

阿里云 对象存储 OSS

常用工具

文档版本：20180808

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 禁止： 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 警告： 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 说明： 您也可以通过按 Ctrl + A 选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
<code>courier</code> 字体	命令。	执行 <code>cd /d C:/windows</code> 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	<code>bae log list --instanceid Instance_ID</code>
[]或者[a b]	表示可选项，至多选择一个。	<code>ipconfig [-all -t]</code>
{ }或者{a b}	表示必选项，至多选择一个。	<code>swich {stand slave}</code>

目录

法律声明.....	I
通用约定.....	I
1 ossimport.....	1
1.1 常见问题.....	1
1.2 说明及配置.....	1
1.3 单机部署.....	12
1.4 分布式部署.....	16
1.5 数据迁移.....	20
2 ossbrowser.....	26
3 ossftp.....	33
3.1 如何快速安装OSS FTP.....	33
3.2 Discuz如何存储远程附件到OSS.....	38
3.3 Phpwind如何存储远程附件到oss.....	42
3.4 Wordpress如何存储远程附件到oss.....	45
3.5 如何结合RAM实现文件共享.....	49
4 ossfs.....	54
4.1 快速安装.....	54
4.2 FAQ.....	56
5 RAM策略编辑器.....	62
6 osscmd.....	64
6.1 快速安装.....	64
6.2 使用示例.....	66
6.3 有关Bucket命令.....	70
6.4 有关Object命令.....	74
6.5 有关Multipart命令.....	80
7 ossprobe.....	84
8 OSS常用工具汇总.....	89
9 ossutil.....	91
9.1 下载和安装.....	91
9.2 查看选项.....	97
9.3 有关bucket的命令.....	98
9.4 有关object的命令.....	101
9.5 有关multipart的命令.....	104

1 ossimport

1.1 常见问题

1.2 说明及配置

概述

OssImport工具可以将本地、其它云存储的数据迁移到OSS，它有以下特点：

- 支持的丰富的数据源，有本地、七牛、百度BOS、AWS S3、Azure Blob、又拍云、腾讯云COS、金山KS3、HTTP、OSS等，并可根据需要扩展；
- 支持断点续传；
- 支持流量控制；
- 支持迁移指定时间后的文件、特定前缀的文件；
- 支持并行数据下载、上传；
- 支持单机模式和分布式模式，单机模式部署简单使用方便，分布式模式适合大规模数据迁移。

运行环境

- Java 1.7及以上

部署方式

OssImport有单机模式和分布式模式两种部署方式。

- 对于小于 30TB 的小规模数据迁移，单机模式即可完成。[下载地址](#)
- 对于大规模的数据迁移，请使用分布式模式。[下载地址](#)

单机

Master、Worker、Tracker、Console运行在一个机器上，系统中有且只有一个Worker。

我们对单机模式的部署和执行进行了封装优化，单机部署和执行都很简单。单机模式下Master、Worker、TaskTracker、Console四个模块统一打包成ossimport2.jar。

单机模式下文件结构如下：

```
ossimport
├── bin
│   └── ossimport2.jar # 包括Master、Worker、Tracker、Console四个模块的
总jar
└── conf
```

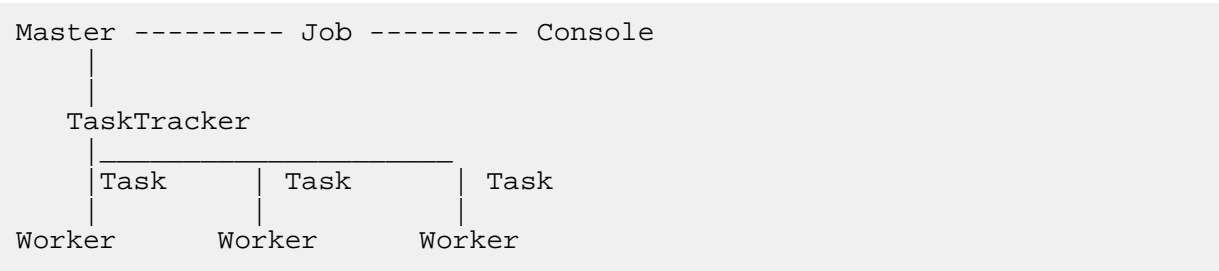
```
| |—— local_job.cfg    # 单机Job配置文件
| |—— sys.properties   # 系统运行参数配置文件
|—— console.bat        # Windows命令行，可以分布执行调入任务
|—— console.sh         # Linux命令行，可以分布执行调入任务
|—— import.bat         # Windows一键导入，执行配置文件为conf/local_job.
                        # cfg配置的数据迁移任务，包括启动、迁移、校验、重试
|—— import.sh          # Linux一键导入，执行配置文件为conf/local_job.
                        # cfg配置的数据迁移任务，包括启动、迁移、校验、重试
|—— logs               # 日志目录
|—— README.md          # 说明文档，强烈建议使用前仔细阅读
```

提示：

- import.bat/import.sh为一键导入脚本，修改完local_job.cfg后可以直接运行；
- console.bat/console.sh为命令行工具，可以用于分布执行命令；
- 脚本或命令时请在ossimport目录下执行，即 *.bat/*.sh 的同级目录。

分布式

OssImport是基于 master-worker 的分布式架构，如下图：



其中：

- Job：用户通过提交的数据迁移任务，对用户来说一个任务对应一个配置文件job.cfg。
- Task：Job按照数据大小和文件个数 可以分成多个 Task ，每个 Task 迁移部分文件。Job切分成 Task的最小单位是文件，同一个文件不会切分到多个Task中。

OssImport各模块说明，如下表：

角色	说明
Master	负责将Job切分成Task，按照数据大小和文件个数分解成Task，数据大小/文件个数可以在sys.properties中配置。Job切分成Task的详细过程是： 1. Master从本地/其它云存储中遍历出完整的待迁移的文件列表。

角色	说明
	2. 按照数据大小和文件个数把完整的文件列表切分成Task，每个Task负责部分文件的迁移或校验。
Worker	<ul style="list-style-type: none"> 负责Task的文件迁移和数据校验，从数据源上拉取指定文件，并上传到OSS的指定目录。迁移的数据源和OSS的配置在job.cfg或local_job.cfg中指定 Worker数据迁移支持限流、指定Task并发数，在sys.properties中配置。
TaskTracker	简称Tracker，负责Task的分发、Task状态跟踪。
Console	负责与用户交互，接受命令显示结果。支持系统管理命令deploy/start/stop，Job管理命令submit/retry/clean。

分布式模式下可以启动多个Worker执行迁移数据，Task平均分配到Worker上执行，一个Worker执行多个Task。每一个机器上只能启动一个Worker。workers配置的第一个Worker上会同时启动Master，TaskTracker，Console也要在该机器上运行。

分布式模式下文件结构如下：

```

ossimport
├── bin
│   ├── console.jar    # Console模块jar包
│   ├── master.jar     # Master模块jar包
│   ├── tracker.jar    # Tracker模块jar包
│   └── worker.jar     # Worker模块jar包
├── conf
│   ├── job.cfg        # Job配置文件模板
│   ├── sys.properties # 系统运行参数配置文件
│   └── workers        # Worker列表
├── console.sh         # 命令行工具，目前支持只Linux
├── logs               # 日志目录
└── README.md         # 说明文档，强烈建议使用前仔细阅读

```

提示：

- 分布式命令行工具 console.sh 目前只支持Linux，Windows暂不支持。

配置文件

单机模式下有两个配置文件`sys.properties`、`local_job.cfg`，分布式模式下有三个配置文件`sys.properties`、`local_job.cfg`、`workers`。其中`local_job.cfg`和`job.cfg`是完全一样的，只是名称不一样，`workers`是分布式环境先独有的。

- `sys.properties`

系统运行参数。

字段	含义	说明
<code>workingDir</code>	工作目录	<ul style="list-style-type: none">• 工具包解压后的目录。• 单机模式下请不要修改此选择。• 分布式模式下每个机器的工作目录必须相同。
<code>workerUser</code>	Worker机器的ssh用户名	<ul style="list-style-type: none">• 如果配置了<code>privateKeyFile</code>，则优先使用<code>privateKeyFile</code>。• 如果没有配置<code>privateKeyFile</code>，则使用<code>workerUser/workerPassword</code>。• 单机模式不需要修改此项。
<code>workerPassword</code>	Worker机器的ssh用户密码	单机模式不需要修改此项。
<code>privateKeyFile</code>	public key文件路径	<ul style="list-style-type: none">• 如果已经打通了ssh通道，则可以指定public key文件路径，否则为空。• 如果配置了<code>privateKeyFile</code>，则优先使用<code>privateKeyFile</code>。• 如果没有配置<code>privateKeyFile</code>，则使用<code>workerUser/workerPassword</code>。• 单机模式不需要修改此项。
<code>sshPort</code>	ssh端口	默认22，一般情况无需更改 单机模式不需要修改此项。

字段	含义	说明
workerTaskThreadNum	Worker执行Task的最大线程数	<ul style="list-style-type: none"> 该参数与机器的内存/网络有关，建议值60。 物理机可以适当加大，比如150，如果网络已经打满，请不要再加大。 如果网络较差，请适当降低，比如30，防止请求竞争网络造成大量请求超时。
workerMaxThroughput(KB/s)	worker数据迁移的流量上限	该值能起到限流作用，默认0表示不限流。
dispatcherThreadNum	Tracker的Task分发与状态确认的线程数	默认值一般够用，没有特殊需要请不要修改默认值。
workerAbortWhenUncatchedException	表示遇到未知错误时是跳过还是Abort	默认跳过。
workerRecordMd5	在OSS中是否使用元数据x-oss-meta-md5记录迁移文件MD5值，默认不记录。	主要用于用户使用MD5校验文件数据。

- job.cfg

数据迁移任务配置，local_job.cfg和job.cfg的配置项是完全一样的，只是名称不一样。

字段	含义	说明
jobName	任务名字，字符串。	<ul style="list-style-type: none"> 任务的唯一标识，命名规则 [a-zA-Z0-9_-]{4,128}，支持提交多个名字不同的任务。 如果重复提交同名任务会提示任务已存在，清理（clean）同名任务前，无法提交同名任务。
jobType	任务类型，字符串	两类import或audit，默认import。 <ul style="list-style-type: none"> import，执行数据迁移操作并校验迁移数据的一致性。

字段	含义	说明
		<ul style="list-style-type: none"> • <code>audit</code>，仅进行数据一致性校验。
<code>isIncremental</code>	是否打开增量迁移模式，布尔类型。	<ul style="list-style-type: none"> • 默认值<code>false</code>。 • 如果设为<code>true</code>，会每间隔<code>incrementalModelInterval</code> (单位秒)重新扫描一次增量数据，并将增量数据迁移到OSS。
<code>incrementalModelInterval</code>	增量模式下的同步间隔，整形，单位秒。	<code>isIncremental=true</code> 时有效。可配置的最小间隔为900秒，不建议配置成小于 3600秒 的值，会浪费大量请求，造成额外的系统开销。
<code>importSince</code>	迁移大于该时间的数据，整形，单位秒。	<ul style="list-style-type: none"> • 该时间为 Unix时间戳，即自1970年1月1日UTC零点以来的秒数，通过命令<code>date +%s</code>获取。 • 默认为0，表示迁移全部数据。
<code>srcType</code>	同步源类型，字符串， 请注意大小写。	<p>目前支持<code>local</code>、<code>oss</code>、<code>qiniu</code>、<code>bos</code>、<code>ks3</code>、<code>s3</code>、<code>youdao</code>十种类型。</p> <ul style="list-style-type: none"> • <code>local</code>，从本地文件迁移数据到OSS，该选项只需要填写<code>srcPrefix</code>，不需要填写<code>srcAccessKey</code>，<code>srcSecretKey</code>，<code>srcDomain</code>，<code>srcBucket</code>。 • <code>oss</code>，从一个 bucket 迁移到另一个 bucket。 • <code>qiniu</code>，从七牛云存储迁移到OSS。 • <code>bos</code>，从百度的云存储迁移到OSS。

字段	含义	说明
		<ul style="list-style-type: none"> • ks3，从金山云存储迁移到OSS。 • s3，从 AWS S3 迁移到OSS。 • youpai，从又拍云迁移到OSS。 • http，通过提供的HTTP链接列表迁移数据到OSS。 • cos，从腾讯云存储COS迁移到OSS。 • azure，从Azuer Blob迁移到OSS。
srcAccessKey	源AccessKey，字符串。	如果srcType设置为oss、qiniu、baidu、ks3、s3，填写数据源的AccessKey。 <ul style="list-style-type: none"> • local、http，该项不需要填写。 • youpai、azure则填写用户名AccountName。
srcSecretKey	源SecretKey，字符串。	如果 srcType 设置为oss、qiniu、baidu、ks3、s3，填写数据源的 SecretKey。 <ul style="list-style-type: none"> • local、http，该项不需要填写。 • youpai，填写操作员密码。 • azure，填写AccountKey。
srcDomain	源Endpoint	如果 srcType 设置为local、http，该配置项不需要填。 <ul style="list-style-type: none"> • oss，从控制台获取的域名，非带bucket前缀的二级域名，列表请参看域名列表。 • qiniu，从七牛控制台获取的对应bucket的域名。

字段	含义	说明
		<ul style="list-style-type: none"> • bos，百度BOS域名，如 <code>http://bj.bcebos.com</code> 或 <code>http://gz.bcebos.com</code>。 • ks3，金山ks3域名，如 <code>http://kss.ksyun.com</code> 或 <code>http://ks3-cn-beijing.ksyun.com</code> 或 <code>http://ks3-us-west-1.ksyun.com</code>。 • S3，AWS S3各 region 的域名请参考S3 Endpoint。 • youpai，又拍云域名，如自动判断最优线路<code>http://v0.api.upyun.com</code> 或电信线路<code>http://v1.api.upyun.com</code> 或联通网通线路<code>http://v2.api.upyun.com</code>或移动铁通线路<code>http://v3.api.upyun.com</code>。 • cos，腾讯云填写bucket所在的区域，比如华南园区：<code>gz</code>、华北园区：<code>tj</code>、华东园区：<code>sh</code>。 • azure，Azure Blob连接字符串中的 EndpointSuffix，如<code>core.chinacloudapi.cn</code>。
srcBucket	源bucket名字或container名称。	如果 srcType 设置为 <code>local</code> 、 <code>http</code> ，不需要填写。 azure ，Azure Blob填写container名称，其它填写bucket名称。
srcPrefix	源前缀，字符串，默认为空。	如果 srcType = <code>local</code> ，填写本地目录，需要完整路径，以/进行分割并且以/结尾，如 <code>c:/example/</code> 或 <code>/data/example/</code> 。

字段	含义	说明
		srcType 为oss、qiniu、bos、ks3、youpai、s3，则为待同步object的前缀，不包括bucket名称，如data/to/oss/，同步所有文件srcPrefix设置为空。
destAccessKey	目的AccessKey，字符串。	OSS的AccessKeyID，请到 阿里云控制台 查看。
destSecretKey	目的SecretKey，字符串。	OSS的AccessKeySecret，请到 阿里云控制台 查看。
destDomain	目的endpoint，字符串。	从 阿里云控制台 获取，非带bucket前缀的二级域名，列表请参看域名列表。
destBucket	目的bucket，字符串。	OSS的bucket名称，不需要以/结尾。
destPrefix	目标前缀，字符串，默认为空。	<ul style="list-style-type: none"> 目标前缀，默认为空，直接放在目标bucket下。 如果要将数据同步到oss的某个目录下，请以/结尾，如data/in/oss/。 注意oss不支持以/作为文件的开头，所以destPrefix请不要配置以/做为开头。 一个本地文件路径为srcPrefix+relativePath的文件，迁移到oss的路径为destDomain/destBucket/destPrefix +relativePath。 一个云端文件路径为srcDomain/srcBucket/srcPrefix+relativePath的文件，迁移到oss的路径为

字段	含义	说明
		destDomain/destBucket/ destPrefix+relativePath。
taskObjectCountLimit	每个Task最大的文件数，整型，默认10000。	该配置项会影响到任务执行的并行度，一般配置为总文件数/Worker总数/迁移线程数(workerTaskThreadNum)，最大值不要超过50000，如果不知道总文件数，请使用默认值。
taskObjectSizeLimit	每个Task最大数据量，整型，单位bytes，默认1GB。	该配置项会影响到任务执行的并行度，一般配置为总数据量/Worker总数/迁移线程数(workerTaskThreadNum)，如果不知道总数据量，请使用默认值。
isSkipExistFile	数据迁移时是否跳过已经存在的文件，布尔类型。	当设置为true时，根据文件的size和LastModifiedTime判断是否跳过；为false时，总是覆盖OSS上已有文件。默认为值为false。jobType为audit时此项不生效。
scanThreadCount	并行扫描文件的线程数，整型，默认1。	该配置项与扫描文件的效率有关，没有特殊需求请不要修改。
maxMultiThreadScanDepth	最大允许并行扫描目录的深度，整型，默认1。	<ul style="list-style-type: none"> 默认值1表示在顶级目录间并行扫描。 没有特殊需求不要修改，随意配置过大值会导致任务无法正常运行。
appld	腾讯云COS的 appld ，整型。	srcType=cos时有效。
httpListFilePath	HTTP列表文件的绝对路径，字符串。	<ul style="list-style-type: none"> srcType=http时有效，源为HTTP链接地址时，需要提供内容为HTTP链接地址文件的绝对路径，如c:/example/http.list。

字段	含义	说明
		<ul style="list-style-type: none"> 该文件中的HTTP链接需要划分成两列，以空格分开，分别代表前缀和上传到OSS后的相对路径。例如c:/example/http.list文件内容如：<code>http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/aa/ bb.jpg</code> <code>http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/cc/dd.jpg</code>两行，迁移到OSS的文件名分别是 <code>destPrefix + bb.jpg</code>和 <code>destPrefix + cc/dd.jpg</code>。

- workers

workers是分布式模式先独有的，每个IP一行。如：

```
192.168.1.6
192.168.1.7
192.168.1.8
```

提示：

- 上述配置情况下，第一行的 192.168.1.6 一定是 master ；即 192.168.1.6 上会启动 Master 、 Worker 、 TaskTracker ， Console 也需要在该机上执行。
- 多个 Worker 机器的用户名、登录方式、工作目录请确保相同。

配置文件示例

下表中是分布式部署下的数据迁移任务配置文件，单机的配置文件名是 `local_job.cfg`，配置项与分布式部署时没有区别。

迁移类型	配置文件	说明
从本地迁移到OSS	job.cfg	srcPrefix 是以 / 结尾的绝对路径，如 <code>D:/work/oss/data /</code> ， <code>/home/user/work/oss/data/</code>

迁移类型	配置文件	说明
从七牛云存储迁移到OSS	job.cfg	srcPrefix 和 destPrefix 可以配置为空；如果不为空，请以 / 结尾，如 destPrefix=docs /
从百度bos迁移到OSS	job.cfg	srcPrefix 和 destPrefix 可以配置为空；如果不为空，请以 / 结尾，如 destPrefix=docs /
从AWS S3迁移到OSS	job.cfg	S3的 域名列表
从又拍云存储迁移到OSS	job.cfg	srcAccessKey/srcSecretKey填操作员账号及密码
从腾讯cos迁移到OSS	job.cfg	srcDomain请按照V4版本填写，如srcDomain=sh；srcPrefix可以为空，当不为空时候，请以 / 开头和结尾，如 srcPrefix=/docs/
从Azure blob迁移到OSS	job.cfg	srcAccessKey/srcSecretKey填存储账号及密钥；srcDomain填连接字符串中的 EndpointSuffix，如 core.chinacloudapi.cn
从OSS迁移到OSS	job.cfg	适用于不同区域之间、不同存储类型之间、不同前缀之间的数据迁移；推荐在ECS上部署，并使用带internal的域名，可以节省流量费用
使用高速通道迁移数据到OSS	job.cfg	适用于所有数据源，如果您有高速迁移需求，请提交工单或联系OSS支持人员

1.3 单机部署

下载

单机部署支持Linux、Windows。

单机版本下载地址 [ossimport-2.3.2.zip](#)，下载到本地后，使用工具或命令unzip，解压后的文件结构如下：

```
ossimport
├── bin
│   └── ossimport2.jar # 包括Master、Worker、TaskTracker、Console四个模块的总jar
├── conf
│   ├── local_job.cfg # Job配置文件
│   └── sys.properties # 系统运行参数配置文件
├── console.bat # Windows命令行，可以分布执行调入任务
├── console.sh # Linux命令行，可以分布执行调入任务
├── import.bat # Windows一键导入，执行配置文件为conf/local_job.cfg配置的数据迁移任务，包括启动、迁移、校验、重试
├── import.sh # Linux一键导入，执行配置文件为conf/local_job.cfg配置的数据迁移任务，包括启动、迁移、校验、重试
├── logs # 日志目录
└── README.md # 说明文档，强烈建议使用前仔细阅读
```

配置

单机版本有两个配置文件conf/sys.properties、conf/local_job.cfg。注意不要修改以下内容：

- conf/sys.properties中的配置项 workingDir、workerUserName、workerPassword、privateKeyFile
- conf/local_job.cfg的名称和位置，配置项 jobName

