policy = "{\"expiration\": \"%s\"," \
        "\"conditions\": [" \
        "{\"bucket\": \"%s\" }," \
        "{\"key\":\"%s\"}" \
        "]}" % (expirationTime, bucketName, objectKey)
print(f"policy:{policy}")
encodePolicy = bytes.decode(base64.b64encode(policy.encode('utf-8')))
print(f"encodePolicy:{encodePolicy}")
 #计算签名
signature = getSignatureKey2(SK,encodePolicy)
print(f"signature:{signature}")
```

如果使用 v4 签名，请参考 SDK 和 Demo 相关代码: SDK 概览

3.准备表单 HTML 页面。

表单 HTML 页面代码示例如下:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>

<form action="http://bucket_name.domain.ctyun.cn/" method="post" enctype="
multipart/form-data">
key
<!-- Object name -->
<input type="text" name="key" value="object-key" />
```

```
<!-- Base64 code of the policy -->
<input type="hidden" name="Policy" value="*** your policy ***" />
<!-- AK -->
<input type="hidden" name="AWSAccessKeyId" value="*** your Access Key ***"
/>
<!-- Signature information -->
<input type="hidden" name="signature" value="*** your signature ***"/>

<input name="file" type="file" />
<input name="submit" value="Upload" type="submit" />
</form>
</body>
</html>
```

注意：

（1）html 表单中的 Policy 值需要为 base64 编码后的值。

（2） html 表单中的 signature 值为你应用服务器的返回的签名结果。

4.选择你需要上传的本地文件，然后进行表单上传。

常见问题

- 跨域问题：当出现跨域问题的时候，请参考跨域资源共享对其进行配置。
- 参考 post 接口 API 相关文档描述，如果桶为 public-read-write，那么 Policy 可以为空。桶权限非 public-read-write 时，Policy 不能为空，其值在鉴权时需要用到。如果 Policy 为空，那么 AWSAccessKeyId 和 signature 都可以为空，如果 Policy 不为空，那么就需要同时填入 AWSAccessKeyId 和 signature 字段。
- 为避免造成 AK/SK 泄露，不建议直接在 WEB 端签名，可在后端直接计算预签名 URL，然后前端使用预签名 URL 授权访问媒体存储。

- 这里以应用服务器使用 python 计算预签名 URL，前端使用临时 URL 访问媒体存储为例。
- 利用 python 在应用服务端计算预签名 URL：

```
import botocore.config
import botocore.session
import botocore.signers



def generate_putobject_presigned_url(access_key, secret_key, end_p
oint, bucket,key,region):
    config = botocore.config.Config(signature_version='s3v4')
    session = botocore.session.get_session()

    s3_client = session.create_client(
        's3',
        aws_access_key_id=access_key,
        aws_secret_access_key=secret_key,
        endpoint_url=end_point,
        region_name=region,
        config=config)

    expiration_time = 3600  # URL 过期时间（单位：秒）
    # 构建上传预签名 URL 的请求参数
    params = {
        'Bucket': bucket,
        'Key': key,
```

```
        'ContentType': 'text/plain'  # 替换为你要上传的文件的 MIME 类型
    }
    presigned_url = s3_client.generate_presigned_url(
        ClientMethod='put_object',
        Params=params,
        ExpiresIn=expiration_time)

    print(f"presigned_url:{presigned_url}")

if __name__ == '__main__':
    AK = 'xxx'
    SK = 'xxx'
    bucketName = 'hp-test'
    objectKey = 'post_dog.png'
    endpoint = 'http://domain.ctyun.cn'
    region='ap-east-1'
    generate_putobject_presigned_url(AK,SK,endpoint,bucketName,o
bjectKey,region)
```

– WEB 端上传的时候，URL 使用从应用服务器获取到的预签名 URL:

```
<html>
  <head>
    <title>使用 PUT 请求上传文件内容</title>
  </head>
  <body>
    <h1>使用 PUT 请求上传文件内容</h1>
    <input type="file" id="fileInput" />
```

```
    <button onclick="uploadFile()">上传文件</button>


    <script>
     function uploadFile() {
       var fileInput = document.getElementById('fileInput');


       if (fileInput.files.length === 0) {
         alert('请选择要上传的文件');
         return;
       }


       var file = fileInput.files[0];
       var xhr = new XMLHttpRequest();
       xhr.open('PUT', '/xxxx', true);  // 替换成实际的上传 URL
       //要上传的文件的 MIME 类型，需要与生成预签名的时候一致
       xhr.setRequestHeader('Content-Type', file.type);  // 设置请求头
的 Content-Type


       xhr.onload = function() {
         if (xhr.status === 200) {
           alert('文件上传成功');
         } else {
           alert('文件上传失败');
         }
       };
       xhr.send(file);
```

```
        }
      </script>
    </body>
  </html>
```

2.2.4.4 移动应用使用临时凭证直传

实践背景

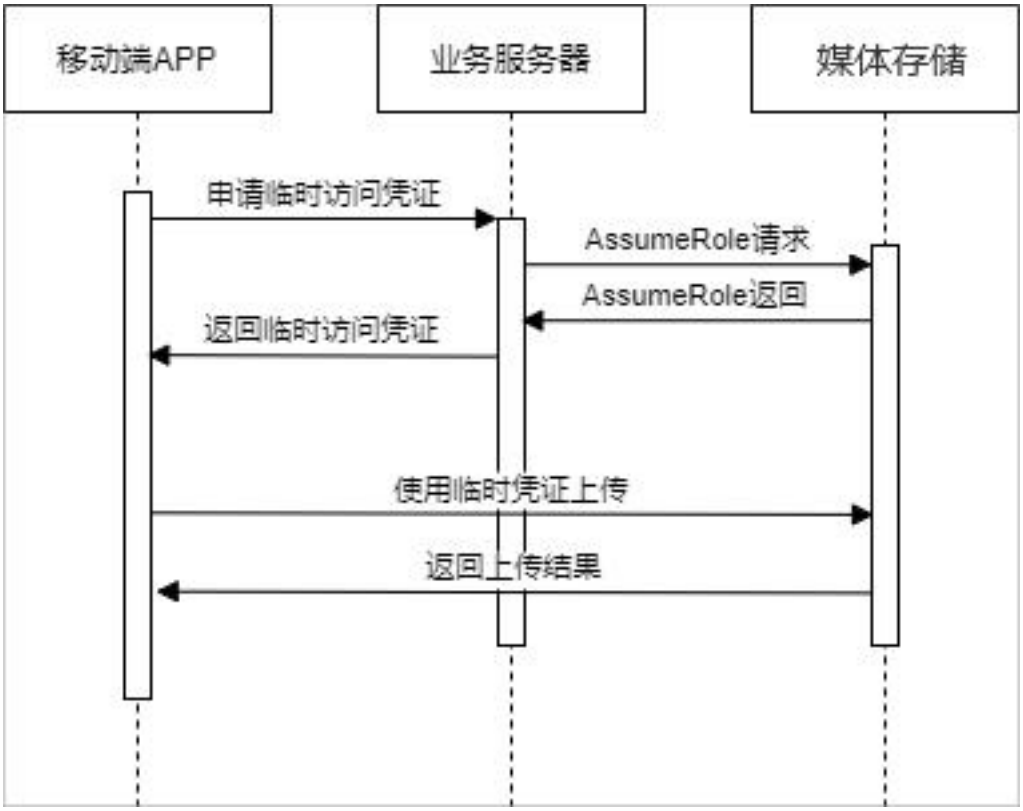
在移动互联网时代，从手机里的照片到各类文件，移动端 APP 需要上传到服务器的文件越来越多。开发者可以使用媒体存储来保存这些文件，媒体存储提供的 SDK 接口可以支持直接在移动端进行文件上传。