其它配置项请按照实际需求配置。



说明：

请在提交任务前确认 sys.properties 和 local_job.cfg 中的参数，任务提交后参数无法再修改。

运行

单机模式下，数据迁移任务有以下两种执行方式：

- 一键导入：是对所有步骤的封装，按照脚本提示执行即可完成数据迁移。
- 分步执行：执行启动服务、提交任务、重试失败子任务等步骤。



说明：

对于初级用户强烈建议使用一键导入。

- 一键导入

1. 执行一键导入，Window系统下在 `cmd.exe` 中执行 `import.bat`，Linux终端中执行 `bash import.sh`。
2. 如果之前执行过程序，会提示有是否从上次断点处继续执行，或者重新执行同步任务。对新的数据迁移任务，或者修改了同步的源端/目的端，请选择重新执行。
3. Windows下任务开始后，会打开一个新的cmd窗口执行同步任务并显示日志，旧窗口会每隔10秒打一次任务状态，数据迁移期间不要关闭两个窗口；Linux下服务在后台执行。
4. 当 Job 完成后，如果发现有任务失败了，会提示是否重试，输入 `y` 重试，输入 `n` 则跳过退出。
5. 如果上传失败，请打开文件 `master/jobs/local_test/failed_tasks/<tasktaskid>/audit.log` 查看，确定失败原因。

- 分步执行



说明：

没有特殊需要，请使用 一键导入方式迁移数据。

1. 清除同名任务。

如果以前运行过同名任务，需要重新执行任务，请先清除同名任务。如果没有运行过，或需要重试失败任务，不要执行清除命令。Window下在 `cmd.exe` 中执行 `console.bat clean`，Linux下在终端执行 `bash console.sh clean`。

2. 提交数据迁移任务。

OssImport不能提交同名任务，如果有请先清除。提交任务的配置文件为 `conf/local_job.cfg`，默认任务名称为 `local_test`。提交任务的命令，Window下在 `cmd.exe` 中执行 `console.bat submit`，Linux下在终端执行 `bash console.sh submit`。

3. 启动服务。

Windows下在 `cmd.exe` 中执行 `console.bat start`，Linux下在终端执行 `bash console.sh start`。

4. 查看任务状态。

Windows下在 `cmd.exe` 中执行 `console.bat stat`，Linux下在终端执行 `bash console.sh stat`。

5. 失败Task重试。

由于网络或其它原因，子任务可能失败。失败重试只重试失败的Task，不会重试成功的Task。Windows下在 `cmd.exe` 中执行 `console.bat retry`，Linux下在终端执行 `bash console.sh retry`。

6. 停止服务。

Windows下在关闭窗口 `%JAVA_HOME%/bin/java.exe` 即可，Linux下在终端执行 `bash console.sh stop`。

• 常见失败原因

- 上传过程中源目录的文件发生了修改，`log/audit.log`里会提示`SIZE_NOT_MATCH`相关字样的错误，这种情况下老的文件已经上传成功，新的修改没有上传到OSS；
- 源文件在上传过程中被删除，导致下载的时候失败；
- 源文件名不符合OSS命名规范（不能以/开头，不能为空），导致上传到OSS失败；
- 下载数据源文件失败；
- 程序异常退出，任务状态为 `Abort`，这种情况请联系我们。

• 任务状态及日志

任务提交后，Master分解成Task，有Worker执行Task，Tracker收集Task状态。任务运行完成后 `ossimport` 目录内容如下：

```
ossimport
├── bin
│   ├── ossimport2.jar    # 单机版jar
├── conf
│   ├── local_job.cfg     # Job配置文件
│   └── sys.properties    # 系统运行参数配置文件
├── console.sh            # 命令行工具
├── import.sh             # 一键导入脚本
├── logs
│   ├── import.log        # 归档日志
│   ├── job_stat.log      # 任务状态记录
│   ├── ossimport2.log    # 单机版运行日志
│   └── submit.log        # 任务提交记录
├── master
│   ├── jobqueue          # 存放尚未分解完成的任务
│   └── jobs              # 存放任务运行状态
│       ├── local_test    # 任务名称
│       └── checkpoints   # Master分解Job到Task的checkpoint点
├── 记录
│   ├── 0
│   │   └── 034DC9DD2860B0CFE884242BC6FF92E7.cpt
│   ├── dispatched       # 已经分配给Worker尚未运行完成的Task
│   └── localhost
```

```

|           |—— failed_tasks      # 运行失败的Task
|           |—— pending_tasks    # 尚未分配的Task
|           |—— succeed_tasks   # 成功运行的Task
|           |—— A41506C07BF1DF12A3EDB4CE31756B93F_1499744
514501@localhost
|           |—— audit.log       # Task运行日志，通过该日志可以查看
错误原因
|           |—— DONE           # Task运行成功标志
|           |—— error.list     # Task错误列表，可以查看错误文件列
表
|           |—— STATUS         # 任务状态标志文件，内容为Failed或
Completed，表示子任务失败或成功
|           |—— TASK           # Task描述信息
|—— worker      # Worker正在运行的Task状态，运行完成后有Master管理
|—— jobs
|—— local_test
|—— tasks

```

提示：

- Job运行信息，可以查看`log/ossimport2.log`；
- Task的失败原因，可以查看`master/jobs/${JobName}/failed_tasks/${TaskName}/audit.log`；
- Task的失败文件，可以查看`master/jobs/${JobName}/failed_tasks/${TaskName}/error.list`。

常见错误及排除

请查看 [常见错误及排除](#)。

1.4 分布式部署

下载

分布式部署目前只支持Linux，Windows暂不支持。

分布式版本下载地址 [ossimirror-2.3.2.tar.gz](https://ossimirror.ossimg.com/ossimirror-2.3.2.tar.gz)，下载到本地后，使用命令 `tar -zxvf ossimirror-2.3.2.tar.gz -C $HOME/ossimirror` 解压，解压后的文件结构如下：

```
ossimport
├── bin
│   ├── console.jar      # Console模块jar包
│   ├── master.jar       # Master模块jar包
│   ├── tracker.jar      # Tracker模块jar包
│   └── worker.jar       # Worker模块jar包
├── conf
│   └── job.cfg          # Job配置文件模板
```

```
|   |—— sys.properties # 系统运行参数配置文件  
|   |—— workers       # Worker列表  
|—— console.sh        # 命令行工具，目前支持只Linux  
|—— logs              # 日志目录  
|—— README.md         # 说明文档，使用前请仔细阅读
```

注意：

- `OSS_IMPORT_HOME`：OssImport的根目录，默认为解压命令中的目录 `$HOME/ossimport`，也可以通过命令 `export OSS_IMPORT_HOME=<dir>` 或修改系统配置文件 `$HOME/.bashrc` 设置，推荐使用默认目录；
- `OSS_IMPORT_WORK_DIR`：OssImport的工作目录，通过 `conf/sys.properties` 的配置项 `workingDir` 指定，推荐为 `$HOME/ossimport/workdir`；
- `OSS_IMPORT_HOME` 或 `OSS_IMPORT_WORK_DIR` 请使用绝对路径配置，如 `/home/<user>/ossimport` 或 `/home/<user>/ossimport/workdir`。

配置

分布式部署有三个配置文件`conf/sys.properties`、`conf/job.cfg`、`conf/workers`。

- `conf/job.cfg`：分布式模式下任务的配置文件模板，数据迁移前请按照实际参数修改；
- `conf/sys.properties`：系统运行参数配置文件，如工作目录、Worker运行参数等请在该文件中配置；
- `conf/workers`：worker列表。

注意：

- 请在提交任务前确认 `sys.properties` 和 `job.cfg` 中的参数，任务提交后参数无法再修改；
- Worker 列表 `workers` 请启动服务前确定，启动后无法再增加或删除。

运行

- 执行迁移任务

分布式部署时，执行任务的一般步骤是 修改任务配置文件、部署服务、清除同名任务、提交任务、启动迁移服务、查看任务状态、重试失败子任务以及停止迁移任务。详细说明如下：

— 部署服务。执行Linux终端执行 `bash console.sh deploy`。



说明：

部署前请保证配置文件 `conf/job.cfg`、`conf/workers` 已经修改完成。

- 清除同名任务。如果运行过同名任务，需要从新执行任务，请先清除同名任务。如果没有运行过，或需要重试失败任务，不需要执行清除命令。Linux终端中执行 `bash console.sh clean job_name`。
- 提交数据迁移任务。OssImport不能重复提交同名任务，如果有请使用clean命令清除。提交任务需要指定任务的配置文件，任务的配置文件模板在 `conf/job.cfg`，建议在模板的基础上修改。Linux终端执行 `bash console.sh submit [job_cfg_file]`，提交配置文件为 `job_cfg_file` 的任务，`job_cfg_file` 为可选参数，不指定是默认为 `$OSS_IMPORT_HOME/conf/job.cfg`，`$OSS_IMPORT_HOME` 默认为 `console.sh` 所在的目录。
- 启动服务。Linux终端执行 `bash console.sh start`。
- 查看任务状态。Linux终端执行 `bash console.sh stat`。
- 失败Task重试。由于网络或其它原因，Task可能运行失败。失败重试只重试失败的Task，成功的Task不会重试。Linux下在终端执行 `bash console.sh retry [job_name]`，`job_name` 为可选参数，指定定时重试任务 `job_name` 的失败子任务，不指定 `job_name` 时重试所有任务的失败子任务。
- 停止服务。Linux终端执行 `bash console.sh stop`。

提示：

- `bash console.sh` 在参数错误时，会自动提示命令格式；
- 配置文件和提交任务中的目录，推荐使用绝对目录；
- 任务的配置，即 `job.cfg` 中的配置项。



说明：

配置项提交后不能修改，请在提交前确定。

- 常见任务失败原因
 - 上传过程中源目录的文件发生了修改，`log/audit.log`里会提示 `SIZE_NOT_MATCH` 相关字样的错误，这种情况下老的文件已经上传成功，新的修改没有上传到OSS；
 - 源文件在上传过程中被删除，导致下载的时候失败；
 - 源文件名不符合OSS命名规范（不能以/开头，不能为空），导致上传到OSS失败；
 - 下载数据源文件失败；
 - 程序异常退出，任务状态为 `Abort`，这种情况请联系我们。
- 任务状态及日志

任务提交后，Master分解成Task，有Worker执行Task，Tracker收集Task状态。任务运行完成后workdir目录内容如下：

```
workdir
├── bin
│   ├── console.jar    # Console模块jar包
│   ├── master.jar     # Master模块jar包
│   ├── tracker.jar    # Tracker模块jar包
│   └── worker.jar     # Worker模块jar包
├── conf
│   ├── job.cfg        # Job配置文件模板
│   ├── sys.properties # 系统运行参数配置文件
│   └── workers         # Worker列表
├── logs
│   ├── import.log     # 归档日志
│   ├── master.log     # Master日志
│   ├── tracker.log    # Tracker日志
│   └── worker.log     # Worker日志
├── master
│   ├── jobqueue       # 存放尚未分解完成的任务
│   └── jobs            # 存放任务运行状态
│       ├── xctooss    # 任务名称
│       └── checkpoints # Master分解Job到Task的checkpoint
├── 点记录
│   ├── 0
│   │   └── ED09636A6EA24A292460866AFDD7A89A.cpt
│   ├── dispatched    # 已经分配给Worker尚未运行完成的Task
│   │   └── 192.168.1.6
│   ├── failed_tasks  # 运行失败的Task
│   │   └── A41506C07BF1DF2A3EDB4CE31756B93F_1499348
│   └── 973217@192.168.1.6
│       ├── audit.log  # Task运行日志，通过该日志可以
│       │   └── 查看错误原因
│       ├── DONE      # Task运行成功标志，失败为空
│       └── error.list # Task错误列表，可以查看错误文
│           └── 件列表
│               ├── STATUS # 任务状态标志文件，内容为
│               │   └── Failed或Completed，表示子任务失败或成功
│               └── TASK   # Task描述信息
│                   ├── pending_tasks # 尚未分配的Task
│                   └── succeed_tasks # 成功运行的Task
│                       └── A41506C07BF1DF2A3EDB4CE31756B93F_1499668
│                           └── 462358@192.168.1.6
│                               ├── audit.log # Task运行日志，通过该日志可以查
│                               │   └── 看错误原因
│                               ├── DONE      # Task运行成功标志
│                               └── error.list # Task错误列表，成功为空
│                                   ├── STATUS # 任务状态标志文件，内容为Failed
│                                   │   └── 或Completed，表示子任务失败或成功
│                                   └── TASK   # Task描述信息
```

```

└── worker # Worker正在运行的Task状态，运行完成后由Master管理
    └── jobs
        ├── local_test2
        │   └── tasks
        ├── local_test_4
        │   └── tasks
    
```

提示：

- Job运行状态查看 `logs/tracker.log`，Worker的运行日志查看 `logs/worker.log`，Master的运行日志查看 `logs/master.log`。
- Task的失败原因，可以查看 `master/jobs/${JobName}/failed_tasks/${TaskName}/audit.log`。
- Task的失败文件，可以查看 `master/jobs/${JobName}/failed_tasks/${TaskName}/error.list`。

常见错误及排除

请查看 [常见错误及排除](#)。

1.5 数据迁移

本文主要介绍OssImport在典型场景下数据迁移需求的实现。

OssImport介绍

部署方式

OssImport有单机模式和分布式模式两种部署方式。

- 对于小于30TB的小规模数据迁移，单机模式即可完成。
- 对于大规模的数据迁移，请使用分布式模式。

分时限流

OssImport是基于 master-worker 的分布式架构。Worker有限流功能，通过修改配置文件 `sys.properties` 的配置项 `workerMaxThroughput (KB/s)` 实现，修改后需要重启服务才能生效。

分布式部署情况下，需要修改每个Worker的 `$OSS_IMPORT_WORK_DIR/conf` 下的 `sys.properties`，然后重启服务。

要实现分时限流，可通过 `crontab` 定时修改 `sys.properties`，然后重启服务生效。

添加Worker

Worker列表请在提交任务前确定，目前不支持动态添加。

只校验不迁移数据

OssImport支持只校验数据不迁移数据，设置任务配置文件 `job.cfg` 或 `local_job.cfg` 的配置项 `jobType`为`audit` 而不是 `import`，其它配置与数据迁相同。

数据迁移增量模式

数据迁移增量模式，是指数据迁移任务启动后，先进行一次全量迁移，每隔一段时间自动的进行增量数据迁移。第一次数据迁移任务为全量迁移，提交任务后立即启动；后面的增量数据迁移每隔一个周期启动一次。数据迁移增量模式适用于数据备份和数据同步。

增量模式有两个配置项：

- `job.cfg` 中的 `isIncremental`，表示是否打开增量迁移模式，`true`表示打开增量模式，`false`表示关闭增量模式，默认关闭。
- `job.cfg` 中的 `incrementalModeInterval`，表示增量模式下的同步间隔，即增量数据迁移的间隔周期，单位秒。`isIncremental=true` 时有效。可配置的最小值为900秒，不建议配置成小于3600秒的值，会浪费大量请求，造成额外的系统开销。

指定迁移文件的过滤条件

迁移文件的过滤条件，即只迁移满足特定条件的文件。OssImport支持指定前缀和最后修改时间：

- `job.cfg` 中的 `srcPrefix`，用来指定迁移文件的前缀，默认为空。
 - 如果`srcType=local`，填写本地目录，需要完整路径，以`/`进行分割并且以`/`结尾，如 `c:/example/` 或 `/data/example/`。
 - 如果`srcType` 为`oss`、`qiniu`、`bos`、`ks3`、`youpai`、`s3`，则为待同步object的前缀，不包括bucket名称，如`data/to/oss/`。迁移所有文件时`srcPrefix`设置为空。
- `job.cfg` 中的 `importSince`，用来指定迁移文件的最后修改时间，整形，单位秒。`importSince` 为Unix时间戳，即自1970年1月1日UTC零点以来的秒数，通过命令`date +%s`获取；默认值0，表示迁移全部数据。增量模式下只对第一次全量迁移有效，非增量模式对整个迁移任务有效。
 - 如果文件的最后修改(`LastModified Time`)在 `importSince` 之前（包含）将被迁移。
 - 如果文件的最后修改(`LastModified Time`)在 `importSince` 之后将不被迁移。

典型场景

场景1：从第三方存储服务无缝切换到OSS

以下步骤可以完成从第三方存储到OSS的无缝切换：

1. 全量迁移数据，此时业务仍在第三方存储上，记下数据迁移的开始时间 T1，注意该时间为 Unix 时间戳，即自 1970 年 1 月 1 日 UTC 零点以来的秒数，通过命令 `date +%s` 获取。
2. 打开 OSS 镜像回源功能，数据迁移完成后，请在 OSS 控制台 设置服务 Bucket 的 [镜像回源](#) 功能，回源地址为第三方存储。
3. 读写切换到 OSS，此时 T1 前的数据从 OSS 读取，T1 后的数据利用镜像回源从第三方服务读取，新数据完全写入 OSS。
4. 增量数据迁移，增量数据迁移任务的配置文件(`job.cfg` 或 `local_job.cfg`)的配置项 `importSince=T1`，增量数据迁移完成的时间为 T2。



说明：

增量数据迁移，并非数据迁移的增量模式。

5. 删除第三方存储，T2 后，您业务的所有的读写都在 OSS 上，第三方存储只是一份历史数据，您可以根据需要决定保留或删除。OssImport 负责数据的迁移和校验，不会删除任何数据。

场景2：本地数据迁移到OSS

本地数据迁移到OSS的工具选择：

- 对于小于 30TB 的数据从本地文件或可以挂载到本地文件系统的迁移情况，推荐使用 [OssUtil](#)，该工具简单方便，OssUtil 支持文件级别的增量上传，通过 `-u/--update` 和 `--snapshot-path` 选项实现，详细说明请使用 `ossutil help cp` 查看。
- 大规模数据迁移，请使用分布式版本的 [OssImport](#)。



说明：

本地数据的增量迁移时，文件系统某些操作不会修改文件的最后修改时间，比如 Windows 的 `cp`、`mv`，Linux 的 `mv`、`rsync` 带 `-t` 或 `-a` 选项，这些操作的数据修改都不会被检测到，也不会同步到 OSS。

场景3：OSS之间的数据迁移

- 什么时候使用 OssImport：
 - 不同区域间的 OSS 数据同步，推荐使用 [跨区域复制](#) 功能，该功能请在控制台上设置。
 - 由于安全原因，没有开通 [跨区域复制](#) 的地域，可以使用 OssImport 迁移或备份数据。
 - 同一区域内，不同账号、不同 Bucket 的数据迁移。
 - OSS 直接的数据迁移，推荐使用阿里云内网，即使用 ECS、OSS 的域名带 `internal`。

- OSS直接数据迁移收费：
 - 如果使用了带internal的域名，不会产生流量费用，只有请求和存储费用。
 - 如果没有带internal的域名，会产生流量费用，具体请参看[计费](#)。
- 不推荐使用场景：
 - 开通了跨区域复制服务的地域之间的数据同步。
 - 利用增量模式在OSS之间同步文件修改操作，OssImport只能同步文件的修改操作(put/append/multipart)，不能同步读取和删除操作，数据同步的及时性没有具体的 SLA 保证，请慎重选择。推荐使用[上传回调](#)。

迁移说明

ECS与流量

对于从云端（非本地）迁移到OSS，且带宽资源不是很充足的用户，建议购买按量付费的ECS进行迁移，ECS配置如下：

- 付费方式选择按量付费。
- 地域选择OSS对应的地域。
- 带宽峰值选100M。

在配置迁移服务时，将targetDomain 设为带internal 的内网域名；如果源端也是OSS，将srcDomain 也设为带 internal的内网域名，可以省掉从OSS源端下载的流量费，仅收取OSS访问次数的费用。

HTTP数据迁移到OSS

HTTP数据迁移任务需要配置的参数：

- job.cfg 中的 srcType 配置为 srcType=http ，请注意字符大小写。
- job.cfg 中的 httpListFilePath，指定的HTTP地址列表文件， 请使用绝对路径指定，如 c:/example/http.list、/root/example/http.list 。一个完整的HTTP链接是 127.0.0.1/aa/bb.jpg，不同的切分方法最后会导致上传到oss的路径会不一样：

```
http://127.0.0.1/aa/    bb.jpg      # 第一行
http://127.0.0.1/      aa/bb.jpg  # 第二行
```

第一行的文件导入到OSS后的的文件名为 destPrefix + bb.jpg，第二行的文件名为 destPrefix + aa/bb.jpg。httpPrefixColumn 指定域名列，默认第一列，如上述的 127.0

.0.1/aa/ 或127.0.0.1/。 `relativePathColumn` 指定在OSS中文件名，如上述的 `bb.jpg` 或 `aa/bb.jpg`。如果文件中有多列，如下：

```
http://127.0.0.1/aa/    bb/cc dd/ee    ff.jpg
```

配置应该如下：`httpPrefixColumn=1`，`relativePathColumn=4`。

- `job.cfg` 中的 `destAccessKey`、`destSecretKey`、`destDomain`、`destBucket` 等OSS的配置。

HTTP数据迁移子任务切分参数：

- `taskObjectCountLimit`，每个 Task 最大的文件数，默认10000；
- `taskObjectSizeLimit`，每个 Task 最大数据量，HTTP数据迁移时该参数无效，原因是 Master 切分 Task 时，如果每个HTTP文件都是源上获取文件大小，每个文件都有一次HTTP请求开销，会影响子任务分配的效率，进而影响子任务的并发执行，降低迁移的效率。
- 域名，`httpListFilePath` 指定的文件中第一列，连续相同的域名任务按照 `taskObjectCountLimit` 的限制切分，连续不同的域名切分成不同的 Task，这种做法的目的是为了更好的复用连接。比如：

```
http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/  import/test1.txt
http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/  import/test2.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/   import/test3.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/   import/test4.txt
```

`taskObjectCountLimit` 大于2的情况下，会切分成 2 个 Task，而以下情况会切分成 4 个 Task：

```
http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/  import/test1.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/   import/test3.txt
http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/  import/test2.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/   import/test4.txt
```

所以 `httpListFilePath` 指定的HTTP地址列表文件，请先按照域名排序。

网络流量与参数配置

以下参数的配置与网络流量有关：

- `sys.properties` 中的 `workerTaskThreadNum`，表示 Worker 并发执行的任务数量，如果网络较差、并发大，会出现大量超时错误，此时应该降低并发量，修改该配置项，并重启服务。

- `sys.properties` 中的 `workerMaxThroughput(KB/s)`，表示 Worker 流量的上限，如果业务需要限流，比如源端流控控制、网络限制等情况。该参数的值应该小于机器的最大网络流量，并根据业务需要评估。
- `job.cfg` 中的 `taskObjectCountLimit`，每个 Task 最大的文件数，默认10000。该参数会影响 Task 的数量，数量过小无法实现有效的并发。
- `job.cfg` 中的 `taskObjectSizeLimit`，每个 Task 最大数据量，默认1GB。该参数会影响 Task 的数量，数量过小无法实现有效的并发。



说明：

- 配置文件参数请尽量在启动迁移前确定。
- `sys.properties` 中的参数修改后，重启迁移服务器后才能生效。
- `job.cfg` 任务提交后，任务的配置参数无法更改。

2 ossbrowser

ossbrowser是OSS官方提供的图形化的管理工具，提供类似Windows资源管理器的功能。您可以方便地浏览文件、上传/下载文件，并支持断点续传。

ossbrowser主要功能包括：支持AK登录、临时授权码登录，管理Bucket，管理文件，提供Policy授权，生成STS临时授权。

下载安装

支持平台	下载地址
Window x32	Window x32
Window x64	Window x64
MAC	MAC
Linux x64	Linux x64

登录ossbrowser

- 使用AK登录ossbrowser

您可以使用AK（比如子账号AK）登录使用ossbrowser。



说明：

不推荐使用主账号AK登录。

1. 登录[RAM控制台](#)创建子账号。

子帐号的权限分为：

- 大权限子账号（即拥有所有Bucket权限，且可以管理RAM配置的子账号）。初级用户推荐如下配置：

编辑个人授权策略

添加授权策略后，该账户即具有该条策略的权限，同一条授权策略不能被重复添加。

可选授权策略名称	类型
请输入关键词查询	
AliyunOSSReadOnlyAccess 只读访问对象存储服务(OSS)的权限	系统
AliyunECSEFullAccess 管理云服务器服务(ECS)的权限	系统
AliyunECSReadOnlyAccess 只读访问云服务器服务(ECS)的权限	系统
AliyunRDSFullAccess 管理云数据库服务(RDS)的权限	系统

已选授权策略名称

类型

AliyunOSSFullAccess
管理对象存储服务(OSS)权限

系统

AliyunRAMFullAccess
管理访问控制(RAM)的权限，即管理...

系统

AliyunSTSAssumeRoleAccess
调用STS服务AssumeRole接...

系统

确定

关闭



说明：

您可以为子帐号授予更小的权限，具体设置请参考[权限管理](#)。

- 小权限子账号（即只拥有部分Bucket或子目录的权限）。初级用户推荐使用[简化Policy授权](#)功能完成授权。

2. 使用子帐号登录ossbrowser。

AK登录

授权码登录

* Endpoint模板:

http://{region}.aliyuncs.com

https

* AccessKeyId:

LTAIEjnjMLLMfcjq

* AccessKeySecret:

.....

预设OSS路径(可选):

oss://test-oss-security0/pics/

* 区域

华东1(杭州)

备注:

杭州 onlyread

☒ 记住密钥

登入

AK历史



说明：

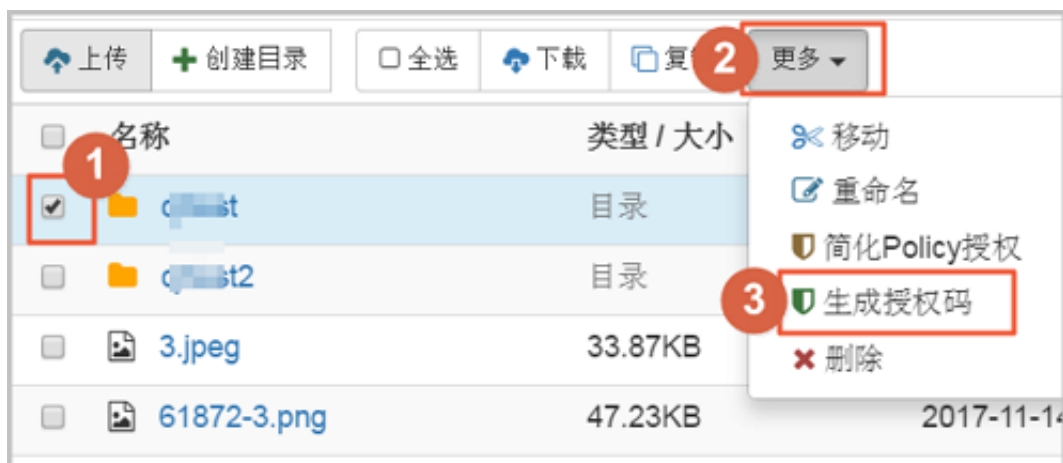
- 预设oss路径 (可选)：当前使用的AK只有某个Bucket或Bucket下某个路径的权限，需要设置预设OSS路径和区域。
- 记住密钥：勾选记住密钥可保存AK密钥。再次登录时，单击**AK历史**，可选择该密钥登录，不需要手动输入AK。请不要在临时使用的电脑上勾选。

- 使用临时授权码登录ossbrowser

OSSBrowser还支持临时授权码登录。适用场景：您可以将临时授权码提供给相应的人员，允许其在授权码到期前，临时访问您Bucket下某个目录。到期后，临时授权码会自动失效。

— 生成临时授权码

您作为管理员，先使用AK登录OSSBrowser，并在管理界面，选择需要临时授权的文件或目录，生成临时授权码。



— 登录时使用授权码

临时授权码可在过期前，可用来登录OSSBrowser。如下图所示：

AK登录

授权码登录

* 授权码: ?

0TFJBai9OZzNEM3ZITFlnMTIBRTZ2L3ZqdEVYT3FMKzVDQ3V3aG1

VbDg0FFa0Z

qMHkXBrc0pJ

VGllcMnU5dF

ZiOVcVJidW5o

VGxI745OUo4

bHRqjoiMjAxN

y0xMtaGFuZ3

pob3UiLCJvc3NwYXRoIjoib3NzOi8vZ3VhbmdjaHVuL2RibW8vliw

JpdmlsZWdlIjoicmVhZE9ubHkifQ==

(有效期至2017-10-27 19:17:02, 剩余时间: 58 分 28 秒)

登入

管理Bucket

登录ossbrowser后，您可以管理Bucket，包括：

- 新建Bucket
- 删除Bucket
- 修改Bucket权限
- 管理碎片

管理文件

ossbrowser提供的文件管理功能包括：

- 目录（包括Bucket）和文件的增加、删除、修改、搜索、复制、文件预览
- 文件传输任务管理：上传（支持拖拽操作）、下载、断点续传
- 地址栏功能：支持oss://协议URL、浏览历史前进后退、保存书签
- 归档型存储的管理：创建归档型存储、恢复归档型存储Bucket



说明：

归档存储型Bucket下所有文件均为Archive存储类型，需要恢复才能访问。

简化Policy授权

1. 勾选一个或多个需要授权的文件或目录，并单击简化Policy授权。



2. 在简化Policy授权窗口，选择权限。
3. 您可以查看生成的Policy文本，将其复制到您需要的地方使用，如RAM子账号、Role等Policy编辑等。



您也在该窗口中将权限授权给子账号（当前登录的AK必须有RAM的配置操作权限）。

简化Policy授权

将下列资源:

- 1. Koala.jpg

的权限:

☒ 只读权限 ☐ 读写权限

[\[查看Policy\]](#)

创建为policy, 命名为:

plc-readOnly-Koala-jpg

并授权给:

[新建一个]

新用户名: [随机生成](#)

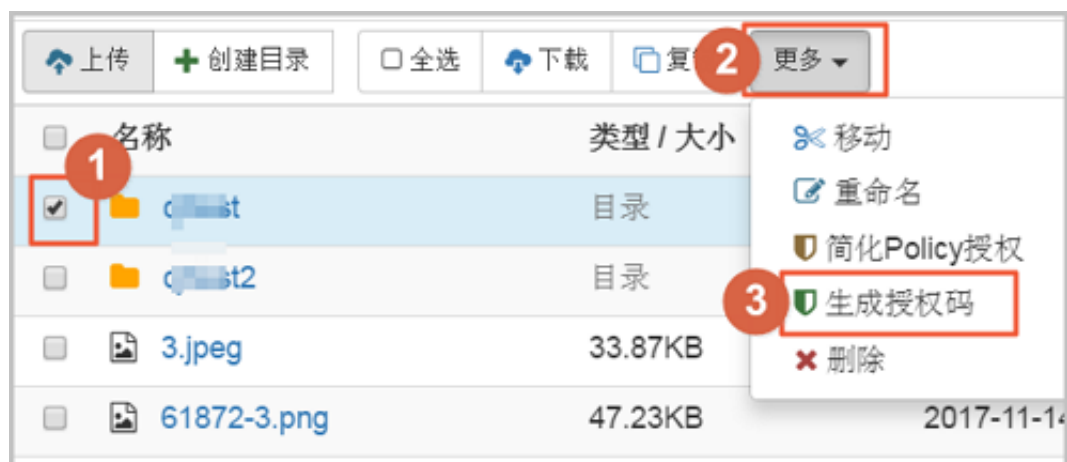
AK发送到邮件:

还没设置邮件发送配置, 需要先设置 [打开设置](#)

[✓ 确定授权](#) [✕ 关闭](#)

生成STS临时授权

1. 勾选文件夹，并选择生成授权码。



2. 在弹出的窗口中，设置参数并单击重新生成，将会生成授权码，如下图所示。

生成授权码

授权给目录:

• demo

的权限

☒ 只读权限

☐ 读写权限

[查看Policy]

有效时长(秒)

3600

还需要指定一个角色: (这个角色需要至少有这个目录的只读权限权限)

guang-oss-upload

生成的授权码:

eyJ...nliwic

点击复制

(使用上面生成的授权码登录OSS浏览器, 可以达到只拥有[demo]这个目录的只读权限权限的效果, 有效期至2017-10-27 19:17:02。)

✓ 重新生成

✕ 关闭

相关文档

- [创建AccessKey](#)
- [权限管理](#)
- [STS临时授权](#)
- [控制台-管理bucket](#)
- [Java SDK-管理bucket](#)
- [API-创建bucket](#)

32

文档版本：20180808

3 ossftp

3.1 如何快速安装OSS FTP

简介

OSS FTP工具是一个特殊FTP server, 它接收普通FTP请求后, 将对文件、文件夹的操作映射为对OSS的操作, 从而使得您可以基于FTP协议来管理存储在OSS上的文件。



说明：

生产环境请使用oss sdk, OSS FTP工具主要面向个人用户使用。

- 主要特性

- 跨平台: 无论是Windows、Linux还是Mac, 无论是32位还是64位操作系统, 无论是图形界面还是命令行都可以运行。
- 免安装: 解压后可直接运行。
- 免设置: 无需设置即可运行。
- 透明化: FTP工具是python写的, 您可以看到完整的源码, 我们稍后也会开源到Github。

- 主要功能

- 支持文件和文件夹的上传、下载、删除等操作。
- 通过Multipart方式, 分片上传大文件。
- 支持大部分FTP指令, 可以满足日常FTP的使用需求。



说明：

- 1. 目前在1.0版本中, 考虑到安装部署的简便, OSS FTP工具没有支持TLS加密。由于FTP协议是明文传输的, 为了防止您的密码泄漏, 建议将FTP server和client运行在同一台机器上, 通过127.0.0.1:port的方式来访问。
- 2. 不支持rename和move操作。
- 3. 安装包解压后的路径不要含有中文。
- 4. FTP server的管理控制页面在低版本的IE中可能打不开。
- 5. FTP server支持的Python版本: Python2.6, Python2.7。

下载

- Windows: [ossftp-1.0.3-win.zip](#)

由于Windows不会默认安装Python2.7，所以安装包中包含了Python2.7，免去您python安装配置的麻烦，解压即可使用。

- Linux/Mac: [ossftp-1.0.3-linux-mac.zip](#)

由于Linux/Mac系统默认会安装Python2.7或Python2.6，所以安装包中不再包含可执行的python，只包含了相关依赖库。

运行

首先解压之前下载的文件，然后根据环境情况选择不同的运行方式。

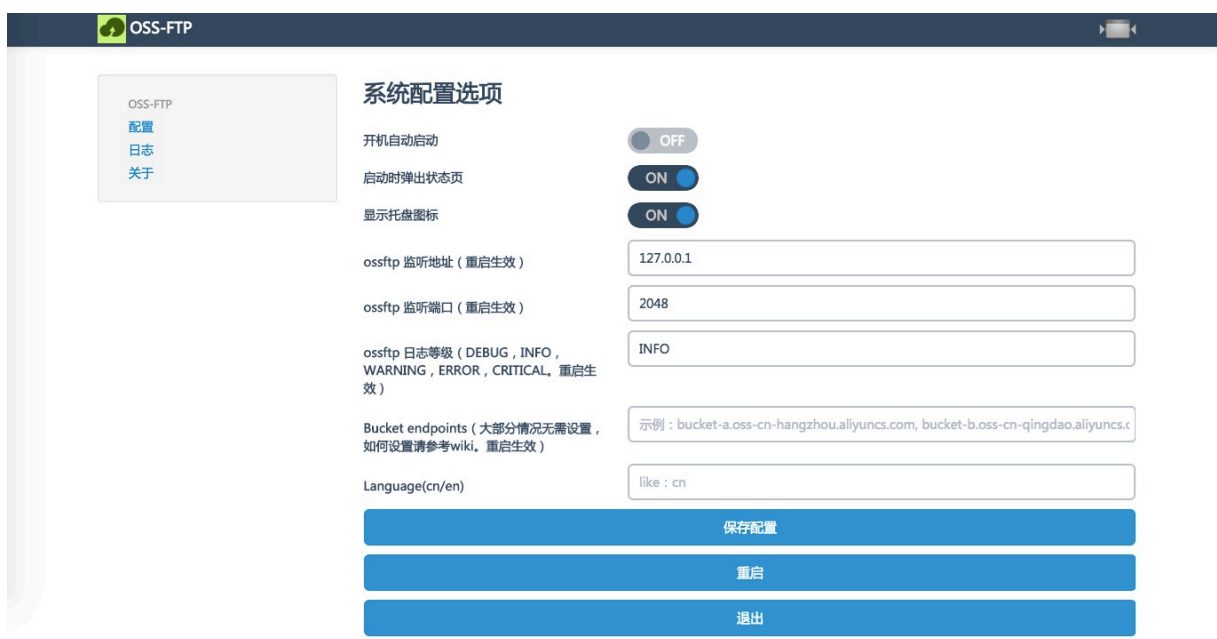
- Windows: 双击运行start.vbs即可
- Linux: 打开终端，运行

```
$ bash start.sh
```

- Mac: 双击start.command，或者在终端运行

```
$ bash start.command
```

上述步骤会启动一个FTP server，默认监听在127.0.0.1的2048端口。同时，为了方便您对FTP server的状态进行管控，还会启动一个web服务器，监听在127.0.0.1的8192端口。如果您的系统有图形界面，还会自动打开控制页面，如下所示：



**说明：**

大部分情况不要任何配置，就可以运行一个FTP server了，如果想对FTP server进行配置，请注意需要重启才能生效。

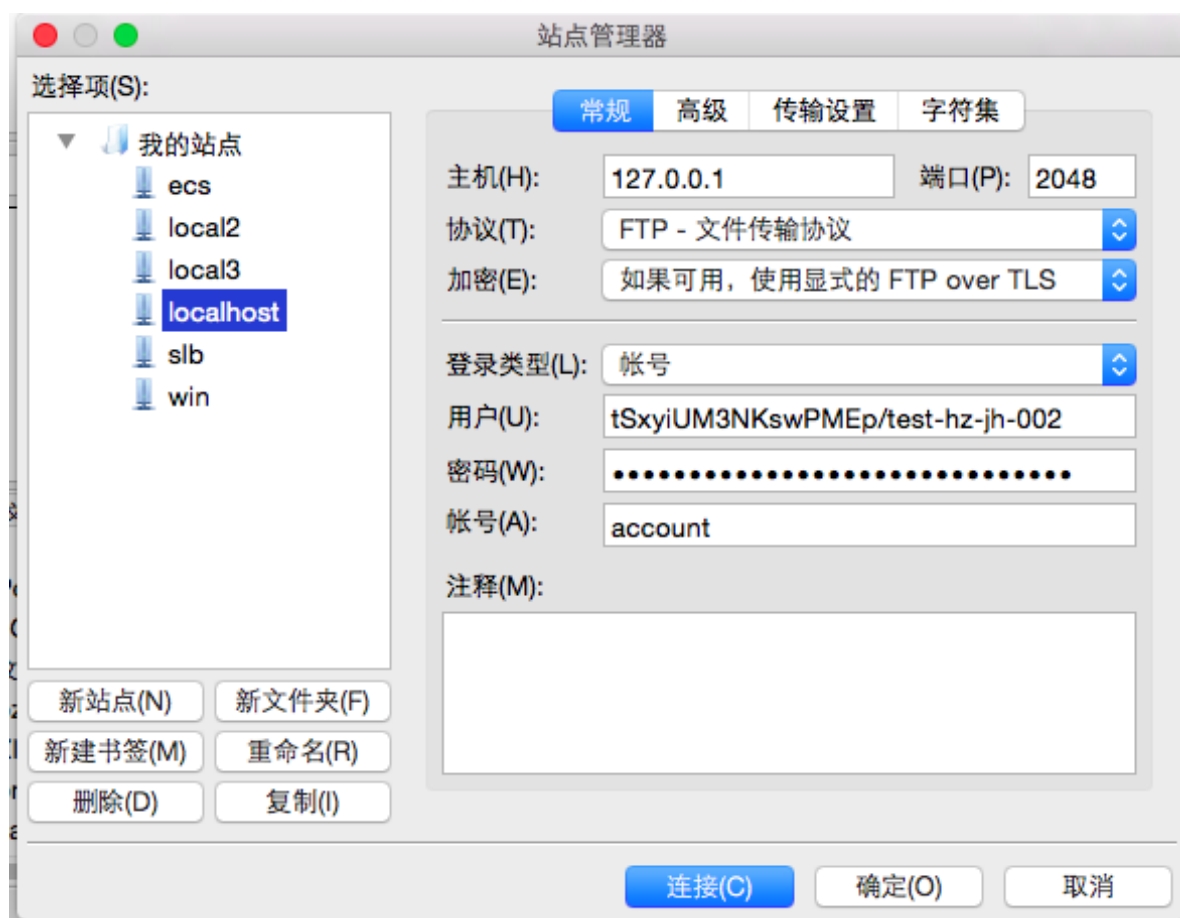
连接到FTP server

推荐使用[FileZilla客户端](#)去连接FTP server。下载安装后，按如下方式连接即可：

- 主机: 127.0.0.1
- 登录类型：正常
- 用户：access_key_id/bucket_name
- 密码：access_key_secret

**说明：**

- 用户中，/是必须的，如用户tSxyiUM3NKswPMEp/test-hz-jh-002。
- access_key_id和access_key_secret的获取，请参见 [OSS访问控制](#)。



高级使用

- 通过控制页面管理FTP server

— 修改监听地址

如果需要通过网络来访问FTP server, 那么需要修改监听地址, 因为默认的监听地址127.0.0.1只允许来自本地的访问。可以修改成内网ip或公网ip。

— 修改监听端口

修改FTP server监听的端口, 建议端口大于1024, 因为监听1024以下的端口时需要管理员权限。

— 修改日志等级

设置FTP server的日志级别。FTP server的日志会输出到data/ossftp/目录下, 可以通过控制页面的日志按钮在线查看。默认的日志级别为INFO, 打印的日志信息较少, 如果需要更详细的日志信息, 可以修改为DEBUG模式。如果希望减少日志的输出, 可以设置级别为WARNING或ERROR等。

— 设置Bucket endpoints

FTP server默认会探索bucket的所属location信息, 随后将请求发到对应的region (如oss-cn-hangzhou.aliyuncs.com或oss-cn-beijing.aliyuncs.com), FTP server会优先尝试内网访问oss。如果您设置了bucket endpoints, 如设置为test-bucket-a.oss-cn-hangzhou.aliyuncs.com, 那么当访问test-bucket-a时, 就会使用oss-cn-hangzhou.aliyuncs.com域名。

— 设置显示语言

通过设置cn/en, 可修改FTP控制页面的显示语言为中文/英文。



说明：

- 所有修改都需要重启才能生效。
- 上述的所有修改其实都是修改的ftp根目录下的config.json, 所以您可以直接修改该文件。

- 直接启动FTP server(Linux/Mac)

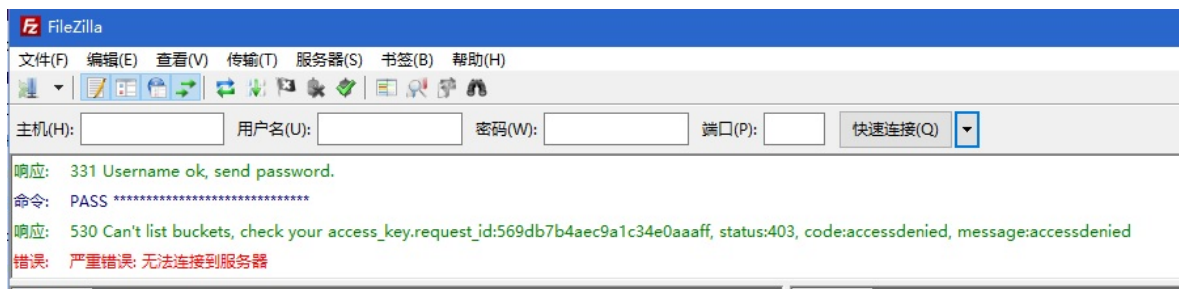
可以直接启动ossftp目录下的ftpserver.py, 免去web_server的开销。

```
$ python ossftp/ftpserver.py &
```