访问媒体存储需要使用密钥（AK/SK），但是如果在移动端直接使用长期密钥访问对象存储，遭受黑客攻击就可能会暴露长期密钥，导致对象存储中的文件泄露或被篡改，存在很大的风险。

媒体存储提供 STS 角色管理功能，可以为移动端颁发一个自定义时效和权限的访问凭证，无需在移动端暴露长期密钥。使用 STS 授权访问时，请务必按照业务情况，以最细粒度的权限原则进行授权，避免放大临时用户的权限，保证资源访问安全。

应用流程

使用临时凭证直传时，具体应用流程如下：



实践步骤

创建用于获取 STS 访问凭证的角色

通过媒体存储控制台，创建 STS 角色，并获取对应的 arn 信息，具体可参考 STS 角色管理 。

对应角色授权并且获取 STS 临时密钥

具体可参照如下 java 示例：

```
// STS endPoint
String endPoint = "<sts-endpoint>";
// 在对象存储控制台访问密钥 AccessKey 和 SecretKey。
String accessKey = "<access-key>";
String secretKey = "<secret-key>";
// 填写步骤一获取的角色 ARN。
String roleArn = "<role-arn>";
// 设置临时访问凭证的名称.
```

```
String roleSessionName = "<session-name>";
// 设置 Policy 允许上传对象
String policy = "{\"Version\":\"2012-10-17\"," + "\"Statement\":[" + "{\"Effect\":
\"Allow\","
                + "\"Action\":[\"s3:PutObject\"],"
                + "\"Resource\":[\"arn:aws:s3:::<bucket-name>/*\"]}" + "]}";
// 创建 STS Client
BasicAWSCredentials basicAWSCredentials = new BasicAWSCredentials(accessKe
y, secretKey);
AwsClientBuilder.EndpointConfiguration endpointConfiguration = new AwsClien
tBuilder.EndpointConfiguration(endPoint, "");
AWSSecurityTokenService stsClient = AWSSecurityTokenServiceClientBuilder.stan
dard()
        .withCredentials(new AWSStaticCredentialsProvider(basicAWSCredentials))
        .withEndpointConfiguration(endpointConfiguration)
        .build();
AssumeRoleRequest assumeRoleRequest = new AssumeRoleRequest();
assumeRoleRequest.setRoleArn(roleArn);
assumeRoleRequest.setRoleSessionName(roleSessionName);
assumeRoleRequest.setPolicy(policy);
AssumeRoleResult assumeRoleRes = stsClient.assumeRole(assumeRoleRequest);
Credentials stsCredentials = assumeRoleRes.getCredentials();
System.out.println("Expiration: " + stsCredentials.getExpiration());
System.out.println("Access Key Id: " + stsCredentials.getAccessKeyId());
```

```
System.out.println("Access Key Secret: " + stsCredentials.getSecretAccessKey());
System.out.println("Security Token: " + stsCredentials.getSessionToken());
```

通过临时密钥访问对象存储资源

本文以 Android 与 IOS 应用为例。

- Android

```
public class MyCredentialsProvider implements AWSCredentialsProvider {
    private AWSCredentials credentials;

    public MyCredentialsProvider(String ak, String sk, String token) {
        this.credentials = new BasicSessionCredentials(ak, sk, token);
    }

    public synchronized AWSCredentials getCredentials() {
        return credentials;
    }

    public synchronized void refresh() {
    }

    // 更新 ak,sk,token
    public synchronized void updateCred(String ak, String sk, String token) {
        this.credentials = new BasicSessionCredentials(ak, sk, token);
    }
}
```

```
String accessKey = "<your-access-key>";
String secretKey = "<your-secret-access-key>";
String endPoint = "<your-endpoint>";
String sessionToken = "<your-session-token>";

MyCredentialsProvider credProvider = new MyCredentialsProvider(accessKey, secretKey, sessionToken);
ClientConfiguration clientConfig = new ClientConfiguration();
clientConfig.setProtocol(Protocol.HTTP);
AmazonS3Client mS3Client = new AmazonS3Client(credProvider, clientConfig);
mS3Client.setEndpoint(endPoint);
```

- IOS

```
#define ACCESS_KEY @"<your-access-key>"
#define SECRET_KEY @"<your-secret-key>"
#define ENDPOINT @"<your-endpoint>"
#define SESSION_TOKEN @"<your-session-token>"

-(id)initWithToken {
    if (self = [super init]) {
        AWSBasicSessionCredentialsProvider *credentialsProvider = [[AWSBasicSessionCredentialsProvider alloc] initWithAccessKey:ACCESS_KEY secretKey:SECRET_KEY sessionToken:SESSION_TOKEN];

        AWSEndpoint *endPoint = [[AWSEndpoint alloc] initWithURLString:ENDPOINT];
```

```
        AWSServiceConfiguration *configuration = [[AWSServiceConfiguration a
lloc]
                                                  initWithRegion:AWSRegionUS
East1
                                                  endpoint:endPoint
                                                  credentialsProvider:credential
sProvider];
        [AWSServiceManager defaultServiceManager].defaultServiceConfiguratio
n = configuration;


        self.s3 = [AWSS3 defaultS3];
    }


    return self;
}
```

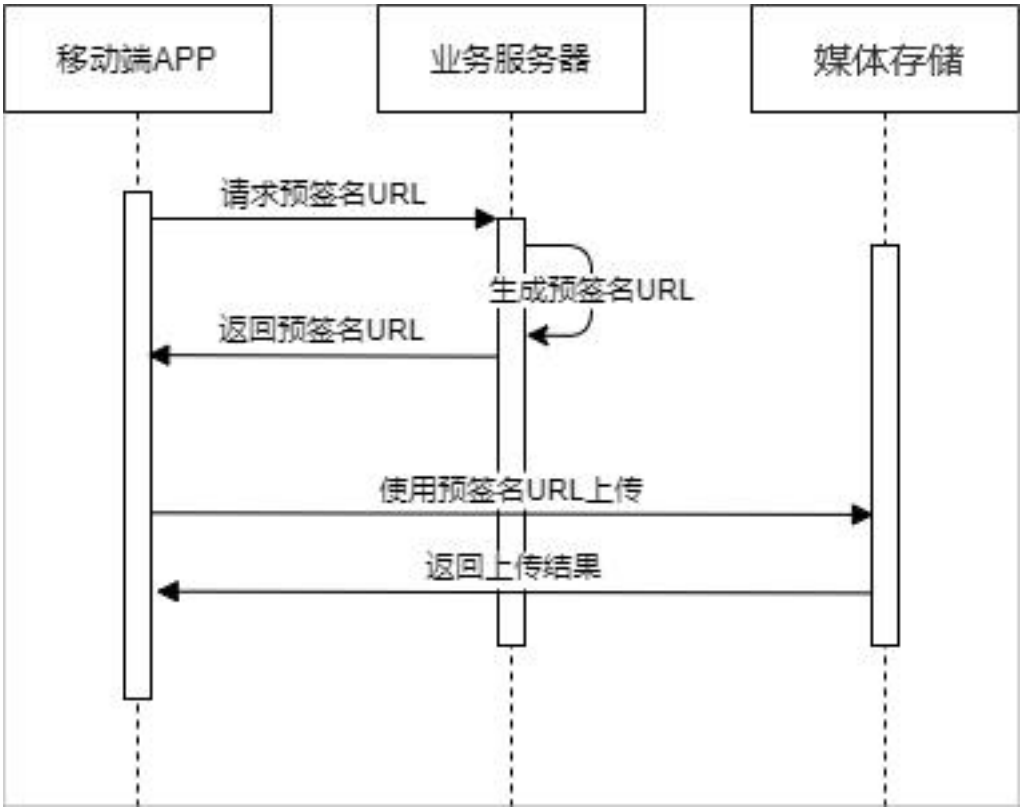
2.2.4.5 使用预签名 URL 直传媒体存储

实践背景

对象存储 SDK 提供预签名接口可以生成预签名 URL，通过预签名 URL，移动端 APP 可以直接上传或者下载文件。不需要使用 SDK 和密钥，使用 HTTP 接口就可以进行文件的上传和下载。