配置修改方式同上。

可能遇到的问题

- 如果连接FTP server时，遇到以下错误：



有两种可能：

- 输入的 `access_key_id` 和 `access_key_secret` 有误。

解决：请输入正确的信息后再重试。

- 所用的 `access_key` 信息为 `ram` 子账户的 `access_key`，而子账户不具有 `List buckets` 权限。

解决：当使用子账户访问时，请在控制页面中指定 `bucket endpoints`，即告诉 FTP server 某个 `bucket` 应该用什么 `endpoint` 来访问。同时，子账户也需要一些必须的权限，关于使用 `ram` 访问 `oss` 时的访问控制，请参考文档[访问控制](#)。具体如下。

■ 只读访问

OSS FTP 工具需要的权限列表为 `ListObjects`、`GetObject`、`HeadObject`。关于如何创建一个具有只读访问的 `ram` 子账户，请参考图文教程[如何结合 ram 实现文件共享](#)。

■ 上传文件

如果允许 `ram` 子账户上传文件，还需要 `PutObject`。

■ 删除文件

如果允许 `ram` 子账户删除文件，还需要 `DeleteObject`。

- 如果您在Linux下运行FTP server，然后用FileZilla连接时遇到如下错误：

```
501 can't decode path (server filesystem encoding is ANSI_X3.4-1968)
```

一般是因为本地的中文编码有问题。在将要运行start.sh的终端中输入下面的命令，然后再重新启动即可。

```
$ export LC_ALL=en_US.UTF-8; export LANG="en_US.UTF-8"; locale
```

3.2 Discuz如何存储远程附件到OSS

前言

网站远程附件功能是指将用户上传的附件直接存储到远端的存储服务器，一般是通过FTP的方式存储到远程的FTP服务器。

目前Discuz论坛、phpwind论坛、Wordpress个人网站等都支持远程附件功能。

本文介绍如何基于Discuz论坛存储远程附件。

准备工作

申请OSS账号，并且创建一个public-read的bucket。这里需要权限为public-read是因为后面需要匿名访问。

详细步骤

测试所用Discuz版本为Discuz! X3.1，以下是详细的设置流程：

1. 登录Discuz站点，进入管理界面后，先点击全局，再点击上传设置，如下图所示。



Discuz! Control Panel

首页 全局 界面 内容 用户 门户 论坛 群组 防灌水 运营 应用 工具 站长 UCenter

全局 » 站点信息 [+]

站点信息

站点名称: Discuz! Board 站点名称, 将显示在浏览器窗口标题等位置

网站名称: Comsenz Inc. 网站名称, 将显示在页面底部的联系方式处

网站 URL: http://www.comsenz.com/ 网站 URL, 将作为链接显示在页面底部

管理员邮箱: admin@admin.com 管理员 E-mail, 将作为系统发邮件的时候的发送人地址

QQ在线客服号码: 设置我的QQ在线状态

网站备案信息代码: 页面底部可以显示 ICP 备案信息, 如果网站已备案, 在此输入您的授权码, 它将显示在

2. 选择远程附件, 然后开始设置。



Discuz! Control Panel

首页 全局 界面 内容 用户 门户 论坛 群组 运营 应用 工具 云平台 站长 UCenter

全局 » 上传设置 [+]

上传设置

基本设置 远程附件 论坛附件 空间附件 门户附件

启用远程附件: ☒ 是 ☐ 否 1

启用 SSL 连接: ☐ 是 ☒ 否 2 注意: FTP 服务器必需开启了 SSL

FTP 服务器地址: 121.40.198.184 3 可以是 FTP 服务器的 IP 地址或域名

FTP 服务器端口: 2048 4 默认为 21

FTP 帐号: tSxyiUM3NKswPMEp/test-hz-jh-002 5 该帐号必需具有以下权限: 读取文件、写入文件、删除文件、创建目录、子目录继承

FTP 密码: R*****8 6 基于安全考虑将只显示 FTP 密码的第一位和最后一位, 中间显示八个 * 号

被动模式(pasv)连接: ☒ 是 ☐ 否 7 一般情况下非被动模式即可, 如果存在上传失败问题, 可尝试打开此设置



说明:

- 启用远程附件, 请选择是。
- 启用SSL连接, 请选择否。

- FTP服务器地址，即运行ossftp工具的地址，通常填写 127.0.0.1 即可。
- FTP服务的端口号，默认为 2048。
- FTP登录用户名，格式为 AccessKeyID/BucketName。注意这里的 / 不是或的意思。
- FTP的登录密码，为 AccessKeySecret。
- 被动模式连接，选择默认的是即可。



Discuz! Control Panel

首頁 全局 界面 内容 用戶 門戶 論壇 群組 防灌水 運營 應用 工具 站長 UCenter

全局 » 上傳設置 [+]

上傳設置 基本設置 遠程附件 論壇附件 空間附件 門戶附件

遠程附件目錄:

8 遠程附件目錄的絕對路徑或相對於 FTP 主目錄的相對路徑，結尾不要加斜槓「/」，「.»表示 FTP 主目錄

遠程訪問 URL:

9 支持 HTTP 和 FTP 協議，結尾不要加斜槓「/」；如果使用 FTP 協議，FTP 服務器必需支持 PASV 模式，無限

FTP 傳輸超時時間:

10 單位：秒，0 為服務器默認

連接測試:

11 無需保存設置即可測試，請在測試通過後再保存

允許的附件擴展名:

只允許這些擴展名結尾的附件使用遠程附件功能，每行一個，不區分大小寫，留空為不限制
雙擊輸入框可擴大/縮小

禁止的附件擴展名:

禁止這些擴展名結尾的附件使用遠程附件功能，每行一個，不區分大小寫，留空為不限制
雙擊輸入框可擴大/縮小

Powered by Discuz! X3.1



说明：

- 远程附件目录，填 . 表示在Bucket的根目录下创建上传目录。
- 远程访问URL, 填 http://BucketName.Endpoint即可。



说明：

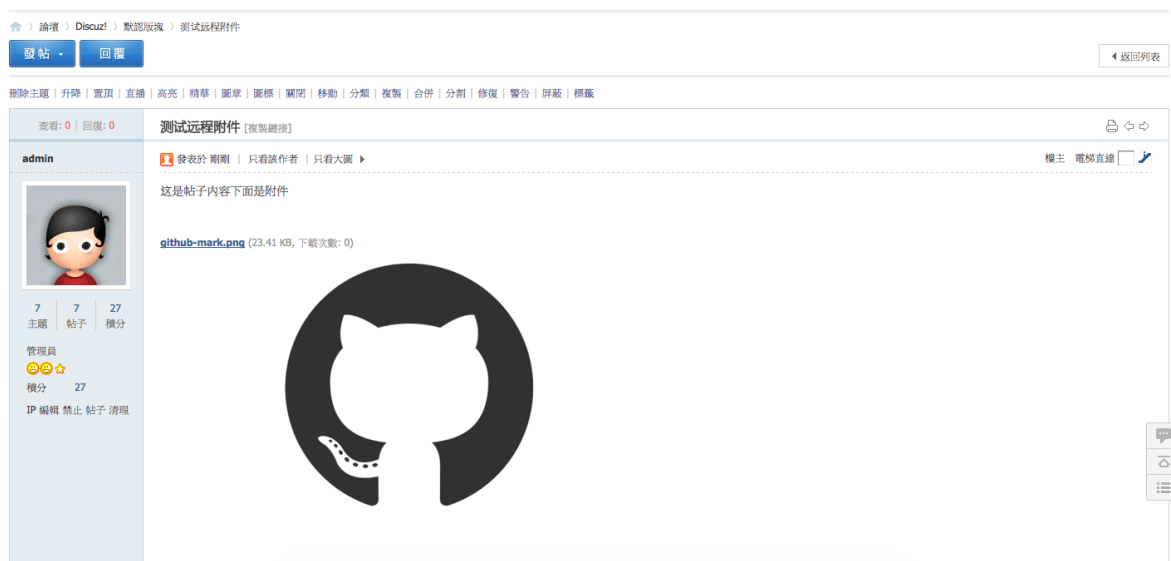
这里测试所用bucket为test-hz-jh-002, 属于杭州区域的，所以这里填写的是http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com。注意BucketName要和Endpoint匹配。

- 超时时间，设置为0即可，表示服务默认。
- 完成设置后，可以点击测试远程附件，如果成功则会出现如下画面。

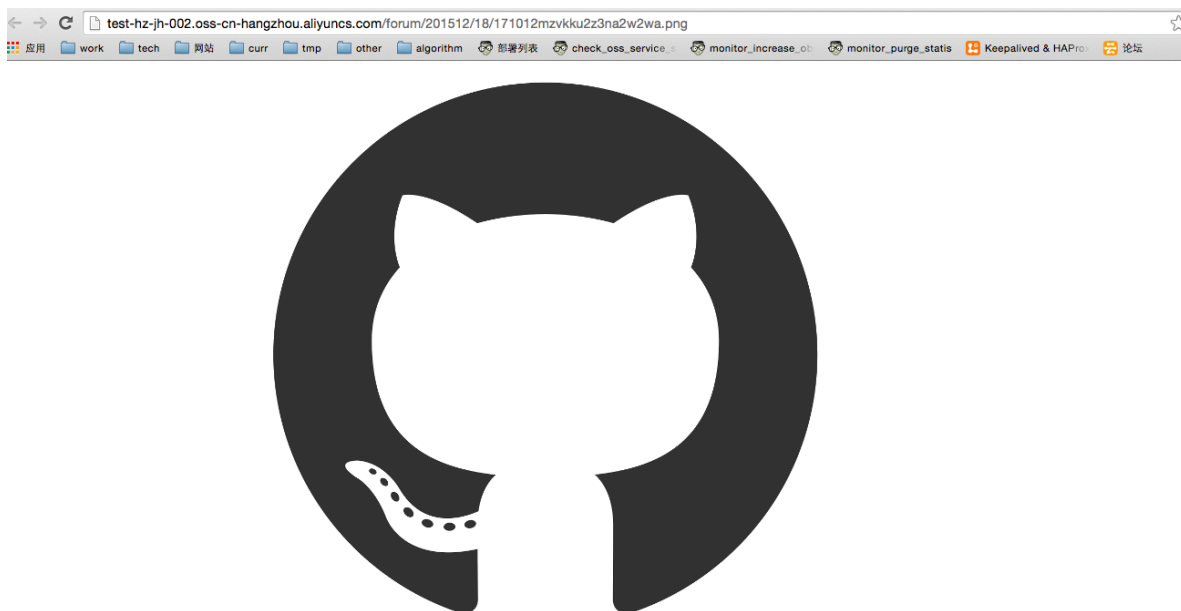


3. 发帖验证

发帖时上传图片附件如下所示。



在图片上右键点击，选择在“新建标签页中打开图片”，如下所示。



这就表示图片已经上传到了OSS的test-hz-jh-002中。

3.3 Phpwind如何存储远程附件到oss

前言

网站远程附件功能是指将用户上传的附件直接存储到远端的存储服务器，一般是通过FTP的方式存储到远程的FTP服务器。

目前Discuz论坛、phpwind论坛、Wordpress个人网站等都支持远程附件功能。

本文介绍如何基于Phpwind论坛存储远程附件。

准备工作

申请OSS账号，并且创建一个public-read的bucket。这里需要权限为public-read是因为后面需要匿名访问。

详细步骤

测试所用版本为phpwind8.7, 以下为详细设置流程。

1. 登录站点

进入管理界面，依次选择全局 > 上传设置 > 远程附件。



2. 开始设置



说明：

- 使用FTP上传，请选择开启。
- 站点附件地址，填写<http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com>，属于杭州区域的#所以这里填写的是<http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com>，注意BucketName要和Endpoint匹配。
- FTP服务器地址，即运行ossftp工具的地址，一般填127.0.0.1即可。
- FTP服务的端口号，默认为2048。
- FTP上传目录，默认填.即可，表示在bucket的根目录开始创建附件目录。
- FTP登录用户名，格式为AccessKeyID/BucketName。注意这里的/不是或的意思。

- FTP的登录密码，为AccessKeySecrete关于AccessKeyID和AccessKeySecrete的获取，可以登录阿里云控制台的Access Key管理进行查看。
- FTP超时时间，设置为10就表示如果10秒内请求没有返回结果就会超时返回。

3. 发帖验证

phpwind不能在设置完成后直接点击测试，我们这里发带图片的帖子来验证下。



在图片点击右键，在新建标签页中打开图片，可以看到下图。



通过图中的URL，我们可以判断图片已经上传到了OSS的test-hz-jh-002 Bucket中。

3.4 Wordpress如何存储远程附件到oss

前言

网站远程附件功能是指将用户上传的附件直接存储到远端的存储服务器，一般是通过FTP的方式存储到远程的FTP服务器。

目前Discuz论坛、phpwind论坛、Wordpress个人网站等都支持远程附件功能。

本文介绍如何基于Wordpress论坛存储远程附件。

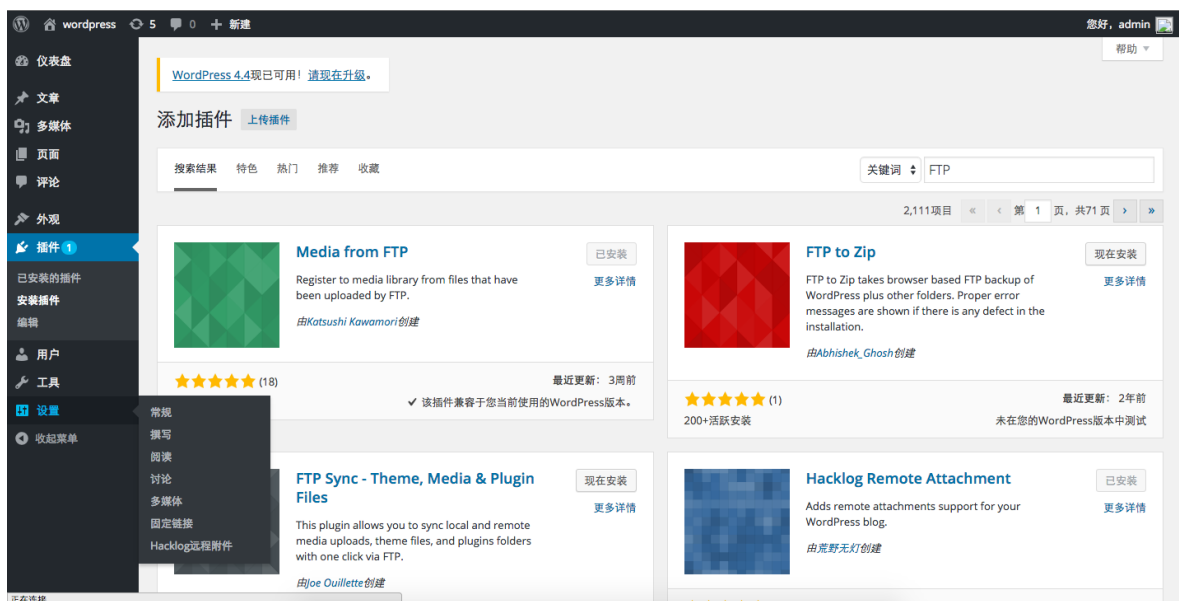
准备工作

申请OSS账号，并且创建一个public-read的bucket。这里需要权限为public-read是因为后面需要匿名访问。

详细步骤

wordpress本身是不支持远程附件功能的，但是可以通过第三方的插件来做远程附件。作者所用wordpress版本为4.3.1, 所用插件为Hacklog Remote Attachment，以下为具体设置步骤。

1. 登录wordpress站点，选择安装插件，搜关键词FTP，选择Hacklog Remote Attachment安装。



2. 设置。

- FTP服务器地址, 即运行ossftp工具的地址, 一般填127.0.0.1即可。
- FTP服务的端口号, 默认为2048。
- FTP登录用户名, 格式为AccessKeyID/BucketName, 注意这里的/不是或的意思。
- FTP的登录密码为AccessKeySecret。



说明：

关于AccessKeyID和AccessKeySecret的获取, 可以登录阿里云控制台的Access Key管理进行查看。

- FTP超时时间, 默认设置为30秒即可。
- 远程基本URL填 `http://BucketName.Endpoint/wp`。这里测试所用bucket为test-hz-jh-002, 属于杭州区域的, 所以这里填写的是`http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com/wp`。
- FTP远程路径, 填wp表示所有附件都会存储在bucket的wp目录下。注意6和7要对应起来。
- HTTP远程路径, 填.即可。

具体信息见下图。



3. 验证。

设置好之后，点击保存的同时，会做测试，测试结果会在页面上方显示，如下图所示表示测试成功。

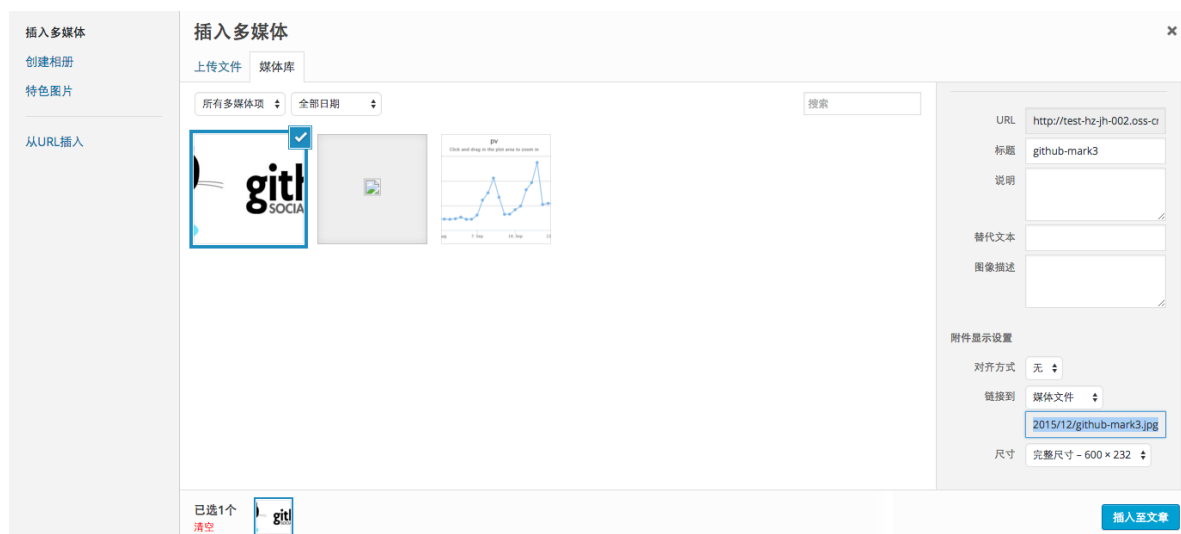


4. 发布新文章，并插入图片。

现在开始写一篇新文章，并测试远程附件。创建好文章后，点击添加媒体来上传附件。



上传附件如下图所示。



5. 上传完附件，点击发布，即可看到文章了。



仍然通过右键点击图片，通过新建链接来打开图片即可看到图片的URL如下图所示。



通过图片的URL，我们可以判定图片已经成功上传到了OSS。

3.5 如何结合RAM实现文件共享

简介

本文主要介绍如何结合RAM服务，共享用户bucket中的文件/文件夹，让其他用户只读，而bucket的owner可以修改。

思路就是：开通RAM -> 新建只读授权策略 -> 创建子用户 -> 向子用户授权 -> FTP登录验证

获取账户ID

获取你的account ID。具体参考下图步骤。



开通RAM

访问控制（Resource Access Management，RAM）是一个稳定可靠的集中式访问控制服务。您可以通过定制策略，产生一个共享读的账户，使得用户可以用此账户登录FTP 工具并 读取您的文件。

RAM的位置请参考下图。



新建授权策略

开通RAM之后，进入RAM控制台，点击左侧的授权管理，按下图步骤依次操作来创建新的授权策略。

填写授权策略时，如下图所示。

创建授权策略

STEP 1: 选择权限策略模板

STEP 2: 编辑权限并提交

STEP 3: 新建成功

授权策略名称: 1

read-list-test-hz-john-001

长度为1-128个字符, 允许英文字母、数字, 或"-"

备注: 2

read-list-test-hz-john-001

策略内容: 3

```

1  {
2    "Version": "1",
3    "Statement": [
4      {
5        "Action": [
6          "oss:GetObject",
7          "oss:HeadObject"
8        ],
9        "Resource": [
10         "acs:oss:*:1884285416482090:test-hz-
john-001/*"
11       ],
12       "Effect": "Allow"
13     },
14     {
15       "Action": [

```

[授权策略格式定义](#)
[授权策略常见问题](#)

4

上一步

新建授权策略

取消

其中第1步和第2步自定义填写即可，第3步的策略内容才是最关键的地方。

```

{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "oss:GetObject",
        "oss:HeadObject"
      ],
      "Resource": [
        "acs:oss:*:*****:test-hz-john-001/*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "oss:ListObjects",
        "oss:GetBucketAcl",
        "oss:GetBucketLocation"
      ],
      "Resource": [
        "acs:oss:*:*****:test-hz-john-001"
      ],
      "Effect": "Allow"
    }
  ]
}

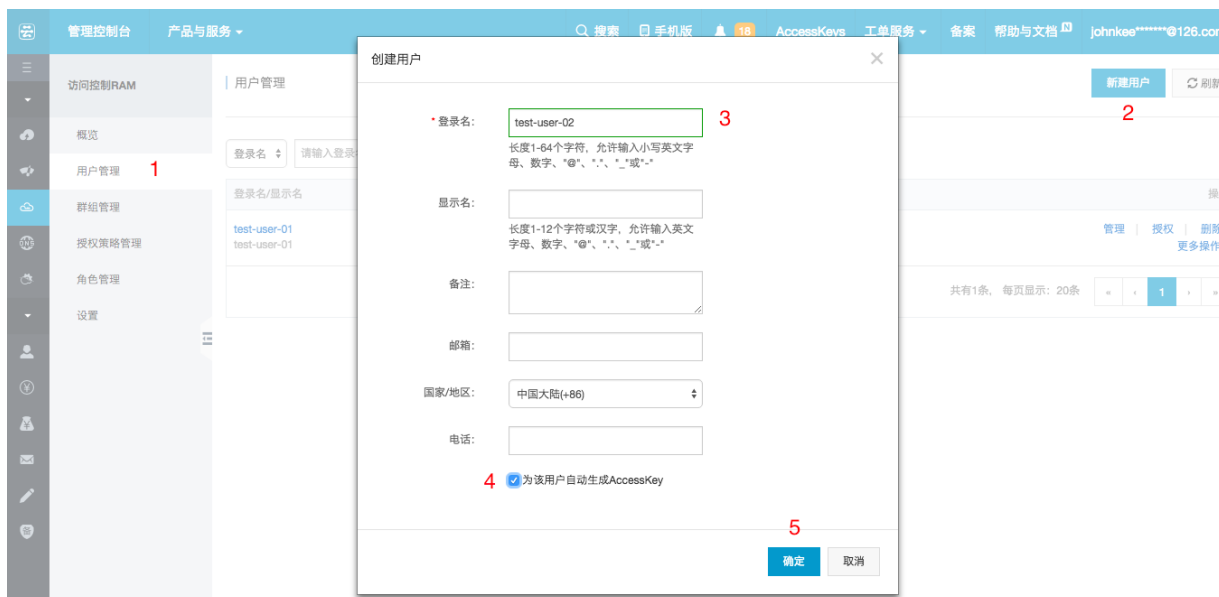
```

[illegible]

将上面的*****替换为你自己的账户ID，test-hz-john-001替换为你自己的bucket名，然后整体拷贝到策略内容里面，最后点击新建授权策略即可。

创建用户

上面的授权策略生命了一种只读策略，下面新建一个用户并给予他这种策略。先是创建用户，步骤如下：



说明：

注意保存新用户的access_key。

给用户授权

下面将之前创建的策略授权给该用户。



用子账户登录

用子账户的access_key和之前授权策略中的bucket登录，即可下载文件夹或文件，但上传会失败。

4 ossfs

4.1 快速安装

ossfs 能让您在Linux系统中把OSS bucket 挂载到本地文件系统中，您能够便捷地通过本地文件系统操作OSS 上的对象，实现数据的共享。

主要功能

ossfs 基于s3fs 构建，具有s3fs 的全部功能。主要功能包括：

- 支持POSIX 文件系统的大部分功能，包括文件读写，目录，链接操作，权限，uid/gid，以及扩展属性 (extended attributes)
- 通过OSS 的multipart 功能上传大文件。
- MD5 校验保证数据完整性。

局限性

ossfs提供的功能和性能和本地文件系统相比，具有一些局限性。具体包括：

- 随机或者追加写文件会导致整个文件的重写。
- 元数据操作，例如list directory，性能较差，因为需要远程访问OSS服务器。
- 文件/文件夹的rename操作不是原子的。
- 多个客户端挂载同一个OSS bucket时，依赖用户自行协调各个客户端的行为。例如避免多个客户端写同一个文件等等。
- 不支持hard link。
- 不适合用在高并发读/写的场景，这样会让系统的load升高。

安装及使用

- 安装包下载

Linux发行版	下载
Ubuntu 16.04 (x64)	ossfs_1.80.5_ubuntu16.04_amd64.deb
Ubuntu 14.04 (x64)	ossfs_1.80.5_ubuntu14.04_amd64.deb
CentOS 7.0 (x64)	ossfs_1.80.5_centos7.0_x86_64.rpm
CentOS 6.5 (x64)	ossfs_1.80.5_centos6.5_x86_64.rpm

由于低版本的Linux发行版本内核版本比较低，ossfs进程在运行过程中容易出现掉线或者其他问题，因此建议用户将操作系统升级到CentOS 7.0或者Ubuntu 14.04以及以上版本。

- 安装方法

— 对于Ubuntu，安装命令为：

```
sudo apt-get update
sudo apt-get install gdebi-core
sudo gdebi your_ossfs_package
```

— 对于CentOS6.5及以上，安装命令为：

```
sudo yum localinstall your_ossfs_package
```

— 对于CentOS5，安装命令为：

```
sudo yum localinstall your_ossfs_package --nogpgcheck
```

- 使用方法

设置bucket name 和 AccessKeyId/Secret信息，将其存放在/etc/passwd-ossfs 文件中，注意这个文件的权限必须正确设置，建议设为640。

```
echo my-bucket:my-access-key-id:my-access-key-secret > /etc/passwd-ossfs
chmod 640 /etc/passwd-ossfs
```

将OSS bucket mount到指定目录。

```
ossfs my-bucket my-mount-point -ourl=my-oss-endpoint
```

示例

将my-bucket这个bucket挂载到/tmp/ossfs目录下，AccessKeyId是faint

，AccessKeySecret是123，oss endpoint是http://oss-cn-hangzhou.aliyuncs.com

```
echo my-bucket:faint:123 > /etc/passwd-ossfs
chmod 640 /etc/passwd-ossfs
mkdir /tmp/ossfs
ossfs my-bucket /tmp/ossfs -ourl=http://oss-cn-hangzhou.aliyuncs.com
```



说明：

如果您使用在阿里云购买的云虚拟机主机（ECS）来提供ossfs服务，您可以使用内网域名，比如在这个例子您可以将oss endpoint 改成oss-cn-hangzhou-internal.aliyuncs.com，这样可以节省带宽方面的费用。OSS的内网域名请参考[访问域名和数据中心](#)。

卸载bucket

```
fusermount -u /tmp/ossfs
```

更多详细内容请参考：[GitHub ossfs](#)

版本日志

请参考：[GitHub ChangeLog](#)

4.2 FAQ

- Q: ossfs适合什么样的程序？
 - ossfs能把oss bucket挂载到本地，如果您使用的软件没有支持OSS，但您又想让数据能自动同步到OSS，那么ossfs是很好的选择。
- Q: ossfs有什么局限性？
 - 由于数据需要经过网络同步到云端，ossfs在性能和功能上可能与本地文件系统有差距。如果您想让数据库等对io要求很高的应用跑在ossfs挂载的盘上，请慎重考虑。和本地文件系统具体差异：
 - 随机或者追加写文件会导致整个文件的重写。
 - 元数据操作，例如list directory，性能较差，因为需要远程访问OSS服务器。
 - 文件/文件夹的rename操作不是原子的。
 - 多个客户端挂载同一个oss bucket时，依赖用户自行协调各个客户端的行为。例如避免多个客户端写同一个文件等等。
 - 不支持hard link。
- Q: ossfs一定要阿里云的机器才能用么？
 - ossfs不限制一定要阿里云的内网才可以使用，外网机器依然可以使用。
- Q: ossfs能不能同时挂载多个OSS bucket？
 - 可以的，在passwd-ossfs文件中写入多个OSS配置信息即可。支持不同帐号的OSS。
- Q: 我在yum/apt-get安装ossfs，遇到conflicts with file from package fuse-devel的错误，请问是怎么回事？
 - 您的系统中存在老版本的fuse，请先使用相关的包管理器卸载，再重新安装ossfs。
- Q: ossfs工作不正常，如何debug？

- 您可以使用在挂载时，加上-d -o f2参数，ossfs会把日志写入到系统日志中。在centos系统中，在/var/log/messages中。
- 您也可以使用在挂载时使用-f -d -o f2参数，ossfs会把日志输出到屏幕上。
- Q: 为什么我在mount时遇到 ossfs: unable to access MOUNTPOINT /tmp/ossfs: Transport endpoint is not connected这样的错误？
 - 请先umount对应的目录。
 - 请检查您在使用ossfs挂载时，填入的url参数是否正确，是否和bucket/access key id/access key secret匹配。
 - 特别注意：url中不包含bucket的名字。例如：您在oss控制台中看到bucket的域名是这样的：
`ossfs-test-1.oss-cn-hangzhou.aliyuncs.com`。那么填入的url则是：`http://oss-cn-hangzhou.aliyuncs.com`。
- Q: ossfs提示ossfs: unable to access MOUNTPOINT /tmp/odat: No such file or directory
 - 这是您未创建该目录导致的，在挂载前需要创建对应目录。
- Q: 我把bucket挂载到本地后，ls目录，却收到operation not permitted错误，这是为什么？
 - 请检查您的bucket中，是否包含目录名含有不可见字符的OSS object。文件系统对文件/目录名有更严格的限制，因此会收到上述错误。使用其他工具对这些object重命名后，ls就能正确显示目录内容了。
- Q: 我的一个目录下有非常多的文件，为什么ls该目录很慢？
 - 假设一个目录下有n个文件，那么ls该目录至少需要n次OSS http requests。在文件非常多的时候，这可能造成严重的性能问题。
 - 您可以采用下面两个办法优化：
 - 通过-omax_stat_cache_size=xxx参数增大stat cache的size，这样第一次ls会较慢，但是后续的ls就快了，因为文件的元数据都在本地cache中。默认这个值是1000，大约消耗4MB内存，请根据您的机器内存大小调整为合适的值。
 - 使用ls -f命令，这个命令会消除与OSS的n次http请求。
 - 具体参见[issue 13](#)
- Q: ossfs挂载时如何设置权限？
 - 如果要允许其他用户访问挂载文件夹，可以在运行ossfs的时候指定allow_other参数：
 - `ossfs your_bucket your_mount_point -ourl=your_endpoint -o`

```
allow_other
```

— 为什么使用allow_other参数，仍然不能访问文件？

■ 注意：allow_other是赋予挂载目录其他用户访问的权限，不是里面的文件！如果您要更改文件夹中的文件，请用chmod命令。

— allow_other默认赋予挂载目录777权限，我想让挂载目录的权限为770，该怎么办？

■ 可以通过umask来设置，参见[这里](#)。

• Q: 如果要使挂载的文件夹(/tmp/ossfs)属于某个user:

— 方法一：如果要使挂载的文件夹(/tmp/ossfs)属于某个user，则需要以user的身份创建挂载文件夹和使用ossfs：

```
■ sudo -u user mkdir /tmp/ossfs
```

```
■ sudo -u user ossfs bucket-name /tmp/ossfs
```

— 方法二：首先通过id命令获得指定用户的uid/gid信息。例如获取www用户的uid/gid信息：id www；然后挂载时指定uid/gid参数：

```
■ ossfs your_bucket your_mountpoint -ourl=your_url -ouid=your_uid  
-ogid=your_gid
```

注意：uid/gid都是数字。

• Q: 我不是root用户，如何umount ossfs挂载的目录

— fusermount -u your_mountpoint

• Q: 如何开机自动挂载ossfs？

— Step 1 首先请参考使用说明，把bucket name，access key id/secret等信息写入/etc/passwd-ossfs，并将该文件权限修改为640。

```
■ echo your_bucket_name:your_access_key_id:your_access_key_secret  
>  
/etc/passwd-ossfs
```

```
■ chmod 640 /etc/passwd-ossfs
```

• Step 2 接下来针对不同的系统版本，设置方式有所不同

— Step 2A 通过fstab的方式自动mount (适用于ubuntu14.04, centos6.5)

■ 在/etc/fstab中加入下面的命令

```
■ ossfs#your_bucket_name your_mount_point fuse _netdev,url=  
your_url,allow_other 0
```

0

- 其中上述命令中的your_xxx信息需要根据您的bucket name等信息填入。
- 保存/etc/fstab文件。执行mount -a命令，如果没有报错，则说明设置正常。
- 到这一步，ubuntu14.04就能自动挂载了。centos6.5还需要执行下面的命令：
- `chkconfig netfs on`

— Step 2B 通过开机自启动脚本mount (适用于centos7.0及以上的系统)

- 在/etc/init.d/目录下建立文件ossfs，把模板文件中的内容拷贝到这个新文件中。并将其中的your_xxx内容改成您自己的信息。
- 执行命令：`chmod a+x /etc/init.d/ossfs`
- 上述命令是把新建的ossfs脚本赋予可执行权限。您可以执行该脚本，如果脚本文件内容无误，那么此时oss中的bucket已经挂载到您指定的目录下了。
- 执行命令：`chkconfig ossfs on`
- 上述命令是把ossfs启动脚本作为其他服务，开机自动启动。
- 好了，现在ossfs就可以开机自动挂载了。总结起来，如果您是ubuntu14.04和centos6.5，您需要执行Step 1 + Step 2A；如果您是centos7.0系统，您需要执行Step 1 + Step 2B。

- Q: 遇到fusermount: failed to open current directory: Permission denied错误如何解决？

— 这是fuse的一个bug，它要求当前用户对当前目录（非挂载目录）有读权限。解决的办法就是cd到一个有读权限的目录再运行ossfs命令

- Q: 我需要以www用户挂载ossfs，此时如何设置开机自动挂载？

— 参照上面的问题的解答，Step 1照做，对Step 2B稍加修改，修改/etc/init.d/ossfs中的命令为：

```
sudo -u www ossfs your_bucket your_mountpoint -ourl=your_url
```

— 设置自启动脚本中允许使用sudo，编辑/etc/sudoers，将其中的Defaults requiretty这行改为#Defaults requiretty（注释掉）

- Q: 遇到fusermount: failed to open current directory: Permission denied错误如何解决？

— 这是fuse的一个bug，它要求当前用户对当前目录（非挂载目录）有读权限。解决的办法就是cd到一个有读权限的目录再运行ossfs命令。

- Q: 使用ECS挂载ossfs，如何避免因后台程序扫描文件而产生费用？

— 程序扫描ossfs挂载的目录，会转换成向OSS的请求，如果请求次数很多，会产生费用（1分钱/1万次）。如果是`updatedb`，可以通过修改`/etc/updatedb.conf`让它跳过。具体做法是：

1. 在`PRUNEFS` =后面加上`fuse.ossfs`
2. 在`PRUNEPATHS` =后面加上挂载的目录

— 如何确定是哪个进程扫了我的目录？

1. 首先安装auditd: `sudo apt-get install auditd`
2. 启动auditd: `sudo service auditd start`
3. 设置监视挂载目录: `auditctl -w /mnt/ossfs`
4. 在auditlog中可以查看是哪些进程访问了这个目录：`ausearch -i | grep /mnt/ossfs`

- Q: 使用ossfs上传到OSS的文件Content-Type全是”application/octet-stream”是怎么回事？

— ossfs通过查询`/etc/mime.types`中的内容来确定文件的Content-Type，请检查这个文件是否存在，如果不存在，则需要添加：

1. 对于ubuntu可以通过`sudo apt-get install mime-support`来添加
2. 对于centos可以通过`sudo yum install mailcap`来添加
3. 也可以手动添加，每种格式一行，每行格式为：`application/javascript js`

- Q: 如何使用supervisor启动ossfs？

1. 安装supervisor，在ubuntu中执行`sudo apt-get install supervisor`
2. 建立一个目录，编辑ossfs的启动脚本：

```
mkdir /root/ossfs_scripts
vi /root/ossfs_scripts/start_ossfs.sh
```

写入如下数据：

```
# 卸载
fusermount -u /mnt/ossfs
# 重新挂载，必须要增加-f参数运行ossfs，让ossfs在前台运行
exec ossfs my-bucket my-mount-point -ourl=my-oss-endpoint -f
```

3. 编辑`/etc/supervisor/supervisord.conf`，在最后加入下面一段：

```
[program:ossfs]
command=bash /root/ossfs_scripts/start_ossfs.sh
logfile=/var/log/ossfs.log
log_stdout=true
log_stderr=true
logfile_maxbytes=1MB
```



```
logfile_backups=10
```

4. 运行supervisor：

```
supervisord
```

```
supervisord
```

5. 确认一切正常：

```
ps aux | grep supervisor # 应该能看到supervisor进程
ps aux | grep ossfs # 应该能看到ossfs进程
kill -9 ossfs # 杀掉ossfs进程，supervisor应该会重启它，不要使用killall
，因为killall发送SIGTERM，进程正常退出，supervisor不再去重新运行ossfs
ps aux | grep ossfs # 应该能看到ossfs进程
```

如果出错，请检查/var/log/supervisor/supervisord.log和/var/log/ossfs.log。

- Q: 遇到“fuse: warning: library too old, some operations may not work”怎么办？

出现的原因是：ossfs编译时所使用的libfuse版本 比运行时链接到的libfuse版本高。这往往是用
户自行安装了libfuse导致的。使用我们提供的rpm包安装ossfs，无需再安装libfuse。

在CentOS-5.x和CentOS-6.x上我们提供的rpm包里包含了libfuse-2.8.4，如果在运行的时候环境
中有libfuse-2.8.3，并且ossfs被链接到了旧版本的fuse上，就会出现上述warning。

1. 如何确认ossfs运行时链接的fuse版本？

- 运行ldd \$(which ossfs) | grep fuse
- 例如结果是“/lib64/libfuse.so.2”，那么通过ls -l /lib64/libfuse*可以看到fuse的版本

2. 如何让ossfs链接到正确的版本？

- 首先通过rpm -ql ossfs | grep fuse找到libfuse的目录
- 例如结果是“/usr/lib/libfuse.so.2”，则通过LD_LIBRARY_PATH=/usr/lib ossfs ...运行ossfs

3. 我能忽略这个WARNING吗？

- 最好不要，见这个bug
- Q: 为什么用ossfs看到的文件信息（例如大小）与其他工具看到的不一致？

因为ossfs默认会缓存文件的元信息（包括大小/权限等），这样就不需要每次ls的时候向OSS发送请
求，加快速度。如果用户通过其他程序（例如SDK/官网控制台/osscli等）对文件进行了修改，那
么有可能在ossfs中看到的文件信息 没有及时更新。

如果想禁止ossfs的缓存，那么可以在挂载的时候加上如下参数：`-omax_stat_cache_size=0`

5 RAM策略编辑器

地址

[RAM Policy Editor](#)

使用

RAM授权策略由若干条规则组成，使用RAM策略编辑器，可以在界面上逐条添加/删除规则，并自动生成策略的JSON文本。用户添加完所有规则后，只需要将JSON文本拷贝，然后粘贴到访问控制（RAM）控制台的创建授权策略内容框内。

具体操作请参见 [创建自定义授权策略](#)。

RAM策略编辑器中，每条规则需要设置其Effect、Actions、Resources和Conditions：

- Effect

指定这条规则是允许访问（Allow）还是禁止访问（Deny）。

- Actions

指定访问资源的动作，可以选择多项。一般来说用户使用提供的通配动作就足够了：

- `oss:*` 表示允许所有动作。
- `oss:Get*` 表示允许所有的读动作。
- `oss:Put*` 表示允许所有的写动作。

更多信息请参见 [RAM Policy Editor README](#)。

- Resources

指定授权访问的OSS的资源，可以指定多个，每个是以下形式：

- 表示某个bucket: `my-bucket`（此时对bucket下的文件没有权限）
- 表示某个bucket下面所有文件: `my-bucket/*`（此时对bucket本身没有权限，例如ListObjects）
- 表示某个bucket下某个目录: `my-bucket/dir`（此时对dir/下面的文件没有权限）
- 表示某个bucket下某个目录下面所有文件: `my-bucket/dir/*`（此时对dir没有权限，例如ListObjects）
- 填写完整的资源路径: `acs:oss:*:1234:my-bucket/dir`，其中1234为用户的User ID（在控制台查看）

EnablePath

当用户需要对某个目录授权时，往往还需要保证对上一层目录也有List权限，例如用户对my-bucket/users/dir/*赋予读写权限，为了在控制台（或其他工具）能够查看这个目录，用户还需要以下权限：

```
ListObjects my-bucket
ListObjects my-bucket/users
ListObjects my-bucket/users/dir
```

勾选EnablePath选项时，上面这些权限会自动添加。

- Conditions

指定授权访问时应该满足的条件，可以指定多个。

更多信息请参见 [RAM Policy Editor README](#)。

例子

授权对my-bucket及其文件全部的权限：

RAM Policy Editor v1.0.4

Star 2

添加规则

Effect:

Actions:

Resources:

- 每添加一个Resource后按回车确认
- 例子: my-bucket, my-bucket/dir/
- [More...](#)

EnablePath ☐ 自动设置父目录权限 ?