应用流程

使用预签名 URL 直传媒体存储应用流程如下：



实践步骤

生成预签名 URL

业务服务器配置长期密钥，调用预签名接口生成预签名 URL，具体可参照如下 java 示例。如需其他语言 SDK 示例，可参考：SDK 概览 。

- 上传预签名

```
String bucketName = "<your-bucket-name>";
String objectKey = "<your-object-key>";
LocalDateTime expirationDateTime = LocalDateTime.now().plusSeconds(5 * 60); //
url 的有效时间 5 分钟
Date expiration = Date.from(expirationDateTime.atZone(ZoneId.systemDefault()).t
oInstant());
try {
    GeneratePresignedUrlRequest generatePresignedUrlRequest = new GeneratePr
esignedUrlRequest(bucketName, objectKey)
```

```
            .withMethod(HttpMethod.PUT)
            .withExpiration(expiration);
    URL url = s3.generatePresignedUrl(generatePresignedUrlRequest);
} catch (AmazonServiceException e) {
    System.err.println(e.getErrorMessage());
}
```

- 下载预签名

```
String bucketName = "<your-bucket-name>";
String objectKey = "<your-object-key>";
try {
    GeneratePresignedUrlRequest generatePresignedUrlRequest = new GeneratePr
esignedUrlRequest(bucketName, objectKey)
            .withMethod(HttpMethod.GET)
            .withExpiration(expiration);
    URL url = s3.generatePresignedUrl(generatePresignedUrlRequest);
} catch (AmazonServiceException e) {
    System.err.println(e.getErrorMessage());
}
```

使用预签名 URL 直传

移动端通过业务服务器获取到预签名 URL，使用 HTTP 接口和预签名 URL 上传或者下载文件。具体可参照如下示例。

- 上传

```
Log.i(TAG, "upload");
try {
    OkHttpClient httpClient = new OkHttpClient.Builder()
```

```
            .followRedirects(false)
            .retryOnConnectionFailure(false)
            .cache(null)
            .build();

    MediaType mediaType = MediaType.parse("text/plain");
    RequestBody body = RequestBody.create("file content", mediaType);
    Request httpRequest = new Request.Builder()
            .url(url)
            .put(body)
                .build();

    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    Log.i(TAG, "Status:" + res.code());
    if (res.header("ETag") != null) {
        Log.i(TAG, "ETag:" + res.header("ETag"));
    }
    res.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

- 下载

```
private void download(String url) {
    Log.i(TAG, "download");
```

```
    try {
        OkHttpClient httpClient = new OkHttpClient.Builder()
                .followRedirects(false)
                .retryOnConnectionFailure(false)
                .cache(null)
                .build();

        Request httpRequest = new Request.Builder()
                .url(url)
                .get()
                .build();
        Call c = httpClient.newCall(httpRequest);
        Response res = c.execute();
        Log.i(TAG, "Status:" + res.code());
        if (res.body() != null) {
            Log.i(TAG, "Content:" + res.body().string());
        }
        res.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

2.2.4.6 使用 Java SDK 实现断点续传

当上传大文件时，经常出现因网络不稳定或程序崩溃导致上传失败的情况。失败后再次重新上传不仅浪费资源，而且当网络不稳定时仍然有上传失败的风险。断点续传上传接口

uploadFile 能有效地解决此类问题引起的上传效率低下的问题。其原理是将待上传的文件分成若干个分片分别上传，如果出现网络异常或程序崩溃导致文件上传失败时，会将中断对象的断点处记录下来，从而能在失败重传时继续上传未上传完成的部分，节省资源提高效率，还因其能够对分片进行并发上传的机制能加快上传速度。

本文主要介绍通过 Java SDK 实现断点续传。SDK 下载地址：SDK 概览。

注意事项

- 断点续传上传，您必须必须是桶拥有者或拥有上传对象的权限，才能上传对象。
- 断点续传上传接口传入的文件大小至少要 5MB 以上，因为最小的分片大小就是 5MB。
- 使用SDK的断点续传接口时，必须开启断点续传选项 *setEnableCheckpoint*为 *true* 才能在再次上传同一对象时读取到之前的上传进度。
- 您可实现 *ProgressListener* 接口来实现对上传进度的监控。

示例代码

断点续传

以下为断点续传示例代码。

```
public class ResumeUploadDemo {
    private static String endpoint = "https://gdoss.xstore.ctyun.cn";//资源池 endpoint，示例以断广东资源池 1 区为例，其他资源池请根据实际情况填写
    private static String accessKeyId = "ak";// 主账号或子账号 ak
    private static String accessKeySecret = "sk";// 主账号或子账号 sk
    private static String bucketName = "bucket";// 上传对象的目标 bucket
    private static String key = "xx.xx";// 对象名
    private static String uploadFile = "xx.xx";// 本地待上传文件路径

    public static void main(String[] args) {
```

```
        ClientConfiguration clientConfiguration = new ClientConfiguration();
        BasicAWSCredentials credentials = new BasicAWSCredentials(accessKeyI
d, accessKeySecret);
        AWSStaticCredentialsProvider credProvider = new AWSStaticCredentials
Provider(credentials);
        AwsClientBuilder.EndpointConfiguration endpointConfiguration = new
AwsClientBuilder.EndpointConfiguration(
                endpoint, Regions.DEFAULT_REGION.getName());

        AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                .withCredentials(credProvider)
                .withClientConfiguration(clientConfiguration)
                .withEndpointConfiguration(endpointConfiguration)
                .withPathStyleAccessEnabled(false)
                .build();

        try {
            // 通过 UploadFileRequest 设置多个参数。
            // 依次填写 Bucket 名称以及对象名称。
            UploadFileRequest uploadFileRequest = new UploadFileRequest(bu
cketName, key);
            // 指定监听器，当您实现以上接口后，可在这设置您的进度监控实例，从而
完成对于上传进度的监控。
            uploadFileRequest.setProgressListener(progressListener);
            // 填写本地文件的完整路径，例如 D:\\localpath\\examplefile.txt。如果未
```

```
指定本地路径，则默认从示例程序所属项目对应本地路径中上传文件。
            uploadFileRequest.setUploadFile(uploadFile);
            // 指定上传并发线程数，默认值为 1。
            uploadFileRequest.setTaskNum(5);
            // 指定上传的分片大小，单位为字节。默认值为 5 MB。
            uploadFileRequest.setPartSize(1024 * 1024 * 5);
            // 开启断点续传，默认关闭。开启后，上传过程中的进度信息会保存在文件中，默认与待上传的本地文件同路径，名称为${uploadFile}.ucp，如果某一分片上传失败，再次上传时会根据文件中记录的点继续上传。上传完成后，该文件会被删除。
            uploadFileRequest.setEnableCheckpoint(true);

            try {
                UploadFileResult uploadFileResult = s3Client.uploadFile(uploadFileRequest);
                CompleteMultipartUploadResult multipartUploadResult = uploadFileResult.getMultipartUploadResult();
                System.out.println(multipartUploadResult);
            } catch (Throwable e) {
                throw new RuntimeException(e);
            }
        } catch (Exception e) {
            System.err.println("Upload failed: " + e.getMessage());
            e.printStackTrace();
        }
```

```
    }
}
```

进度监控接口

通过实现上传过程监控接口，从而可以在上传过程中掌握您的大文件上传进度。常用的监听事件有：

- REQUEST_CONTENT_LENGTH_EVENT（要在请求中发送的对象内容长度的事件）
- TRANSFER_STARTED_EVENT（上传开始事件）
- TRANSFER_PART_STARTED_EVENT（开始上传分片事件）
- TRANSFER_PART_COMPLETED_EVENT（分片上传完毕事件）
- TRANSFER_COMPLETED_EVENT（上传完毕事件）
- TRANSFER_FAILED_EVENT（上传失败事件）
- TRANSFER_PART_FAILED_EVENT（上传分片失败事件）