Conditions (Optional)

授权策略

```
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "oss:*"
      ],
      "Resource": [
        "acs:oss:*:*:my-bucket",
        "acs:oss:*:*:my-bucket/*"
      ],
      "Condition": {}
    }
  ]
}
```

规则列表

Effect	Actions	Resources	Conditions
Allow	oss:*	acs:oss:*:*:my-bucket acs:oss:*:*:my-bucket/	

更多例子请参见 [RAM Policy Editor README](#)。

6 osscmd

6.1 快速安装

概述

osscmd是基于 Python 2.x 的命令行工具，支持Bucket管理、文件管理等功能。



说明：

非必要场景下建议使用 [ossutil](#) 代替osscmd。

- 使用场景
 - API级别的开发、调试，比如发送特定格式的请求、分步骤执行分片上传等。
 - Bucket配置，不方便使用控制台情况下的Bucket配置，如logging/website/lifecycle等。
 - 局限
 - osscmd支持的运行环境包括Python 2.5/2.6/2.7，不支持Python 3.x。
 - 在Python SDK 0.x基础上开发，Python SDK 0.x已经不再维护，目前维护的[Python SDK](#)是2.x.x。
 - OSSCMD不再支持新功能，只进行BUG修改，比如不支持低频存储/归档存储、跨区域复制、镜像回源等。
- 强烈建议使用 [ossutil](#) 代替osscmd，ossutil具有下列优势：
- ossutil支持Windows/Linux/Mac多种平台。
 - ossutil基于Go SDK实现，安装简单、性能优越。
 - ossutil命令简单、帮助丰富、支持中文/英文双语。

环境要求

osscmd是随Python sdk 0.x一起发布的，请点击[这里](#)下载。注意，Python SDK 2.x暂时并未提供相应版本的osscmd工具。

Python的版本要求要在2.5和2.7之间。SDK适用于Windows平台和Linux平台，但由于Python3.0并不完全兼容2.x的版本，所以SDK暂时不支持3.0及以上的版本。

安装好Python后：

- Linux shell环境下输入python并回车，来查看Python的版本。如下所示：

```
Python 2.5.4 (r254:67916, Mar 10 2010, 22:43:17)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-46)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- Windows在cmd环境下输入python并回车，来查看Python的版本。如下所示：

```
C:\Documents and Settings\Administrator>python
Python 2.7.5 (default, May 15 2013, 22:43:36) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

以上说明python安装成功。

- 异常情况，如Windows在cmd环境下输入python并回车后，提示“不是内部或外部命令”，请检查配置“环境变量”-“Path”，增加python的安装路径。如图：



如果没有安装Python，可以从[python官网](#)获取Python的安装包。网站有详细的安装说明来指导用户如何安装和使用Python。

安装使用

对下载的python SDK压缩包进行解压后，在osscmd所在目录直接执行python osscmd + 操作即可。比如上传一个文件到bucket：

```
python osscmd put myfile.txt oss://mybucket
```

需要特别说明的是osscmd中用oss://bucket或者oss://bucket/object表示这是一个bucket或者object。oss://只是一种资源的表示方式，没有其他意义。

如果需要详细的命令列表输入：python osscmd。

如果需要详细的参数列表说明输入：python osscmd help。

6.2 使用示例

安装配置osscmd

在Linux或者Windows上下载SDK安装包后，解压缩后就可以使用 `osscmd`了。

使用时直接调用`python osscmd`即可获得相应的说明。每种命令有两种执行模式。以查询用户所创建的bucket为例子。执行的是`gs`命令，`get service`的简写。

- 方法1：不指定ID和KEY，`osscmd`从默认文件中读取ID和KEY。

```
$ python osscmd gs
can't get accessid/accesskey, setup use : config --id=accessid --key=accesskey
```



说明：

如果出现这样的提示表明没有配置好ID和KEY，见步骤2中提示的配置命令。

如果配置好ID和KEY，并且ID和KEY有效，执行

```
$ python osscmd gs
2013-07-19 08:11 test-oss-sample
Bucket Number is: 1
```

- 方法2：直接在命令中指定ID和KEY，`osscmd`从命令行中读取ID和KEY。如果ID和KEY有效，执行后会得之后的结果。

```
$ python osscmd gs --id=your_id --key=your_key --host=your_endpoint
2013-07-19 08:11 test-oss-sample
Bucket Number is: 1
```

如果要配置用户的ID和KEY到默认的文件中，请运行如下命令用来配置访问OSS所需要的ID和KEY。默认的OSS HOST为`oss.aliyuncs.com`。

```
$python osscmd config --id=your_id --key=your_key --host=your_endpoint
```

如果出现类似“Your configuration is saved into ”的提示表明ID和KEY已经保存成功了。

基础操作

- 列出创建的bucket

```
$python osscmd getallbucket
```

如果是刚刚使用OSS的用户因为没有创建bucket，输出是空。

- 创建bucket

创建一个Bucket名字为mybucketname的bucket。

```
$python osscmd createbucket mybucketname
```

创建“mybucketname”的bucket，有可能不成功。因为OSS中的bucket名字是全局唯一的，并且有人已经创建了这个bucket。这个时候需要换一个名字。例如在bucket名字中加入特定的日期。

- 查看是否创建成功

```
$python osscmd getallbucket
```

如果没有成功请检查osscmd返回的错误信息。

- 查看Object

成功创建bucket后，查看bucket中有哪些object。

```
$python osscmd list oss://mybucketname/
```

由于bucket中还没有object，输出是空的。

- 上传object

向bucket中上传一个object。假如本地文件名叫local_existed_file，其MD5值如下所示。

```
$ md5sum local_existed_file 7625e1adc3a4b129763d580ca0a78e44  
local_existed_file  
$ python osscmd put local_existed_file oss://mybucketname/test_objec  
t
```



说明：

md5sum 为 Linux 命令，Windows下无此命令。

- 再次查看Object

如果创建成功，再次查看bucket中有哪些object。

```
$python osscmd list oss://mybucketname/
```

- 下载object

从bucket中下载object到本地文件，并比对下载的文件md5值

```
$ python osscmd get oss://mybucketname/test_object download_file  
$ md5sum download_file  
7625e1adc3a4b129763d580ca0a78e44 download_file
```



说明：

md5sum为 Linux 命令，Windows下无此命令。

- 删除object

```
$ python osscmd delete oss://mybucketname/test_object
```

- 删除bucket



说明：

如果bucket中还有object的话则这个bucket不能被删除。

```
$ python osscmd deletebucket mybucketname
```

使用lifecycle

- 配置一个lifecycle的xml格式的文本文件

```
<LifecycleConfiguration>
  <Rule>
    <ID>1125</ID>
    <Prefix>log_backup/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>2</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

这个表示删除bucket下，以log_backup/ 为前缀的，并且相对当前时间超过2天的object。详细的规则配置可以参考[API文档](#)

- 写入lifecycle

```
python osscmd putlifecycle oss://mybucket lifecycle.xml
0.150(s) elapsed
```

- 读取lifecycle

```
python osscmd getlifecycle oss://mybucket
<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration>
  <Rule>
    <ID>1125</ID>
    <Prefix>log_backup/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>2</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```



```
0.027(s) elapsed
```

- 删除lifecycle

```
python osscmd deletelifecycle oss://mybucket
0.139(s) elapsed
```

- 读取lifecycle

```
python osscmd getlifecycle oss://mybucket
Error Headers:
[('content-length', '288'), ('server', 'AliyunOSS'), ('connection',
'close'), ('x-oss-request-id', '54C74FEE5D7F6B24E5042630'), ('date',
'Tue, 27 Jan 2015 08:44:30 GMT'), ('content-type', 'application/
xml')]
Error Body:
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <BucketName>mybucket</BucketName>
  <Code>NoSuchLifecycle</Code>
  <Message>No Row found in Lifecycle Table.</Message>
  <RequestId>54C74FEE5D7F6B24E5042630</RequestId>
  <HostId>mybucket.oss-maque-hz-a.alibaba.net</HostId>
</Error>
Error Status:
404
getlifecycle Failed!
```

防盗链设置

- 允许空referer访问

```
$osscmd putreferer oss://test --allow_empty_referer=true
0.004(s) elapsed
```

- 获取设置的referer

```
$osscmd getreferer oss://test
<?xml version="1.0" encoding="UTF-8"?>
<RefererConfiguration>
  <AllowEmptyReferer>true</AllowEmptyReferer>
  <RefererList />
</RefererConfiguration>
```

- 不允许空referer，只允许referer为test的请求

```
$osscmd putreferer oss://test --allow_empty_referer=false --referer
='www.test.com'
0.092(s) elapsed
```

- 获取设置的referer

```
$osscmd getreferer oss://test
<?xml version="1.0" encoding="UTF-8"?>
<RefererConfiguration>
  <AllowEmptyReferer>false</AllowEmptyReferer>
  <RefererList>
```

```
<Referer>www.test.com</Referer>
</RefererList>
</RefererConfiguration>
```

- 不允许空referer,只允许referer为test和test1的请求

```
$osscmd putreferer oss://test --allow_empty_referer=false --referer
='www.test.com,www.test1.com'
```

- 获取设置的referer

```
$osscmd getreferer oss://test
<?xml version="1.0" encoding="UTF-8"?>
<RefererConfiguration>
  <AllowEmptyReferer>false</AllowEmptyReferer>
  <RefererList>
    <Referer>www.test.com</Referer>
    <Referer>www.test1.com</Referer>
  </RefererList>
</RefererConfiguration>
```

使用logging

- 设置logging

```
$osscmd putlogging oss://mybucket oss://myloggingbucket/mb
```

- 获取设置的logging

```
$osscmd getlogging oss://mybucket
```

6.3 有关Bucket命令

config

命令说明：

```
config --id=[accessid] --key=[accesskey] --host=[host] --sts_token=[
sts_token]
```

使用示范：

- `python osscmd config --id=your_id --key=your_key`
- `python osscmd config --id=your_id --key=your_key
--host=oss-internal.aliyuncs.com`

getallbucket(gs)

命令说明：

`getallbucket(gs)`

用来显示用户创建的bucket。gs是get service的简写。(gs)表示和getallbucket是同样的效果。

使用示范：

- `python osscmd getallbucket`
- `python osscmd gs`

createbucket(cb,mb,pb)

命令说明：

```
createbucket(cb,mb,pb) oss://bucket --acl=[acl]
```

创建bucket的命令，cb是create bucket的简写，mb是make bucket的简写，pb是put bucket的简写，oss://bucket表示bucket。--acl参数可以传入，也可以不传入。这几个命令都是同样的效果。

使用示范：

- `python osscmd createbucket oss://mybucket`
- `python osscmd cb oss://myfirstbucket --acl=public-read`
- `python osscmd mb oss://mysecondbucket --acl=private`
- `python osscmd pb oss://mythirdbucket`

deletebucket(db)

命令说明：

```
deletebucket(db) oss://bucket
```

删除bucket的命令，db是delete bucket的简写。deletebucket和db是同样的效果。

使用示范：

- `python osscmd deletebucket oss://mybucket`
- `python osscmd db oss://myfirstbucket`

deletewholebucket

注意：该命令十分危险，将会删除所有的数据，并且不可恢复。请慎重使用。命令说明：

```
deletewholebucket oss://bucket
```

删除bucket及其内部object以及multipart相关的内容。

使用示范：

- `python osscmd deletewholebucket oss://mybucket`

getacl

命令说明：

```
getacl oss://bucket
```

获取bucket的访问控制权限

使用示范：

- `python osscmd getacl oss://mybucket`

setacl

命令说明：

```
setacl oss://bucket --acl=[acl]
```

修改bucket的访问控制权限。acl只允许为private，public-read，public-read-write三个中的一个。

使用示范：

- `python osscmd setacl oss://mybucket --acl=private`

putlifecycle

命令说明：

```
putlifecycle oss://mybucket lifecycle.xml
```

设置lifecycle规则。其中lifecycle.xml为xml格式的lifecycle配置文件，详细的规则配置可以参考[API文档](#)

使用示范：

- `python osscmd putlifecycle oss://mybucket lifecycle.xml`

其中lifecycle.xml为XML格式的lifecycle配置规则，举例为：

```
<LifecycleConfiguration>
  <Rule>
    <ID>1125</ID>
    <Prefix>log_backup</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>2</Days>
    </Expiration>
  </Rule>
```

```
</LifecycleConfiguration>
```

getlifecycle

命令说明：

```
osscmd getlifecycle oss://bucket
```

获取该Bucket lifecycle规则。

使用示范：

- `python osscmd getlifecycle oss://mybucket`

deletelifecycle

命令说明：

```
osscmd deletelifecycle oss://bucket
```

删除该bucket下所有的lifecycle规则。

使用示范：

- `python osscmd deletelifecycle oss://mybucket`

putreferer

命令说明：

```
osscmd putreferer oss://bucket --allow_empty_referer=[true|false]
--referer=[referer]
```

设置防盗链规则。其中参数allow_empty_referer用来设置是否允许为空，为必选参数。参数referer来设置允许访问的白名单，比如“www.test1.com,www.test2.com”，以“,”作为分隔。详细的配置规则参考[产品文档](#)。

使用示范：

- `python osscmd putreferer oss://mybucket --allow_empty_referer=true --referer="www.test1.com,www.test2.com"`

getreferer

命令说明：

```
osscmd getreferer oss://bucket
```

获取该Bucket下防盗链设置规则。

使用示范：

- `python osscmd getreferer oss://mybucket`

putlogging

命令说明：

```
osscmd putlogging oss://source_bucket oss://target_bucket/[prefix]
```

其中source_bucket表示需要记录日志的bucket，而target_bucket则是用来存放产生的日志。同时可以对源bucket产生的日志文件设置前缀，方便用户归类查询。

使用示范：

- `python osscmd getlogging oss://mybucket`

getlogging

命令说明：

```
osscmd getlogging oss://bucket
```

获取该bucket的logging设置规则，返回为一个xml。

使用示范：

- `python osscmd getlogging oss://mybucket`

6.4 有关Object命令

ls(list)

命令说明：

```
ls(list) oss://bucket/[prefix] [marker] [delimiter] [maxkeys]
```

列出bucket中的object。

使用示范：

- `python osscmd ls oss://mybucket/folder1/folder2`
- `python osscmd ls oss://mybucket/folder1/folder2 marker1`
- `python osscmd ls oss://mybucket/folder1/folder2 marker1 /`
- `python osscmd ls oss://mybucket/`
- `python osscmd list oss://mybucket/ " " " 100`

命令说明：

```
ls(list) oss://bucket/[prefix] --marker=xxx --delimiter=xxx --maxkeys=xxx
--encoding_type=url
```

列出bucket中的object。其中encoding_type可以指定传输中使用的编码，当指定为url编码时，能够支持显示含控制字符的object。

使用示范：

- `python osscmd ls oss://mybucket/folder1/folder2 --delimiter=/`
- `python osscmd ls oss://mybucket/folder1/folder2 --marker=a`
- `python osscmd ls oss://mybucket/folder1/folder2 --maxkeys=10`

mkdir

命令说明：

```
mkdir oss://bucket/dirname
```

创建一个以“/”结尾的object，并且size为0。

使用示范：

- `python osscmd mkdir oss://mybucket/folder`

listallobject

命令说明：

```
listallobject oss://bucket/[prefix]
```

显示bucket下所有的object，可以指定prefix来显示。

使用示范：

- `python osscmd listallobject oss://mybucket`
- `python osscmd listallobject oss://mybucket/testfolder/`

deleteallobject

命令说明：

```
deleteallobject oss://bucket/[prefix]
```

删除bucket下所有的object，可以指定特定的prefix来删除。

使用示范：

- `python osscmd deleteallobject oss://mybucket`
- `python osscmd deleteallobject oss://mybucket/testfolder/`

downloadallobject

命令说明：

```
downloadallobject oss://bucket/[prefix] localdir --replace=false
--thread_num=5
```

将bucket下的object下载到本地目录，并且保持目录结构。可以指定prefix下载。—replace=false表示如果下载时，本地已经存在同名文件，不会覆盖。true则会覆盖。同时可以通过thread_num来配置下载线程。

使用示范：

- `python osscmd downloadallobject oss://mybucket /tmp/folder`
- `python osscmd downloadallobject oss://mybucket /tmp/folder --replace=false`
- `python osscmd downloadallobject oss://mybucket /tmp/folder --replace=true --thread_num=5`

downloadtodir

命令说明：

```
downloadtodir oss://bucket/[prefix] localdir --replace=false
```

将bucket下的object下载到本地目录，并且保持目录结构。可以指定prefix下载。—replace=false表示如果下载时，本地已经存在同名文件，不会覆盖。true则会覆盖。同downloadallobject 效果一样。

使用示范：

- `python osscmd downloadtodir oss://mybucket /tmp/folder`
- `python osscmd downloadtodir oss://mybucket /tmp/folder --replace=false`
- `python osscmd downloadtodir oss://mybucket /tmp/folder`


```
--replace=true
```

uploadfromdir

命令说明：

```
uploadfromdir localdir oss://bucket/[prefix] --check_point=check_poin  
t_file --replace=false  
--check_md5=false --thread_num=5
```

将本地目录里的文件上传到bucket中。例如localdir为/tmp/

里面有a/b, a/c, a三个文件，则上传到OSS中为oss://bucket/a/b, oss://bucket/a/c, oss://bucket/a。如果指定了prefix为mytest，则上传到OSS中为oss://bucket/mytest/a/b, oss://bucket/mytest/a/c, oss://bucket/mytest/a。

--check_point=check_point_file是指定文件。指定文件后，osscmd会将已经上传的本地文件以时间戳的方式放到check_point_file中，uploadfromdir命令会将正在上传的文件的时间戳和check_point_file记录的时间戳进行比较。如果有变化则会重新上传，否则跳过。默认情况下是没有check_point_file的。--replace=false表示如果下载时，本地已经存在同名文件，不会覆盖。true则会覆盖。--check_md5=false表示上传文件时，不会做携带Content-MD5请求头校验。true则会做校验。

注意：由于check_point_file文件中记录的是上传的所有文件的。
所以当上传文件特别多的时候，check_point_file会特别巨大。

使用示范：

- `python osscmd uploadfromdir /mytemp/folder oss://mybucket`
- `python osscmd uploadfromdir /mytemp/folder oss://mybucket
--check_point_file=/tmp/mytemp_record.txt`
- `python osscmd uploadfromdir C:\Documents and Settings\User\My
Documents\Downloads
oss://mybucket --check_point_file=C:\cp.txt`

put

命令说明：

```
put localfile oss://bucket/object --content-type=[content_type]
```

```
--headers="key1:value1#key2:value2" --check_md5=false
```

上传一个本地的文件到bucket中，可以指定object的content-type，或则指定自定义的headers。--check_md5=false表示上传文件时，不会做携带Content-MD5请求头校验。true则会做校验。

使用示范：

- `python osscmd put myfile.txt oss://mybucket`
- `python osscmd put myfile.txt oss://mybucket/myobject.txt`
- `python osscmd put myfile.txt oss://mybucket/test.txt --content-type=plain/text`
`--headers="x-oss-meta-des:test#x-oss-meta-location:CN"`
- `python osscmd put myfile.txt oss://mybucket/test.txt`
`--content-type=plain/text`

upload

命令说明：

```
upload localfile oss://bucket/object --content-type=[content_type]  
--check_md5=false
```

将本地文件以object group的形式上传。不推荐使用。--check_md5=false表示上传文件时，不会做携带Content-MD5请求头校验。true则会做校验。

使用示范：

- `python osscmd upload myfile.txt oss://mybucket/test.txt`
`--content-type=plain/text`

get

命令说明：

```
get oss://bucket/object localfile
```

将object下载到本地文件。

使用示范：

- `python osscmd get oss://mybucket/myobject /tmp/localfile`

multiget(multi_get)

命令说明：

```
multiget(multi_get) oss://bucket/object localfile --thread_num=5
```

将object以多线程的方式下载到本地文件。同时可以配置线程数。

使用示范：

- `python osscmd multiget oss://mybucket/myobject /tmp/localfile`
- `python osscmd multi_get oss://mybucket/myobject /tmp/localfile`

cat

命令说明：

```
cat oss://bucket/object
```

读取object的内容，直接打印出来。在object内容比较大的时候请不要使用。

使用示范：

- `python osscmd cat oss://mybucket/myobject`

meta

命令说明：

```
meta oss://bucket/object
```

读取object的meta信息，打印出来。meta信息包括content-type，文件长度，自定义meta等内容。

使用示范：

- `python osscmd meta oss://mybucket/myobject`

copy

命令说明：

```
copy oss://source_bucket/source_object oss://target_bucket/target_obj
ect
    --headers="key1:value1#key2:value2"
```

将源bucket的源object 复制到目的bucket中的目的object。

使用示范：

- `python osscmd copy oss://bucket1/object1 oss://bucket2/object2`

rm(delete,del)

命令说明：

```
rm(delete,del) oss://bucket/object --encoding_type=url
```

删除object。当指定encoding-type为url编码时，传入待删除的字串也需为url编码。

使用示范：

- `python osscmd rm oss://mybucket/myobject`
- `python osscmd delete oss://mybucket/myobject`
- `python osscmd del oss://mybucket/myobject`
- `python osscmd del oss://mybucket/my%01object --encoding_type=url`

signurl(sign)

命令说明：

```
signurl(sign) oss://bucket/object --timeout=[timeout_seconds]
```

生成一个包含签名的URL，并指定超时的时间。适用于bucket为私有时将特定的object提供他人访问。

使用示范：

- `python osscmd sign oss://mybucket/myobject`
- `python osscmd signurl oss://mybucket/myobject`

6.5 有关Multipart命令

init

命令说明：

```
init oss://bucket/object
```

初始化生成一个Upload ID。这个Upload ID可以配合后面的multiupload命令来使用。

使用示范：

- `python osscmd init oss://mybucket/myobject`

listpart

命令说明：

```
listpart oss://bucket/object --upload_id=xxx
```

显示指定object的Upload ID下已经上传的Parts。相关概念见OSS API文档。必须要指定Upload ID

。

使用示范：

- ```
python osscmd listpart oss://mybucket/myobject --upload_id=75835E389EA648C0B93571B6A46023F3
```

## listparts

命令说明：

```
listparts oss://bucket
```

显示bucket中未完成的multipart Upload ID和object。一般在删除bucket提示bucket非空的情况下可以用这个命令查看是否有multipart相关的内容。

使用示范：

- ```
python osscmd listparts oss://mybucket
```

getallpartsize

命令说明：

```
getallpartsize oss://bucket
```

显示bucket中还存在的Upload ID已经上传的Parts的总大小。

使用示范：

- ```
python osscmd getallpartsize oss://mybucket
```

## cancel

命令说明：

```
cancel oss://bucket/object --upload_id=xxx
```

终止Upload ID对应的Multipart Upload事件。

使用示范：

- ```
python osscmd cancel oss://mybucket/myobject --upload_id=D9D278DB6F8845E9AFE797DD235DC576
```

multiupload(multi_upload,mp)

命令说明：

```
multiupload(multi_upload,mp) localfile oss://bucket/object --check_md5=false
```

```
--thread_num=10
```

将本地文件以multipart的方式上传到OSS。

使用示范：

- `python osscmd multiupload /tmp/localfile.txt oss://mybucket/object`
- `python osscmd multiup_load /tmp/localfile.txt oss://mybucket/object`
- `python osscmd mp /tmp/localfile.txt oss://mybucket/object`