当有相应场景发生时 *progressChanged*，则会产生相应的事件回调，即可调用您的监控代码从而掌握对象的上传进度。

以下为进度监控代码。

```
ProgressListener progressListener = new ProgressListener() {
    private long totalBytes = -1;
    private long transferredBytes = 0;
    private long startTime = System.currentTimeMillis();
    private final Object lock = new Object(); // 用于线程安全更新进度

    @Override
    public void progressChanged(ProgressEvent event) {
        switch (event.getEventType()) {
            case REQUEST_CONTENT_LENGTH_EVENT:
```

```
                totalBytes = event.getBytes();
                System.out.printf("总大小: %,d bytes%n", totalBytes);
                break;
            case TRANSFER_STARTED_EVENT:
                System.out.println("[开始] 文件传输启动");
                startTime = System.currentTimeMillis();
                break;
            case CLIENT_REQUEST_SUCCESS_EVENT:
                System.out.printf("[秒传] 文件已存在服务端 (大小: %.2fMB)%n", ev
ent.getBytes() / 1024.0 / 1024);
                break;
            case TRANSFER_PART_STARTED_EVENT:
                long encodedStart = event.getBytes();
                int partNumber = (int) (encodedStart >> 32);
                long partSize = encodedStart & 0xFFFFFFFFL;
                System.out.printf("[分片] #%d 开始上传 (大小: %.2fMB)%n",
                partNumber, partSize / 1024.0 / 1024);
                break;
            case TRANSFER_PART_COMPLETED_EVENT:
                long encodedComplete = event.getBytes();
                long actualBytes = encodedComplete & 0xFFFFFFFFL;
                synchronized (lock) {
                    transferredBytes += actualBytes;
                    }
                printProgress();
                break;
```

```
            case TRANSFER_COMPLETED_EVENT:
                System.out.println("\n[完成] 所有分片上传成功");
                printFinalStats();
                break;
                case TRANSFER_FAILED_EVENT:
                System.err.println("\n[失败] 文件传输异常终止");
                printFinalStats();
                break;
            case TRANSFER_PART_FAILED_EVENT:
                long encodedFail = event.getBytes();
                int failedPart = (int) (encodedFail >> 32);
                System.err.printf("[异常] 分片 #%d 上传失败%n", failedPart);
                break;
                }
        }

        private void printProgress() {
            synchronized (lock) {
                if (totalBytes <= 0)
                    return;

                double percent = transferredBytes * 100.0 / totalBytes;
                System.out.printf("\r[进度] %.2f%% - %.2fMB/%.2fMB",
                        percent,
                        transferredBytes / 1024.0 / 1024,
                        totalBytes / 1024.0 / 1024);
```

```
                }
            }

            private void printFinalStats() {
                long endTime = System.currentTimeMillis();
                double elapsed = (endTime - startTime) / 1000.0;
                double speed = (transferredBytes / 1024.0 / 1024) / elapsed;

                System.out.printf("耗时: %.1fs | 平均速度: %.1fMB/s%n", elapsed,
speed);
            }
        };
```

2.3 块存储

2.3.1 Linux 主机挂载

实践背景

块空间使用的是 iSCSI 协议，因此在新建块空间后需要使用 iSCSI 客户端来连接块空间。

本实践是在 centos 7 上，具有免密 sudo 权限的普通用户使用 iSCSI 客户端连接天翼云媒体存储块空间，并且对其进行格式化的过程。

操作步骤

1.执行以下命令，安装 iSCSI 客户端。

```
sudo yum install -y iscsi-initiator-utils
sudo yum install -y device-mapper-multipath
```

当系统出现如下所示的更新完毕-作为依赖被升级-完毕的提示时，说明软件安装完成。

```
更新完毕:
  iscsi-initiator-utils.x86_64 0:6.2.0.874-10.el7

作为依赖被升级:
  iscsi-initiator-utils-iscsiuio.x86_64 0:6.2.0.874-10.el7

完毕!
```

```
更新完毕:
  device-mapper-multipath.x86_64 0:0.4.9-123.el7

作为依赖被升级:
  device-mapper-multipath-libs.x86_64 0:0.4.9-123.el7                    kpartx.x86_64 0:0.4.9-123.el7

完毕!
```

若已安装过所需软件，系统会提示对应的软件包已安装并且是最新版本无须任何处理。

2.配置 iSCSI 多路径，具体步骤如下：

（1） 执行以下命令，生成配置文件/etc/multipath.conf。

```
mpathconf --enable --with_multipathd y
```

（2）执行以下命令，修改多路径配置。

```
sudo vi /etc/multipath.conf
```

（3）添加如下内容。

```
defaults {

        user_friendly_names yes

        path_grouping_policy failover

        failback immediate

        no_path_retry fail

}
```

```
devices {
        device {
                vendor  "LIO-ORG"
                hardware_handler  "1  alua"
                path_grouping_policy  "failover"
                path_selector  "queue-length  0"
                failback  60
                path_checker  tur
                prio  alua
                prio_args  exclusive_pref_bit
                fast_io_fail_tmo  25
                no_path_retry  queue
        }
```

```
        device {

                vendor "CTyun"

                path_grouping_policy "failover"

                path_selector "queue-length 0"

                failback 60

                path_checker tur

                prio_args exclusive_pref_bit

                fast_io_fail_tmo 25

                no_path_retry queue

        }

}
```

（4）执行以下命令，进行服务重启。

```
sudo systemctl restart multipathd
```

3.修改 iSCSI Client 的 InitiatorName，具体步骤如下：

（1）运行以下命令。

```
sudo  vi  /etc/iscsi/initiatorname.iscsi
```

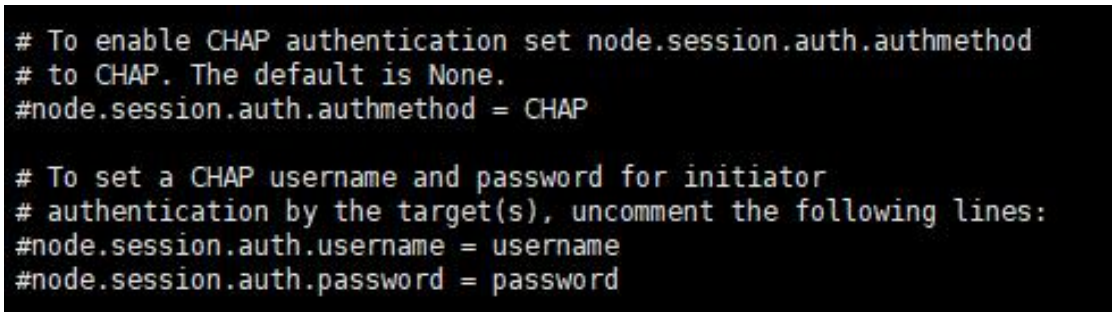
（2）将 InitiatorName（下图红框内内容）修改为创建块空间时填写的 CHAP iqn；具体可参考操作步骤【块空间管理】查看所需挂载块空间的 CHAP iqn 信息（图中的例子为 iqn.2099-01.com.client.cicd-testcy:230323）。