命令说明：

```
multiupload(multi_upload,mp) localfile oss://bucket/object --upload_id
=xxx --thread_num=10
    --max_part_num=1000 --check_md5=false
```

将本地文件以multipart的方式上传到OSS。本地文件划分的块数由max_part_num来指定。这个命令在实现的时候，会先去判断Upload ID对应的Parts的ETag是否和本地文件的MD5值是否相等，相等则跳过上传。所以如果在使用之前生成一个Upload ID，作为参数传进来。即使上传没有成功，重复执行相同的multiupload命令可以达到一个断点续传的效果。--check_md5=false表示上传文件时，不会做携带Content-MD5请求头校验。true则会做校验。

使用示范：

- `python osscmd multiupload /tmp/localfile.txt oss://mybucket/object --upload_id=D9D278DB6F8845E9AFE797DD235DC576`
- `python osscmd multiup_load /tmp/localfile.txt oss://mybucket/object --thread_num=5`
- `python osscmd mp /tmp/localfile.txt oss://mybucket/object --max_part_num=100`**copylargefile**

命令说明：

```
copylargefile oss://source_bucket/source_object oss://target_bucket/
target_object
    --part_size=10*1024*1024 --upload_id=xxx
```

对于超过1G的大文件进行复制时，采用multipart的方式将object复制到指定位置，（源bucket必须与目标bucket处于同一region）。其中upload_id为可选参数，如果当需要对某一次multipart copy事件进行续传的时候，可以传入该事件的upload_id。part_size用来设定分块大小，分块最小需要大于100KB，最多支持10000块分块。如果part_size设定值导致与OSS限制冲突，程序会帮你自动调节分块大小。

使用示范：

- ```
python osscmd copylargefile oss://source_bucket/source_object
oss://target_bucket/target_object --part_size=10*1024*1024
```

### uploadpartfromfile (upff)

命令说明：

```
uploadpartfromfile (upff) localfile oss://bucket/object --upload_id=
xxx
--part_number=xxx
```

主要用于测试，不推荐使用。

### uploadpartfromstring(upfs)

命令说明：

```
uploadpartfromstring(upfs) oss://bucket/object --upload_id=xxx --
part_number=xxx
--data=xxx
```

主要用于测试，不推荐使用。

## 7 ossprobe

---

### 简介

ossprobe是一款针对oss访问的检测工具，用于排查上传下载过程中，因网络故障或基本参数设置错误导致的问题。用户执行上传下载命令后，ossprobe会提示可能的错误原因，帮助用户快速找出错误。

### 版本

版本号：1.0.0

### 主要功能

- 检测网络环境是否正常
- 检查基本参数是否正常
- 测试基本上传下载速度

### 支持平台

- linux
- windows
- mac

### 软件下载

- [windows64 ossprobe](#)
- [linux64 ossprobe](#)
- [mac ossprobe](#)

### 检测下载问题

- 用法

```
ossprobe --download [-i AccessKeyId] [-k AccessKeySecret] [-p
EndPoint] [-b BucketName] [-o ObjectName] [-t LocalPath]
[-f Url] [-a Address]
-f --from Object的Url
-i --id AccessKeyId
-k --key AccessKeySecret
-p --endpoint EndPoint
-b --bucket BucketName
-o --object ObjectName
-t --to 保存下载内容的文件路径，默认为当前目录下的临时文件的
路径。
```



`-a` `--addr` 检测的网络地址，默认的地址为`www.aliyun.com`，如果您使用的是专有云，请一定要选择该网络内可以访问的地址。

- 示例

检测Url下载是否正常（[获取Url的方法](#)），可以使用下面的命令：

| 方式                   | 命令                                                         |
|----------------------|------------------------------------------------------------|
| 从指定Url下载             | <code>ossprobe --download -f Url</code>                    |
| 从指定Url下载到指定文件        | <code>ossprobe --download -f Url -t tmp/example.txt</code> |
| 从指定Url下载、并检测指定地址网络状况 | <code>ossprobe --download -f Url -a Addr</code>            |

检测指定参数(AccessKeyId, AccessKeySecret, EndPoint, BucketName)下载是否正常，可以通过以下命令检测：

| 方式                 | 命令                                                                                                                            |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------|
| 下载随机文件             | <code>ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName</code>                                  |
| 下载指定的文件            | <code>ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o ObjectName</code>                    |
| 下载指定的文件并保存到本地指定文件  | <code>ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o ObjectName -t tmp/example.txt</code> |
| 下载随机文件、并检测指定地址网络状况 | <code>ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -a Addr</code>                          |



说明：

- 用户下载的是二进制可执行程序，在linux上需要通过`chmod +x ossprobe`添加ossprobe的可执行权限。

- -t 的参数的默认值是当前目录下的一个临时文件的路径(文件名格式为：ossfilestore20160315060101)。
- 如果-t 的参数值为一个目录，那么就在该目录产生一个临时文件(文件名格式为：ossfilestore20160315060101)，用于保存数据。
- 采用Url 下载时，保存文件名取Url 中以“/”分割之后最后的一个字符串，比如说Url 为http://aliyun.com/a.jpg，文件名就是a.jpg。

## 检测上传问题


- 用法

```
ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b
BucketName [-m normal|append|multipart]
[-s UploadFilePath] [-o ObjectName] [-a Addr]
-i --id AccessKeyId
-k --key AccessKeySecret
-p --endpoint EndPoint
-b --bucket BucketName
-s --src 待上传文件路径，默认为本地临时文件的路径。
-m --mode 文件上传方式，默认为normal上传。
-o --object 上传后的object名称，当-s非空，默认值为上传文件名。当-s
为 空，默认值为以tem开头的临时文件的文件名。
-a --addr 检测的网络地址，默认为阿里云的官网地址，如果您使用的是
专有云，请一定要选择该网络内可以访问的地址。
```

- 示例

| 方式                      | 命令                                                                                                    |
|-------------------------|-------------------------------------------------------------------------------------------------------|
| 生成临时文件，并采用normal方式上传    | ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName                         |
| 生成临时文件，并采用append方式上传    | ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o ObjectName -m append |
| 生成临时文件，并采用multipart方式上传 | ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o                      |

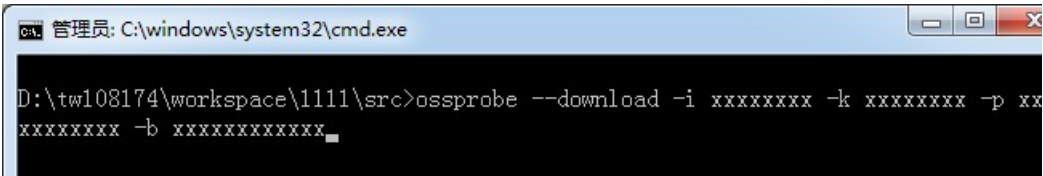
| 方式                              | 命令                                                                                                                                              |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
|                                 | <code>ObjectName -m<br/>multipart</code>                                                                                                        |
| 采用multipart上传方式，上传指定的内容         | <code>ossprobe --upload -i AccessKeyI<br/>d -k AccessKeySecret -p EndPoint<br/>-b BucketName -o<br/>ObjectName -m<br/>multipart -s src</code>   |
| 采用multipart上传指定的内容，并给出Object名称  | <code>ossprobe --upload -i AccessKeyI<br/>d -k AccessKeySecret -p EndPoint<br/>-b BucketName -m<br/>multipart -s src -<br/>o example.txt</code> |
| 生成临时文件，采用normal方式上传，并检测指定地址网络状况 | <code>ossprobe --upload -i AccessKeyI<br/>d -k AccessKeySecret -p EndPoint<br/>-b BucketName -a<br/>Addr</code>                                 |

说明：

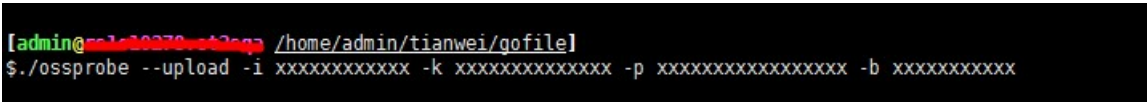
随机产生的文件名以ossuploadtmp开头。

平台差异

- windows按下Win + R调出运行对话框，输入命令cmd并按回车执行。在弹出的命令行终端界面中，输入该工具的所在的路径，然后填入相关检测参数后即可执行。



- linux&mac打开终端，在弹出的终端界面中，输入该工具的所在的路径，然后填入相关检测参数后即可执行。



## 查看报告结果

命令运行结束时，生成一个文件名为logOssProbe20060102150405.txt ( logOssProbe后面的数字为当前时间的格式化输出 ) 的报告文件。可能的错误原因会在命令行打印。如果错误提示不够具体，用户可以查看报告进行排查问题。如果还是无法解决，您也可以在工单里附上检测报告。

### 控制台显示

控制台显示的主要内容有：

- 执行步骤后出现×表示没有通过，否则表示通过。
- 结果显示整个上传下载成功还是失败。当成功时，会给出文件的大小和上传下载时间。
- 修改建议项提示导致错误的原因，或直接给出修改建议。
- 用户如果对oss错误码比较了解，也可以通过oss返回的详细的错误信息进行排查。
- 日志信息提示日志名称和日志的地址，方便用户查找具体的日志。



#### 说明：

并不是每次错误的检测都能提示出修改建议，对于没有提示修改建议的检测，请根据错误码提示，并结合[oss错误码ErrorCode](#)进行问题排查。

### 日志文件

日志文件不同于控制台显示主要是可以查看详细的网络检测过程，ping可以查看到指定的网络是否正常，可以查看到指定的EndPoint的网络是否正常，tracert可以查看到访问EndPoint的路由情况，最后一个nslookup可以查看DNS是否正常。

## 参考资料

- [oss错误码列表](#)
- [获取Url的方法](#)

## 8 OSS常用工具汇总

OSS除了控制台还有以下常用工具，可以帮助您更高效的使用OSS。

| 工具                         | 简介                                                                                                                          | 备注                                                                                                       |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <a href="#">ossbrowser</a> | 图形化的Object管理工具。支持Windows、Linux、Mac平台。                                                                                       | 官方工具。提供类似Windows资源管理器的功能。用户可以方便的浏览文件、上传下载文件、支持断点续传等。                                                     |
| <a href="#">ossutil</a>    | 命令行管理工具。提供方便、简洁、丰富的Object管理命令。                                                                                              | 官方工具，支持Windows, Linux, Mac平台，不依赖于任何第三方组件，下载后即用不需要安装。                                                     |
| <a href="#">osscommand</a> | 命令行管理工具。提供完备的Bucket、object管理命令。                                                                                             | 官方工具。基于Python2.5 - 2.7版本，支持多平台。将逐步被ossutil替代，除非需要ossutil不具备的Bucket管理功能外，强烈推荐使用 <a href="#">ossutil</a> 。 |
| <a href="#">ossfs</a>      | 挂载bucket到本地文件系统，能够通过本地文件系统操作OSS上的对象，实现数据的访问和共享。                                                                             | 官方工具。支持Linux平台。                                                                                          |
| <a href="#">ossftp</a>     | FTP工具，使用FTP协议来管理OSS的object，可以使用FileZilla、WinSCP、FlashFXP等FTP客户端操作OSS。<br>OSSFTP本质是FTP Server，接收FTP请求，将对文件、文件夹的操作映射为对OSS的操作。 | 官方工具。基于Python2.7及以上，支持Windows、Linux、Mac平台。                                                               |
| <a href="#">ossimport</a>  | 数据同步工具。可以将本地或第三方云存储服务上的文件同步到OSS上。                                                                                           | 官方工具。依赖JRE7及以上。支持Windows、Linux平台。                                                                        |
| <a href="#">可视化图片服务工具</a>  | 可以直观的看到OSS图片服务处理的结果，图片处理的调试不可或缺的工具。                                                                                         | 第三方工具。网页版。支持浏览器Chrome、Firefox、Safari。                                                                    |

| 工具                                | 简介                                                   | 备注                                    |
|-----------------------------------|------------------------------------------------------|---------------------------------------|
| <a href="#">可视化签名工具</a>           | 可视化签名工具。可以调试签名遇到的问题。当您实现OSS的签名中遇到问题，请跟该工具的签名对比，错误立现。 | 第三方工具。网页版。支持浏览器Chrome、Firefox、Safari。 |
| <a href="#">RAM Policy Editor</a> | OSS授权策略自动化生产工具。当您需要生成自己的授权策略时，强烈推荐使用该工具。             | 官方工具。网页版。支持浏览器Chrome、Firefox、Safari。  |
| <a href="#">ossprobe</a>          | 网络问题检查工具。常见错误给出建议。当您的环境访问OSS出错误时，推荐首先使用该工具检查。        | 官方工具。支持Windows、Linux、Mac。             |
| <a href="#">oss-emulator</a>      | 轻量级的OSS服务模拟器，用于基于OSS应用的调试、性能测试等。                     | 官方工具。支持Windows、Linux、Mac。             |

## 9 ossutil

---

### 9.1 下载和安装

ossutil工具旨在为用户提供一个方便的，以命令行方式管理OSS数据的途径。当前版本未提供完整的Bucket管理功能和Multipart管理功能，相关功能会在后续版本中开发。现在如果有使用上述功能的需要，可以先使用`osscmd`命令行工具。

#### 工具下载

- 当前版本

当前版本：1.4.1

- 运行环境

— Windows/Linux/Mac

— 支持架构

■ x86 (32bit, 64bit)

- binary下载

— [Linux x86 32bit] [ossutil32](#)

— [Linux x86 64bit] [ossutil64](#)

— [Windows x86 32bit] [ossutil32.zip](#)

— [Windows x86 64bit] [ossutil64.zip](#)

— [mac x86 64bit] [ossutilmac64](#)

- 安装使用

根据您的操作系统选择相应的binary或者压缩包下载后，运行相应的binary（如果binary为不可执行文件，请给binary增加可执行权限：`chmod 755 ossutil`），即：

linux系统下：

```
./ossutil
```

windows系统下有两种方法（以64位系统为例）：

1) 解压压缩包，双击运行其中的bat文件，再键入：`ossutil64.exe`

2) 解压压缩包，cmd进入压缩包中binary所在的目录，键入：`ossutil64.exe`

mac系统下：

```
./ossutilmac64
```

## 快速使用

- 设置ossutil的语言

在使用ossutil的命令时，可以使用-L选项设置语言，可选范围为CH/EN，即：中文或英文。大小写不敏感。默认语言为CH（中文），如果设置成CH（中文），需要确保您的系统为utf-8编码，否则可能会显示乱码。

如：

```
./ossutil help ls 显示ls默认语言的帮助
```

```
./ossutil help ls -L ch 显示ls的中文帮助
```

```
./ossutil help ls -L en 显示ls的英文帮助
```

```
./ossutil config -L ch 运行ossutil config的交互式配置命令，其中的提示语言为中文。
```

```
./ossutil config -L en 运行ossutil config的交互式配置命令，其中的提示语言为英文。
```



说明：

ossutil输出的错误默认都为英文，不会受上述选项影响。

- 获取命令列表

```
./ossutil或 ./ossutil help
```

```
$. /ossutil
用法: ossutil [command] [args...] [options...]
请使用ossutil help command来显示command命令的帮助
Commands:
 mb cloud_url [options]
 创建Bucket
 ls [cloud_url] [options]
 列举Buckets或者Objects
 rm cloud_url [options]
 删除Bucket或Objects
 stat cloud_url [options]
 显示bucket或者object的描述信息
 set-acl cloud_url [acl] [options]
 设置bucket或者objects的acl
 set-meta cloud_url [meta] [options]
 设置已上传的objects的元信息
 cp src_url dest_url [options]
 上传，下载或拷贝Objects
```



```

restore cloud_url [options]
 恢复冷冻状态的Objects为可读状态
create-symlink cloud_url target_url [options]
 创建符号链接
read-symlink cloud_url [options]
 读取符号链接文件的描述信息
Additional Commands:
help [command]
 获取命令的帮助文档
config [options]
 创建配置文件用以存储配置项
hash file_url [options]
 计算本地文件的crc64或md5
update [options]
 更新ossutil

$./ossutil -L en
Usage: ossutil [command] [args...] [options...]
Please use 'ossutil help command' to show help of command
Commands:
mb cloud_url [options]
 Make Bucket
ls [cloud_url] [options]
 List Buckets or Objects
rm cloud_url [options]
 Remove Bucket or Objects
stat cloud_url [options]
 Display meta information of bucket or objects
set-acl cloud_url [acl] [options]
 Set acl on bucket or objects
set-meta cloud_url [meta] [options]
 set metadata on already uploaded objects
cp src_url dest_url [options]
 Upload, Download or Copy Objects
restore cloud_url [options]
 Restore Frozen State Object to Read Ready Status
create-symlink cloud_url target_url [options]
 Create symlink of object
read-symlink cloud_url [options]
 Display meta information of symlink object
Additional Commands:
help [command]
 Get help about commands
config [options]
 Create configuration file to store credentials
hash file_url [options]
 Get crc64 or md5 of local file
update [options]
 Update ossutil

```

- 查看某命令的帮助文档

`./ossutil help cmd`强烈建议在使用某命令前先使用`help`来查阅帮助文档。

```

./ossutil help config -L ch
SYNOPSIS
 创建配置文件用以存储配置项
SYNTAX

```

```
ossutil config [-e endpoint] [-i id] [-k key] [-t token] [-L language] [--output-dir outdir] [-c file]
```

#### DETAIL DESCRIPTION

该命令创建配置文件，将用户设置的配置项信息存储进该配置文件，配置项用以访问OSS时提供访问信息（某命令是否需要配置项，参见其是否支持--config-file选项，具体可见该命令的帮助）。

配置文件路径可由用户指定，默认为/home/admin/.ossutilconfig。如果配置文件存在，假设其为:a，ossutil会将文件a另存为:a.bak，然后重新创建文件a并写入配置，此时，如果a.bak存在，其会被文件a覆盖。

注意：

（1）如果指定的配置文件路径非默认路径，在使用命令时请将--config-file选项设置为你配置时指定的配置文件路径（如果不指定--config-file选项，则运行命令时默认会读取/home/admin/.ossutilconfig）。

（2）某些配置可在使用命令时通过选项进行设置，如--endpoint，--access-key-id，等选项（具体请见每个命令的帮助），如果使用命令时指定了这些选项，并且同时配置文件中也配置了这些信息，则优先级为：选项 > 配置文件。

（3）如果使用命令时指定了--endpoint、--access-key-id、--access-key-secret

或--sts-token选项，则ossutil不强求配置文件一定要存在。

用法：

该命令有两种用法，交互式1)和非交互式2)，推荐用法为交互式，因为交互式用法拥有更好的安全性。

1) ossutil config [-c file]

该用法提供一种交互式的方法来配置信息，ossutil交互式地询问用户如下信息：

(1) config file

配置文件路径，如果用户键入回车，ossutil会使用默认的配置文件：/home/admin/.ossutilconfig。

如果用户自己指定了配置文件，在使用命令时需要将--config-file选项设置为用户设置的配置文件路径。哪些命令支持--config-file选项可由查看每个命令的帮助。

(2) language

当首次配置（配置文件不存在）时，ossutil会向用户询问语言设置，可选值为中文或者英文（CH/EN），如果键入回车，ossutil将根据用户输入的--language选项配置，如果此时用户也未输入--language选项，将配置成默认语言中文。

件

如果配置文件已存在，ossutil会综合用户输入的language选项和配置文

在

件中的语言信息，配置该项，而不会询问。

ossutil在运行时会从配置文件中读取该language选项，如果该选项不存

或者非法，将采用默认语言：CH。

注意：该配置项在此次config成功结束后才会生效，在执行config命令过程中语言显示不会受用户的选择影响。

(3) endpoint, accessKeyID, accessKeySecret

回车代表着跳过相应配置项的设置。注意：endpoint应该为一个二级域名(SLD)，例如：oss.aliyuncs.com。

以上选项一般为必选项。

(4) stsToken

如果用户需要使用临时token来访问oss，用户需要填入该项，否则请输入回车跳过该项配置。

(5) outputDir

该选项配置输出文件所在目录的路径。交互式模式时不提供该选项的配置，但配置文件中该项配置起效。

outputDir的默认目录为：当前目录下的：ossutil\_output，ossutil会在运行过程中将输出文件都生成到该文件夹下。输出文件目前包含：在cp命令中批量操作出错时，记录每个文件操作的错误信息的report文件。

关于outputDir和report文件的更多信息请参见cp命令的帮助。

注意：outputDir如果不存在，ossutil在产生输出文件时会自动创建该目录，如果outputDir存在且并非目录，将会报错。

下述交互式Bucket-Endpoint和Bucket-Cname配置被取消，但配置文件中该两项配置仍然起效。

(6) Bucket-Endpoint

Bucket-Endpoint对每个指定的bucket单独配置endpoint，此配置会优先于配置文件中关于默认endpoint的配置。

在该版本中，ossutil取消了交互式配置中，关于Bucket-Endpoint配对的配置，但配置文件中该项配置仍然起效，所以如果用户想对每个bucket单独指定endpoint，仍然可以在配置文件中配置。注意：此处的endpoint应该为一个二级域名(SLD)，例如：oss.aliyuncs.com。

如果配置了Bucket-Endpoint选项，当对某bucket进行操作时，ossutil会在该选项中寻找该bucket对应的endpoint，如果找到，该endpoint会覆盖基本配置中endpoint。但是运行命令时如果指定了--endpoint选项，--endpoint选项为最高优先级。

(7) Bucket-Cname

Bucket-Cname为每个指定的bucket单独配置CNAME域名(CDN加速域名)，此配置会优先于配置文件中Bucket-Endpoint及endpoint的配置。