4.修改 CHAP 权限，具体步骤如下：

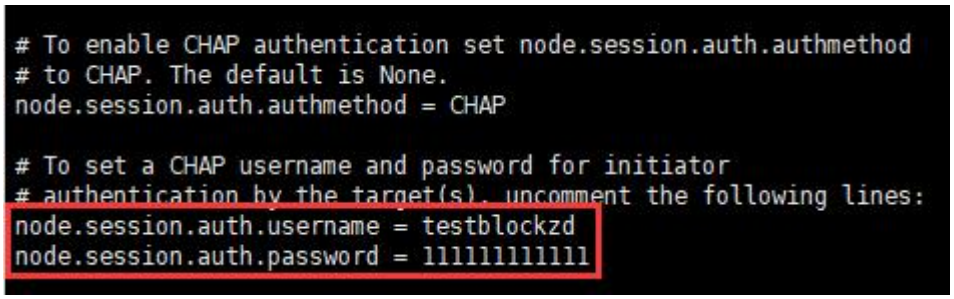
（1）运行以下命令。

```
sudo  vi  /etc/iscsi/iscsid.conf
```

（2）找到以下内容：



修改为下图中内容，其中红框内的 username 的等号后面填写创建块空间时设定的 CHAP 用户（图中的例子为 testblockzd），password 的等号后面填写创建块空间时设定的 CHAP 密钥（图中的例子为 111111111111）。



1.连接块空间，具体步骤如下：

（1）为确保连接顺利，先执行 setenforce 0 命令临时禁用防火墙。

（2）执行以下命令，重启相关组件。

```
sudo  systemctl  restart  iscsid
```

（3）执行以下命令增加 iSCSI 连接 target，具体的 targetname 参数与 portal 参见块空间信息中的 targetIqn、网关地址与数据端口，具体查看步骤可见操作步骤【块空间管理】。

```
sudo iscsiadm -m node --targetname=iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw
 --portal=14.211.109.226:13260 --op=new
sudo iscsiadm -m node --targetname=iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw
 --portal=14.211.109.226:23260 --op=new
```

（4）执行以下命令，进行登入操作。

```
sudo iscsiadm -m node -T iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw -l
```

✧ **说明**

对于步骤 5-(3)、5-(4)，可直接通过块空间详情复制对应的操作命令，具体界面如图所示：

更多块空间管理操作可参考：块空间管理。

（5）若系统出现以下提示表示登录成功。

```
Login to [iface: default, target: iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw, por
tal: 14.215.109.226,13260] successful.
Login to [iface: default, target: iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw, por
tal: 14.215.109.226,23260] successful.
```

（6）可通过以下命令查看新增磁盘。

```
sudo lsscsi
sudo fdisk -l
sudo lsblk
sudo multipath -l
```

注意：

- 执行后会发现除了机器的原有磁盘外，还挂载了三个新的磁盘，一个是以/dev/mapper/开头的磁盘，另外两个是以/dev/sd 开头的磁盘，我们挂载的是前者，即是以/dev/mapper/的磁盘。
- 每次挂载，该磁盘后面的名字都可能不一样。
- 如果你有多个块空间使用了同一个 iqn，每个块在查看磁盘时均会看到一个以/dev/mapper/开头的磁盘以及两个以/dev/sd 开头的磁盘，请总是使用以/dev/mapper/开头的磁盘来进行挂载。

（7）根据系统协议执行以下命令，进行磁盘格式化：

- xfs 文件系统：

```
sudo  mkfs.xfs  /dev/mapper/mpatha
```

- ext4 文件系统：

```
sudo  mkfs.ext4  /dev/mapper/mpatha
```

注意 ：可以根据实际需求选择命令，将命令中“/dev/mapper/mpatha”部分改为步骤（6）中查找到的新增磁盘地址。

（8）执行以下命令，挂载到本地目录。

```
sudo  mount  /dev/mapper/mpatha  /mnt/iscsiMnt
```

注意：

- mount 后第一部分为新增磁盘的实际地址。
- mount 后第二部分为进行挂载的本地目录地址。
- 可以通过 mount -l 命令查看是否挂载成功，若显示信息中包含挂载信息则挂载成功，以上样例对应的挂载信息如下，该样例已成功挂载。

```
/dev/mapper/mpatha on /mnt/iscsiMnt type ext4 (rw,relatime,seclabel,stripe=16,data=ordered)
```

（9）执行以下命令，查看已连接的目标。

```
sudo  iscsiadm  -m  session
```

6.如需断开连接，则执行以下命令。

```
sudo iscsiadm -m node -T iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw -u
```

7.执行以下命令，删除所有记录。

```
sudo iscsiadm -m node --targetname=iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw
 --portal=14.211.109.226:13260 --op=delete
sudo iscsiadm -m node --targetname=iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw
 --portal=14.211.109.226:23260 --op=delete
```

✧ **说明**

对于步骤 6、7，可直接通过块空间详情复制对应的操作命令，具体界面如图所示：

更多块空间管理操作可参考：块空间管理。

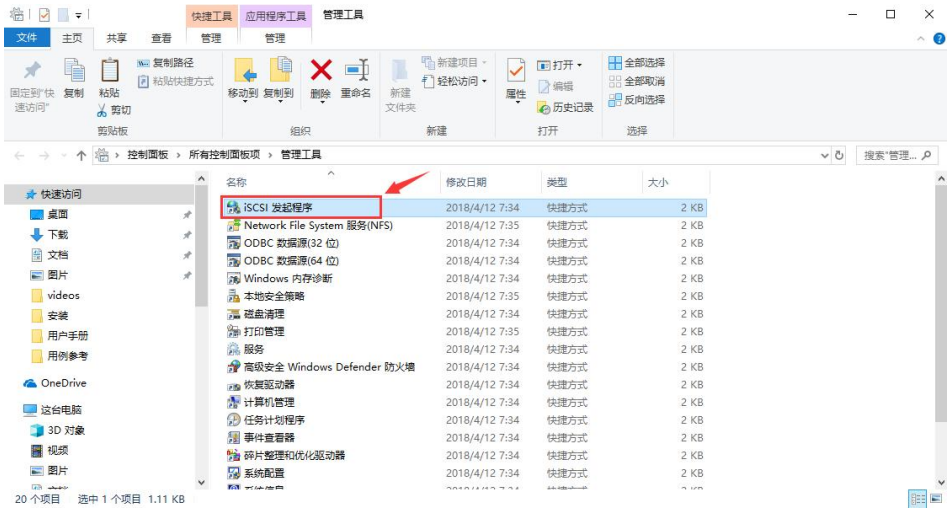
2.3.2 Windows 主机挂载

实践背景

使用 iSCSI 协议的块设备也可以在 windows 操作系统下使用，下面将以 win10 为例连接天翼云媒体存储块设备，其他版本的操作系统类似。

操作步骤

1.打开控制面板->管理工具-> iSCSI 发起程序。



2.修改 iSCSI 发起程序中"配置"页下的发起程序名称为媒体存储控制台块设备创建时填写的 CHAP iqn（本例以 iqn.2099-01.com.client.cicd-testcy:230323 为例）；

3.右键开始菜单，管理员模式下运行命令行程序，如下图。

并在命令行中输入如下命令(所有*都需要)：

iscsicli AddTarget iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw 14.211.109.226 13260 * * * * * * * * * * * *

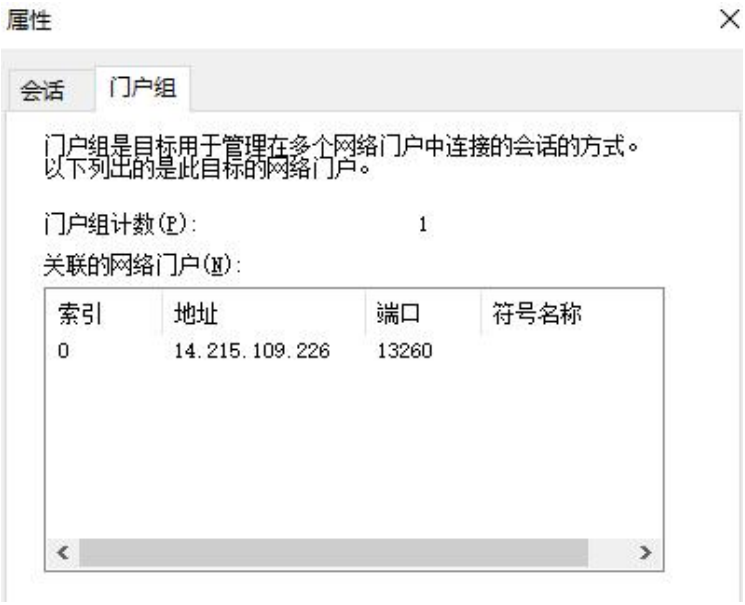
✧ **说明**

1. 以上示例中： iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw 是 TargetName、TargetAlias，对应 TargetIqn，14.211.109.226 是网关地址，13260 是数据端口。挂载时，您需要对应替换成需要挂载的块空间信息。
2. 你也可以可直接通过块空间详情复制对应的操作命令，具体界面如下图所示。如您通过控制台复制挂载命令进行操作，此步操作完成后，可直接跳转第 5 步查看新增的磁盘信息。

3. 更多块空间管理操作可参考：块空间管理。

结果如下：

并且在"属性"中的"门户组"页下可以看到关联的网络门户为 14.211.109.226:13260。

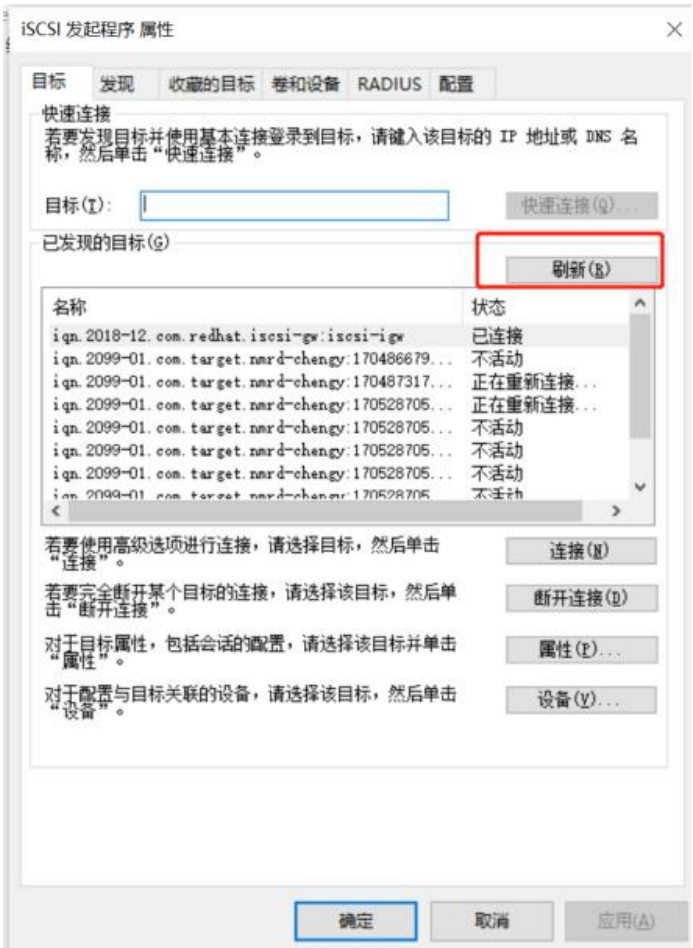


4.连接到目标网关：点击【连接】按钮，然后点击【高级】，再勾选【启用 CHAP 登录】，并输入名称和目标机密，最后点击【确定】，就会进行连接，如下图所示。

这里具体的参数可参考：块空间管理，获取所需挂载块空间的 CHAP 用户与 CHAP 密钥。

1.实现连接后，可以在计算机管理->磁盘管理中找到新加入的网络盘，如下图，然后进行磁盘的初始化工作。初始化完成后，即可在资源管理器中看到新增的设备，可进行正常的文件系统操作。

6.断开连接：点击 iSCSI 发起程序目标下面的刷新，在已连接磁盘上右键点击"断开连接"，或者在命令行中输入命令：iscsicli SessionList ，找到连接 ID；再输入：iscsicli LogoutTarget，完成断开操作。



7. 删除连接，在命令行中输入：iscsicli RemoveTarget iqn.2018-10.com.redhat.iscsi-gw:iscsi-igw ，完成删除操作。

2.4 文件存储

2.4.1 NFS 协议挂载

挂载说明

NFS 协议的文件空间推荐在 Linux 系统下挂载。需要使用 NFS 客户端连接。

以下操作除天津资源池 2 区、天津资源池 3 区外适用

以下是在 centos 7 上安装 NFS 客户端并且挂载天翼云媒体存储文件资源过程。操作步骤如下：

1. 执行以下命令，在主机上创建挂载目录：*mkdir 本地挂载目录*。
2. 执行以下命令安装 NFS 工具包：*sudo yum install -y nfs-utils*。
3. 执行以下命令进行资源连接：*sudo mount -t nfs4 直连模式资源挂载地址:/mnt/媒体存储控制台用户名/文件系统名称 本地挂载目录*。
4. 也可以通过控制台获取挂载命令，点击对应文件空间的【查看】按钮，并在详情弹窗获取挂载命令。