在该版本中，ossutil取消了交互式配置中，关于Bucket-Cname配对的配置，但配置文件中该项配置仍然起效，所以如果用户想对每个bucket单独指定CNAME域名，仍然可以在配置文件中配置。

如果配置了Bucket-Cname选项，当对某bucket进行操作时，ossutil会在该选项中寻找该bucket对应的CNAME域名，如果找到，则找到的CNAME域名会覆盖Bucket-Endpoint选项和基本配置中的endpoint。运行命令时如果指定了--endpoint选项，--endpoint选项为最高优先级。

优先级：--endpoint > Bucket-Cname > Bucket-Endpoint > endpoint > 默认endpoint

2) ossutil config options

如果用户使用命令时输入了除--language和--config-file之外的任何选项，则该命令进入非交互式模式。所有的配置项应当使用选项指定。

配置文件格式：

[Credentials]

```

 language = CH
 endpoint = oss.aliyuncs.com
 accessKeyID = your_key_id
 accessKeySecret = your_key_secret
 stsToken = your_sts_token
 outputDir = your_output_dir
 [Bucket-Endpoint]
 bucket1 = endpoint1
 bucket2 = endpoint2
 ...
 [Bucket-CName]
 bucket1 = cname1
 bucket2 = cname2
 ...
SAMPLE
 ossutil config
 ossutil config -e oss-cn-hangzhou.aliyuncs.com -c ~/.myconfig
OPTIONS
 -c, --config-file
 ossutil工具的配置文件路径, ossutil启动时从配置文件读取配置, 在config
命令中, ossutil将配置写入该文件。
 -e, --endpoint
 ossutil工具的基本endpoint配置 (该选项值会覆盖配置文件中的相应设
置), 注意其必须为一个二级域名。
 -i, --access-key-id
 访问oss使用的AccessKeyID (该选项值会覆盖配置文件中的相应设置)。
 -k, --access-key-secret
 访问oss使用的AccessKeySecret (该选项值会覆盖配置文件中的相应设置)。
 -t, --sts-token
 访问oss使用的STSToken (该选项值会覆盖配置文件中的相应设置), 非必须设
置项。
 --output-dir=ossutil_output
 指定输出文件所在的目录, 输出文件目前包含: cp命令批量拷贝文件出错时所产生的
report文件 (关于report文件更多信息, 请参考cp命令帮助)。默认值为: 当前目录下的
ossutil_output目录。
 -L CH, --language=CH
 设置ossutil工具的语言, 默认值: CH, 取值范围: CH/EN, 若设置成"CH", 请
确保您的系统编码为UTF-8。

```

- 配置ossutil

在使用访问oss的命令时, 需要先配置访问AK, 关于AK的更多信息见: [RAM和STS介绍](#)。

配置ossutil由两种方式: 交互式和非交互式。

关于配置命令的更多帮助, 请使用 `ossutil help config` 查看。

#### — 交互式配置ossutil

```
./ossutil config
```

```
$./ossutil config -L ch
```

该命令创建将一个配置文件, 在其中存储配置信息。

请输入配置文件路径 ( 默认为 : /home/admin/.ossutilconfig , 回车将使用默认路径。如果用户设置为其它路径, 在使用命令时需要将--config-file选项设置为该路径 ) :

### — 非交互式配置ossutil

```
./ossutil config -e oss.aliyuncs.com -i your_id -k your_key
```

## 9.2 查看选项

通过-h选项来查看所有ossutil支持的所有option。

```

$./ossutil -h
Usage of ossutil:
Options:
 -s --short-format 显示精简格式, 如果未指定该选项, 默认显示长格式。
 --snapshot-path= 该选项用于在某些场景下加速增量上传批量文件 (目
前, 下载和拷贝不支持该选项)。在cp上传文件时使用该选项, ossutil在指定的目录下生
成文件记录文件上传的快照信息, 在下次指定该选项上传时, ossutil会读取指定目录下的
快照信息进行增量上传。用户指定的snapshot目录必须为本地文件系统上的可写目录, 若该
目录不存在, ossutil会创建该文件用于记录快照信息, 如果该目录已存在, ossutil会读
取里面的快照信息, 根据快照信息进行增量上传 (只上传上次未成功上传的文件和本地进
行过修改的文件) , 并更新快照信息。注意: 因为该选项通过在本地图录记录成功上传
的文件的本地 lastModifiedTime, 从而在下次上传时通过比较lastModifiedTime来决
定是否跳过相同文件的上传, 所以在使用该选项时, 请确保两次上传期间没有其他用户
更改了oss上的对应 object。当不满足该场景时, 如果想要增量上传批量文件, 请使
用--update选项。另外, ossutil不会主动删除snapshot-path下的快照信息, 为了 避
免快照信息过多, 当用户确定快照信息无用时, 请用户自行清理snapshot-path。
 -j --jobs= 多文件操作时的并发任务数, 默认值: 5, 取值范围: 1-
10000
 -v --version 显示ossutil的版本 (1.0.0.Beta2) 并退出。
 --output-dir= 指定输出文件所在的目录, 输出文件目前包含: cp命
令批量拷贝文件出错时所产生的report文件 (关于report文件更多信息, 请参考cp命令帮
助)。默认值为 : 当前目录下的ossutil_output目录。
 --parallel= 单文件内部操作的并发任务数, 取值范围: 1-10000
, 默认将由ossutil根据操作类型和文件大小自行决定。
 -L --language= 设置ossutil工具的语言, 默认值: CH, 取值范围: CH
/EN
 -t --sts-token= 访问oss使用的STSToken (该选项值会覆盖配置文件中的
相应设置) , 非必须设置项。
 -m --multipart 指定操作的对象为bucket中未完成的Multipart事
件, 而非默认情况下的object。
 -b --bucket 对bucket进行操作, 该选项用于确认操作作用于
bucket
 --delete 删除操作
 -e --endpoint= ossutil工具的基本endpoint配置 (该选项值会覆盖
配置文件中的相应设置) , 注意其必须为一个二级域名。
 -k --access-key-secret= 访问oss使用的AccessKeySecret (该选项值会覆盖配
置文件中的相应设置)。
 --bigfile-threshold= 开启大文件断点续传的文件大小阈值, 默认值: 100M
, 取值范围: 0B-9223372036854775807B

```

|     |                   |                                                                                                                              |
|-----|-------------------|------------------------------------------------------------------------------------------------------------------------------|
| 500 | --retry-times=    | 当错误发生时的重试次数，默认值：3，取值范围：1-                                                                                                    |
| -a  | --all-type        | 指定操作的对象为bucket中的object和未完成的Multipart事件。                                                                                      |
| -r  | --recursive       | 递归进行操作。对于支持该选项的命令，当指定该选项时，命令会对bucket下所有符合条件的objects进行操作，否则只对url中指定的单个object进行操作。                                             |
| -f  | --force           | 强制操作，不进行询问提示。                                                                                                                |
| -u  | --update          | 更新操作                                                                                                                         |
| -c  | --config-file=    | ossutil工具的配置文件路径，ossutil启动时从配置文件读取配置，在config命令中，ossutil将配置写入该文件。                                                             |
| -i  | --access-key-id=  | 访问oss使用的AccessKeyID ( 该选项值会覆盖配置文件中的相应设置 )。                                                                                   |
|     | --acl=            | acl信息的配置。                                                                                                                    |
| -d  | --directory       | 返回当前目录下的文件和子目录，而非递归显示所有子目录下的所有object。                                                                                        |
|     | --checkpoint-dir= | checkpoint目录的路径(默认值为:.ossutil_checkpoint)，断点续传时，操作失败ossutil会自动创建该目录，并在该目录下记录checkpoint信息，操作成功会删除该目录。如果指定了该选项，请确保所指定的目录可以被删除。 |
| md5 | --type=           | 计算的类型，默认值：crc64，取值范围：crc64/md5                                                                                               |
| -h  | --help            | Show usage message                                                                                                           |

ossutil的每个命令支持以上option中的部分option，若要查看某个命令command支持哪些option，请使用ossutil help command来查看。

## 9.3 有关bucket的命令

### Bucket的相关命令

ossutil提供了创建、删除、列举Bucket、以及为Bucket设置acl的功能，关于Bucket更多的管理功能暂时不支持，如需要请使用osscli。

在使用这些命令前，请先使用config命令配置访问AK。

- 创建Bucket

```
ossutil mb oss://bucket [--acl=acl] [--storage-class sc] [-c file]
```

其中acl如果不指定，则默认为private权限，如果成功创建，ossutil会打印消耗时间并退出，否则会输出错误信息。可以通过--storage-class选项指定存储方式。

关于创建Bucket的帮助信息，请使用ossutil help mb命令查看。

```
$.ossutil mb oss://test
0.220478(s) elapsed
```

- 删除Bucket

关于删除Bucket的帮助信息，请使用ossutil help rm命令查看。注意：

- 删除bucket必须设置-b选项；
- 被删除的bucket可能被其他用户重新创建，从而不再属于您；
- bucket中的数据一旦被删除则无法恢复。

( 1 ) 如果您的Bucket中没有数据

```
ossutil rm oss://bucket -b
```

```
$. /ossutil rm oss://test -b
Do you really mean to remove the Bucket: test(y or N)? y
0.220478(s) elapsed
```

( 2 ) 如果您的Bucket中有object或multipart等数据，需要先删除所有数据再删除Bucket，可以使用以下命令来一并删除所有数据和您的Bucket

```
ossutil rm oss://bucket -bar
```

关于删除Bucket的帮助信息，请使用ossutil help rm命令查看。

- 列举Buckets

```
./ossutil ls或 ./ossutil ls oss://
```

可以使用-s选项来显示精简格式，更多帮助见: ossutil help ls

```
$. /ossutil ls
CreationTime Region StorageClass
BucketName
2016-10-21 16:18:37 +0800 CST oss-cn-hangzhou Archive
oss://go-sdk-test-bucket-xyz-for-object
2016-12-01 15:06:21 +0800 CST oss-cn-hangzhou Standard
oss://ossutil-test
2016-07-18 17:54:49 +0800 CST oss-cn-hangzhou Standard
oss://ossutilconfig
2016-07-20 10:36:24 +0800 CST oss-cn-hangzhou IA
oss://ossutilupdate
2016-11-14 13:08:36 +0800 CST oss-cn-hangzhou IA
oss://yyyyy
2016-08-25 09:06:10 +0800 CST oss-cn-hangzhou Archive
oss://ztzt
2016-11-21 21:18:39 +0800 CST oss-cn-hangzhou Archive
oss://ztztzt
Bucket Number is: 7
0.252174(s) elapsed
```

- 列举Bucket中的文件

ossutil可以列举Bucket中的Object和UploadID，默认情况下显示Object，使用-m选项来显示UploadID，使用-a选项同时显示Object和UploadID。

## — 列举Object

```
./ossutil ls oss://bucket
```

```
$./ossutil ls oss://ossutil-test
LastModifiedTime Size(B) StorageClass ETAG
ObjectName
2016-12-01 15:06:37 +0800 CST 10363812 Standard
61DE142E5AFF9A6748707D4A77BFBCFB oss://ossutil-test/a1
2016-12-01 15:06:42 +0800 CST 10363812 Standard
61DE142E5AFF9A6748707D4A77BFBCFB oss://ossutil-test/a2
2016-12-01 15:06:45 +0800 CST 10363812 Standard
61DE142E5AFF9A6748707D4A77BFBCFB oss://ossutil-test/a3
Object Number is: 3
0.007379(s) elapsed
```

## — 列举Object和Multipart

```
./ossutil ls oss://bucket -a
```

```
$ ossutil ls oss://bucket1 -a
LastModifiedTime Size(B) StorageClass ETAG
ObjectName
2015-06-05 14:06:29 +0000 CST 201933 Standard
7E2F4A7F1AC9D2F0996E8332D5EA5B41 oss://bucket1/dir1/obj11
2015-06-05 14:36:21 +0000 CST 201933 Standard
6185CA2E8EB8510A61B3A845EAFE4174 oss://bucket1/obj1
2016-04-08 14:50:47 +0000 CST 6476984 Standard
4F16FDAE7AC404CEC8B727FCC67779D6 oss://bucket1/sample.txt
Object Number is: 3
InitiatedTime UploadID
ObjectName
2017-01-13 03:45:26 +0000 CST 15754AF7980C4DFB8193F190837520BB
oss://bucket1/obj1
2017-01-13 03:43:13 +0000 CST 2A1F9B4A95E341BD9285CC42BB950EE0
oss://bucket1/obj1
2017-01-13 03:45:25 +0000 CST 3998971ACAF94AD9AC48EAC1988BE863
oss://bucket1/obj2
2017-01-20 11:16:21 +0800 CST A20157A7B2FEC4670626DAE0F4C0073C
oss://bucket1/tobj
UploadId Number is: 4
0.191289(s) elapsed
```

可以使用-s选项显示精简模式。可以使用-d选项显示首层目录下的内容。

```
$ ossutil ls oss://bucket1 -d
oss://bucket1/obj1
oss://bucket1/sample.txt
oss://bucket1/dir1/
Object and Directory Number is: 3
UploadID ObjectName
15754AF7980C4DFB8193F190837520BB oss://bucket1/obj1
2A1F9B4A95E341BD9285CC42BB950EE0 oss://bucket1/obj1
3998971ACAF94AD9AC48EAC1988BE863 oss://bucket1/obj2
A20157A7B2FEC4670626DAE0F4C0073C oss://bucket1/tobj
UploadId Number is: 4
```



```
0.119884(s) elapsed
```

- 为Bucket设置acl

创建Bucket时，Bucket默认的acl为private，可以通过set-acl命令来修改Bucket的acl。在设置Bucket的acl权限时，需要设置-b选项。

将bucket1设置为private权限：

```
./ossutil set-acl oss://bucket1 private -b
```

关于设置acl的更多信息请使用help set set-acl来查看。

## 9.4 有关object的命令

ossutil提供了上传/下载/拷贝文件、设置object的acl、设置object的meta、查看object的meta信息等功能。

使用这些命令前请使用config命令配置访问AK。

- 上传/下载/拷贝文件

强烈建议在使用cp命令前使用ossutil help cp先查看帮助。

可以使用cp命令进行上传/下载/拷贝文件，使用-r选项来拷贝文件夹，对大文件默认使用分片上传并可进行断点续传（开启分片上传的大文件阈值可用--bigfile-threshold选项来设置）。

使用-f选项来默认强制上传，当目标端存在同名文件时，不询问，直接覆盖。

当批量上传/下载/拷贝文件时，如果某个文件出错，ossutil默认会将错误信息记录在report文件，并跳过该文件，继续其他文件的操作（当错误为Bucket不存在、accessKeyID/accessKeySecret错误造成的权限验证非法等错误时，不再继续其他文件拷贝）。更多信息请见ossutil help cp。

ossutil支持特定场景下的增量上传策略：--update和--snapshot-path选项，请参见ossutil help cp。

ossutil从1.0.0.Beta1版本开始，上传文件默认打开crc64。

（1）上传单个文件：

```
$. /ossutil cp a oss://ossutil-test
Succeed: Total num: 1, size: 230. OK num: 1(upload 1 files).
```

```
0.699795(s) elapsed
```

## (2) 上传文件夹：

```
$. /ossutil cp -r dir oss://ossutil-test
Succeed: Total num: 35, size: 464,606. OK num: 35(upload 34 files, 1
directories).
0.896320(s) elapsed
```

### — 上传/下载/拷贝文件的性能调优

在cp命令中，通过jobs项和-parallel项控制并发数。-jobs项控制多个文件上传/下载/拷贝时，文件间启动的并发数。-parallel制分片上传/下载/拷贝一个大文件时，每一个大文件启动的并发数。

默认情况下，ossutil会根据文件大小来计算parallel个数（该选项对于小文件不起作用，进行分片上传/下载/拷贝的大文件文件阈值可由-bigfile-threshold选项来控制），当进行批量大文件的上传/下载/拷贝时，实际的并发数为jobs个数乘以parallel个数。该两个选项可由用户调整，当ossutil自行设置的默认并发达不到用户的性能需求时，用户可以自行调整该两个选项来升降性能。

注意：

如果并发数调得太大，由于线程间资源切换及抢夺等，ossutil上传/下载/拷贝性能可能会下降，所以请根据实际的机器情况调整这两个选项的数值，如果要进行压测，可以一开始将两个数值调低，慢慢调大寻找最优值。

如果--jobs选项和--parallel选项值太大，在机器资源有限的情况下，可能会因为网络传输太慢，产生EOF错误，这个时候请适当降低--jobs选项和--parallel选项值。

- 设置object的acl

ossutil使用set-acl命令设置object的acl，使用-r选项可以批量设置object的acl。

更多信息见：ossutil help set-acl

```
$. /ossutil set-acl oss://dest/a private
0.074507(s) elapsed
```

批量设置acl：

```
$. /ossutil set-acl oss://dest/a private -r
Do you really mean to recursively set acl on objects of oss://dest/a
(y or N)? y
Succeed: Total 3 objects. Setted acl on 3 objects.
0.963934(s) elapsed
```

- 设置object的meta

ossutil使用set-meta命令设置object的meta信息，使用-r选项可以批量设置object的meta。

更多信息见：ossutil help set-meta

```
./ossutil set-meta oss://dest/a x-oss-object-acl:private -u
```

- 查看object描述信息 ( meta )

ossutil使用stat命令查看object的描述 ( meta ) 信息。

更多信息见：ossutil help stat

```
./ossutil stat oss://dest/a
ACL : default
Accept-Ranges : bytes
Content-Length : 230
Content-Md5 : +5vbQC/MSQK0xXSiyKBZog==
Content-Type : application/octet-stream
Etag : FB9BDB402FCC4902B4C574A2C8A059A2
Last-Modified : 2017-01-13 15:14:22 +0800 CST
Owner : aliyun
X-Oss-Hash-Crc64ecma : 12488808046134286088
X-Oss-Object-Type : Normal
0.125417(s) elapsed
```

- 恢复冷冻状态的object为可读状态

ossutil使用restore命令恢复冷冻状态的object为可读状态。可以使用-r选项批量恢复冷冻状态的objects为可读状态。

更多信息见：ossutil help restore

```
./ossutil restore oss://utiltest/a
0.037729(s) elapsed
```

- 创建符号链接

ossutil使用create-symlink创建符号链接。

更多信息见：ossutil help create-symlink

```
./ossutil create-symlink oss://utiltest/b a
0.037729(s) elapsed
```

- 读取符号链接文件的描述信息

ossutil使用read-symlink读取符号链接文件的描述信息。

更多信息见：ossutil help read-symlink

```
./ossutil read-symlink oss://utiltest/b
Etag : D7257B62AA6A26D66686391037B7D61A
Last-Modified : 2017-04-26 15:34:27 +0800 CST
```

```
X-Oss-Symlink-Target : a
0.112494(s) elapsed
```

## 9.5 有关multipart的命令

### Multipart的相关命令

ossutil提供了列举UploadID、删除指定object的所有UploadID的功能。

关于Multipart的更多介绍请参见[断点续传](#)。



说明：

ossutil在上传/拷贝文件时，对于大文件自动进行分片断点续传，而不提供UploadPart命令。

- 列举UploadID

使用-m选项来列举指定object下的所有未完成UploadID，使用-a选项来列举object和UploadID。

```
$ ossutil ls oss://bucket1/obj1 -m
InitiatedTime UploadID
ObjectName
2017-01-13 03:45:26 +0000 CST 15754AF7980C4DFB8193F190837520BB
oss://bucket1/obj1
2017-01-13 03:43:13 +0000 CST 2A1F9B4A95E341BD9285CC42BB950EE0
oss://bucket1/obj1
UploadId Number is: 2
0.070070(s) elapsed
```

- 删除指定object的所有UploadID

使用-m选项删除指定object下的所有未完成UploadID，当同时指定-r选项时，会删除以指定object为前缀的所有object的未完成UploadID。

假设bucket1下有如下文件：

```
$ ossutil ls oss://bucket1 -a
LastModifiedTime Size(B) StorageClass ETAG
 ObjectName
2015-06-05 14:06:29 +0000 CST 201933 Standard
7E2F4A7F1AC9D2F0996E8332D5EA5B41 oss://bucket1/dir1/obj11
2015-06-05 14:36:21 +0000 CST 241561 Standard
6185CA2E8EB8510A61B3A845EAFE4174 oss://bucket1/obj1
2016-04-08 14:50:47 +0000 CST 6476984 Standard
4F16FDAE7AC404CEC8B727FCC67779D6 oss://bucket1/sample.txt
Object Number is: 3
InitiatedTime UploadID
ObjectName
2017-01-13 03:45:26 +0000 CST 15754AF7980C4DFB8193F190837520BB
oss://bucket1/obj1
2017-01-13 03:43:13 +0000 CST 2A1F9B4A95E341BD9285CC42BB950EE0
oss://bucket1/obj1
2017-01-13 03:45:25 +0000 CST 3998971ACAF94AD9AC48EAC1988BE863
oss://bucket1/obj2
```

```
2017-01-20 11:16:21 +0800 CST A20157A7B2FEC4670626DAE0F4C0073C
 oss://bucket1/tobj
UploadId Number is: 4
0.191289(s) elapsed
```

删除obj1的2个UploadID :

```
$. /ossutil rm -m oss://bucket1/obj1
Succeed: Total 2 uploadIds. Removed 2 uploadIds.
1.922915(s) elapsed
```

删除obj1和obj2的3个UploadID :

```
$. /ossutil rm -m oss://bucket1/obj1
Succeed: Total 4 uploadIds. Removed 4 uploadIds.
1.922915(s) elapsed
```

同时删除obj1 , obj1和obj2的3个UploadID :

```
$. /ossutil rm oss://dest1/.a -a -r -f
Do you really mean to remove recursively objects and multipart
uploadIds of oss://dest1/.a(y or N)? y
Succeed: Total 1 objects, 3 uploadIds. Removed 1 objects, 3
uploadIds.
```