5. 在本地挂载目录下可以进行正常的文件操作。

6. 如需卸载，可使用以下命令：*sudo umount 本地挂载目录*。

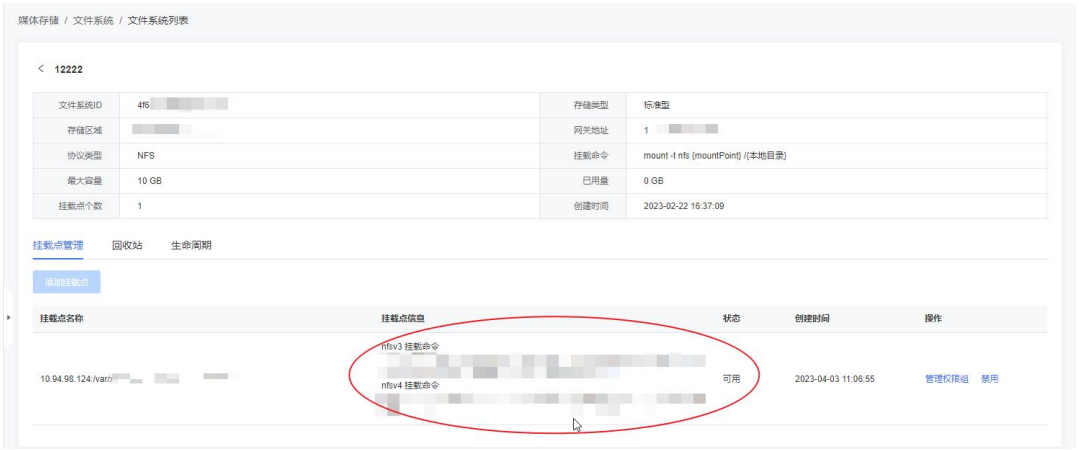
以下操作适用于天津资源池 2 区、天津资源池 3 区

以下是在 centos 7 上安装 NFS 客户端并且挂载天翼云媒体存储文件资源过程。操作步骤如下：

1. 执行以下命令，在主机上创建挂载目录：*mkdir 本地挂载目录*。

2. 执行以下命令安装 NFS 工具包：*sudo yum install -y nfs-utils*。

3. 执行挂载命令进行资源连接，具体的挂载命令可在对应的文件系统点击【管理】，从“挂载点信息“字段获取。建议优先使用 NFSv4 挂载命令。



4. 在本地挂载目录下可以进行正常的文件操作。

5. 如需卸载，可使用以下命令：*sudo umount 本地挂载目录*，完成卸载操作。

2.4.2 CIFS 协议挂载

挂载说明

本章节挂载操作除天津资源池 2 区、天津资源池 3 区外适用。

Linux 系统挂载

Linux 使用 CIFS 文件资源需要使用 CIFS 客户端连接。以下是在 centos 7 上安装 CIFS 客户端并且挂载的过程。操作步骤如下：

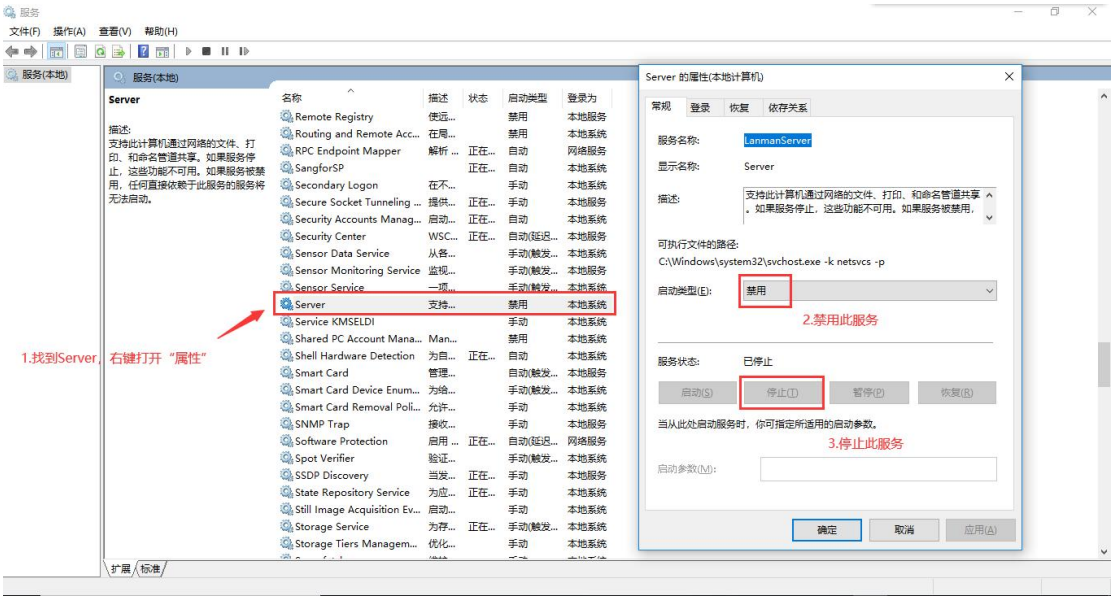
1. 安装 CIFS 客户端：*yum install cifs-utils*。
2. 执行以下命令，在主机上创建挂载目录：*mkdir 本地挂载目录*。
3. 执行以下命令进行资源连接：sudo mount -v -t cifs -o username="cifsUser",password="password" //直连模式资源挂载地址/媒体存储控制台用户名_文件系统名称 本地挂载目录。
4. 在本地挂载目录下可以进行正常的文件操作。
5. 如需卸载，可使用以下命令：*sudo umount 本地挂载目录*。

温馨提示：步骤 3 的连接命令中，“媒体存储控制台用户名”可在对应的文件系统点击 【查看】 ，从“用户标识“字段获取。

Windows 系统挂载

通过直连模式使用的 CIFS 文件资源需要使用 Windows 系统内置的客户端。以下是在 win10 上连接 CIFS 文件资源的示例。操作步骤如下：

1. 出于安全性考虑，直连文件系统使用了非默认的 CIFS 服务端口，因此需要在您的电脑上进行一些设置方可进行正常的连接与使用。
2. 关闭 445 端口，services.msc 中找到 Server 的服务，属性禁用，然后停止服务（必要时可重启电脑）。



3. 设置端口转发，打开 cmd 窗口执行下列命令：*netsh interface portproxy add v4tov4 listenport=445 connectaddress=xxx.xxx.xxx.xxx connectport=9445*。其中 connectaddress 是对应文件空间的网关地址，可在对应的文件空间详情中获取，具体可参考：文件空间管理。
4. 右键点击"此电脑"，选择"映射网络驱动器"，在文件夹一项中按图例填写，勾选"登录时重新连接"和"使用其他凭证连接"，点击"完成"。

5. 输入网络凭证窗口，输入创建文件系统时设定的账号和密码，点击"确定"。

6. 连接成功后能够在此电脑-网络位置中查看到文件系统。

7. 在连接成功的文件系统中可以进行正常的文件操作。

8. 如需卸载，可在磁盘上点击右键，选择"断开连接"。

2.4.3 SMB 协议挂载

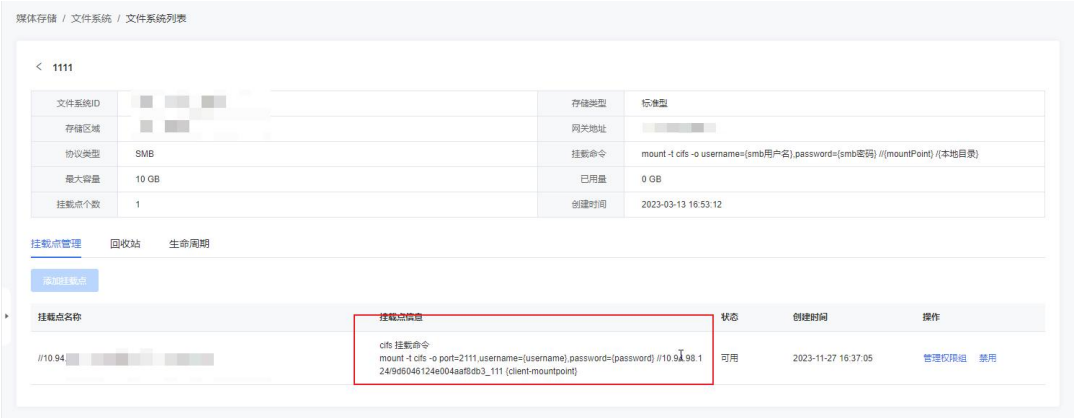
挂载说明

本章节挂载操作适用于天津资源池 2 区、天津资源池 3 区。

Linux 系统挂载

Linux 使用 SMB 文件资源需要使用 CIFS 客户端连接。以下是在 centos 7 上安装 CIFS 客户端并且挂载的过程。操作步骤如下：

1. 安装 CIFS 客户端：*yum install cifs-utils*。
2. 执行以下命令，在主机上创建挂载目录：*mkdir 本地挂载目录*。
3. 执行以下命令进行资源连接，具体的挂载命令可在对应的文件系统点击【管理】，从“挂载点命令“字段获取，username 与 password 需要根据具体的 SMB 用户填写。



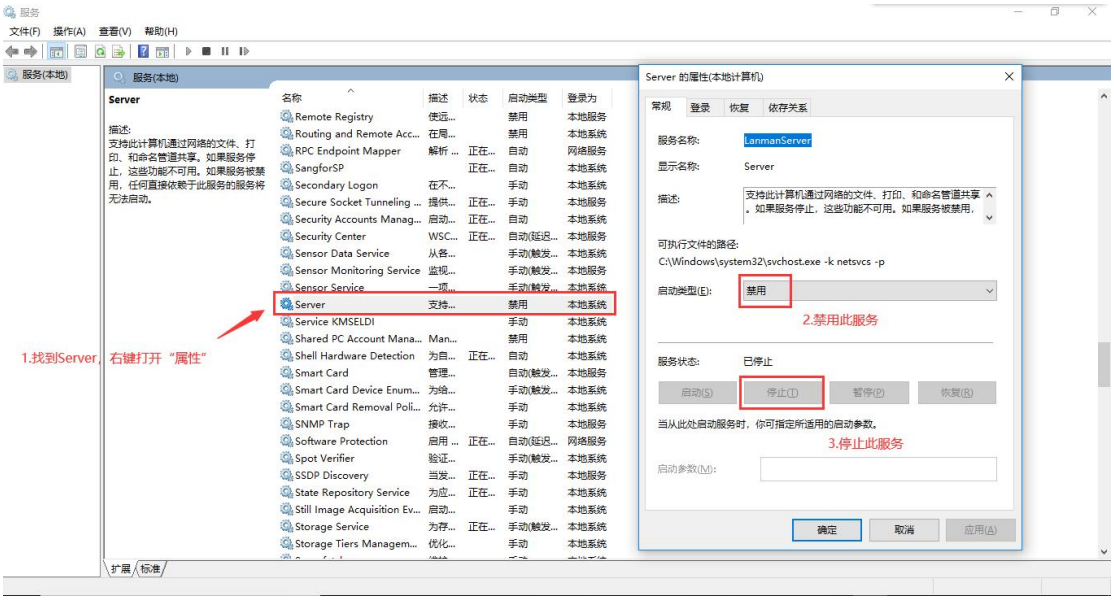
4. 在本地挂载目录下可以进行正常的文件操作。
5. 如需卸载，可使用以下命令：*sudo umount 本地挂载目录*。

Windows 系统挂载

通过直连模式使用的 CIFS 文件资源需要使用 Windows 系统内置的客户端。以下是在 win10 上连接 CIFS 文件资源的示例。操作步骤如下：

1. 出于安全性考虑，直连文件系统使用了非默认的 CIFS 服务端口，因此需要在您的电脑上进行一些设置方可进行正常的连接与使用。

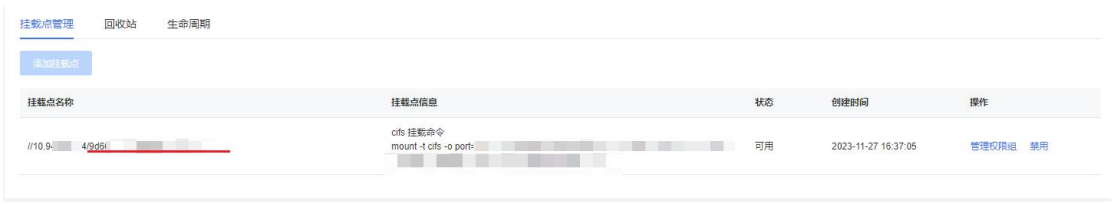
2. 关闭 445 端口，services.msc 中找到 Server 的服务，属性禁用，然后停止服务（必要时可重启电脑）。



3. 设置端口转发，打开 cmd 窗口执行下列命令(其中 connectaddress 为您服务提供地址)：*netsh interface portproxy add v4tov4 listenport=445 connectaddress=xxx.xxx.xxx.xxx connectport="port"*。 其中 connectaddress 是对应文件空间的网关地址，可在对应的文件空间详情中获取，具体可参考：文件空间管理。connectport 可查看挂载点信息获取（下图红框内容）。

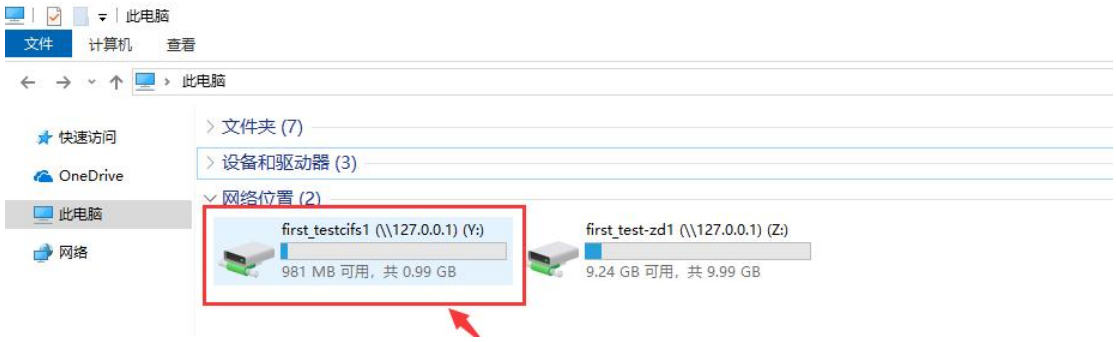
4. 右键点击"此电脑"，选择"映射网络驱动器"，在文件夹一项中填写：\127.0.0.1\媒体存储控制台用户_文件系统名称，勾选"登录时重新连接"和"使用其他凭证连接"，点击"完成"。

挂载点名称可查看挂载点名称获取（下图红线处内容）。



5. 输入网络凭证窗口，输入创建文件系统时设定的账号和密码，点击"确定"。

6. 连接成功后能够在此电脑-网络位置中查看到文件系统。



7. 在连接成功的文件系统中可以进行正常的文件操作。

8. 如需卸载，可在磁盘上点击右键，选择"断开连接"。

2.4.4 Windows 主机自动挂载 CIFS

本文档主要介绍 Windows 云主机开机自动挂载 cifs 文件系统的最佳实践。

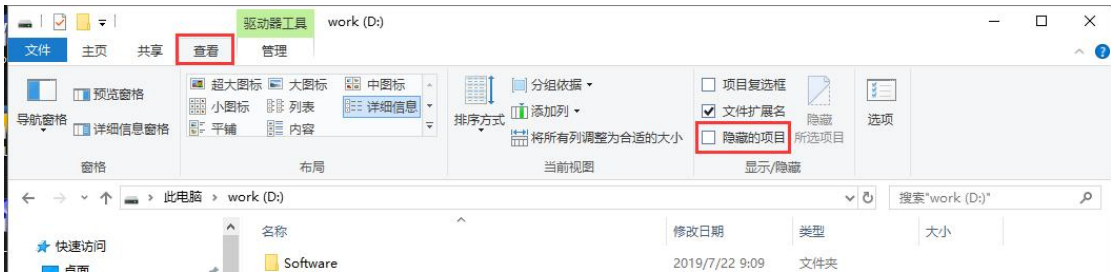
1.编写 xxx.bat 文件，具体示例如下：

```
@echo  off
net  use  z:  \\127.0.0.1\bssUserxxxxxxxxxx_file-test1  "testuser"/user:"testpassword
"
```

- z 为网络驱动器盘符。
- bssUserxxxxxxxxxx 为用户标识。
- testuser 为需要登录的 CIFS 账号。
- testpassword 为对应的 CIFS 密码。
- bssUserxxxxxxxxxx、testuser、testpassword 您均可以通过控制台在对应的文件系统点击 【查看】 ，从“用户标识“、 “CIFS 账号”、 “CIFS 密码”字段获取，如下图：

2. 将该文件放至目录 C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup 下，重启即可自动挂载。

3.文件系统中可能有隐藏目录，需要在文件“查看”属性中勾选“隐藏的项目”，如下图：



2.4.5 Linux 主机自动挂载 CIFS/NFS

如果您需要在系统启动时自动挂载文件系统，可以在/etc/fstab 中添加配置。

自动挂载 NFS 文件系统

在/etc/fstab 中添加一行配置：*{直连模式资源挂载地址}:/mnt/{控制台用户标识}/{文件系统名称} {本地挂载目录} nfs4 port=9049 0 0* ，添加后，系统启动时将自动挂载。

示例如下：

自动挂载 CIFS 文件系统

在/etc/fstab 中添加一行配置：*//{直连模式资源挂载地址}/{控制台用户标识}_{文件系统名称} {本地挂载目录} cifs port=9445,username={cifs 用户名},password={cifs 密码} 0 0*，添加后，系统启动时将自动挂载。

示例如下：

温馨提示

上述步骤中，控制台用户标识、文件系统名称、CIFS 用户名、CIFS 密码可以通过媒体存储控制台获取。

获取具体步骤如下：

1. 进入文件空间菜单页面，找到对应的文件系统，点击【查看】 。

2. 在弹窗页面找到控制台用户名、文件系统名称、cifs 用户名、cifs 密码。