zosutil

用户使用指南

天翼云科技有限公司

目录

1 简介

zosutil 是一款用于管理天翼云对象存储服务的命令行工具，支持访问天翼云 S3 接口的对象存储服务（ZOS），应用于 windows/ctyunos/centos/ubuntu 平台。相比于 ZOS SDK ，zosutil 允许用户使用命令行的方式与 ZOS 进行交互，用户使用起来更加简洁、方便。

2 用户指南

2.1 账户管理

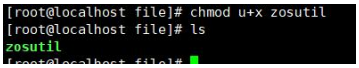
2.1.1 添加账户

（1）以下使用以 Centos7 系统上的 zosutil 为例，首先需要将 Centos 版本的 zosutil 文件上传到相应的服务器，后直接执行该文件即可（注：如果出现提示权限不够，则需要添加权限可执行权限）。

① 在天翼云官网下载 zosutil 的压缩包，进行解压。

② 进入 centos 目录下，在天翼云服务器上使用 rz 命令将 centos 目录下的 zosutil 二进制文件上传至您的工作目录下。

③ 执行： chmod u+x zosutil 给 zosutil 二进制文件添加可执行权限

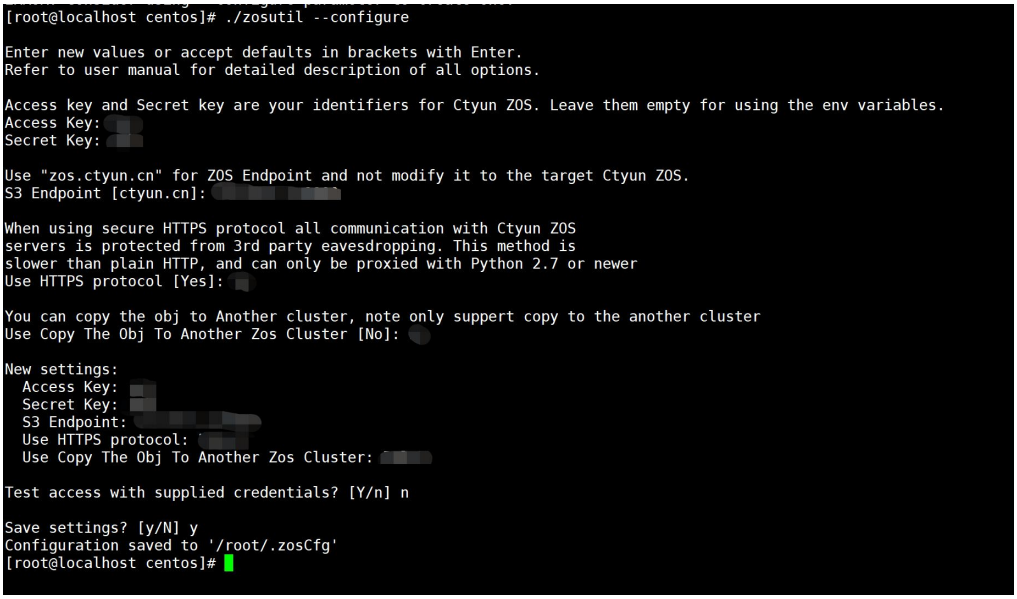


在首次使用时，首先需要添加用户相关信息，直接执行 zosutil 则会出现如下提示。

```
[root@localhost centos]# ./zosutil
ERROR: /root/.zosCfg: Connection refused
ERROR: Configuration file not available.
ERROR: Consider using --configure parameter to create one.
[root@localhost centos]#
```

执行命令 zosutil --configure 进行账户信息配置，根据相应的提示输入 Access Key、Secret Key、Endpoint、是否使用 https，是否使用远端集群拷贝功能等信息，在信息输入完毕后，根据提示对输入凭据进行测试和保存，在 Centos 环境中，

凭据默认保存在~/.zosCfg 文件中。



相关输入解释：

Access Key：对象存储的 access key，可通过控制台获取。

Secret Key：对象存储的 secret key，可通过控制台获取。

S3 Endpoint：对象存储的访问 IP 或域名。

Use HTTPS protocol：是否使用 https 协议。

Use Copy The Obj To Another Zos Cluster：是否需要拷贝对象到另一个 ZOS 集群，如果选择 Yes，则需要根据提示填写另一个 ZOS 集群的 AK、SK、Endpoint、Use HTTPS Protocol 信息。

2.1.2 账户信息修改和删除

如果需要对已经输入的账户信息进行修改，则重新执行命令 zosutil --configure，再次进行账户信息填写。如果需要删除账户信息，则直接删除相应的.zosCfg 文件即可。

```
[root@localhost centos]# ./zosutil --configure

Enter new values or accept defaults in brackets with Enter.
Refer to user manual for detailed description of all options.

Access key and Secret key are your identifiers for Ctyun ZOS. Leave them empty for using the env variables.
Access Key [ ]:
Secret Key [ ]:

Use "zos.ctyun.cn" for ZOS Endpoint and not modify it to the target Ctyun ZOS.
S3 Endpoint [ ]:

When using secure HTTPS protocol all communication with Ctyun ZOS
servers is protected from 3rd party eavesdropping. This method is
slower than plain HTTP, and can only be proxied with Python 2.7 or newer
Use HTTPS protocol [ ]:

You can copy the obj to Another cluster, note only suppert copy to the another cluster
Use Copy The Obj To Another Zos Cluster [ ]:

New settings:
  Access Key:
  Secret Key:
  S3 Endpoint:
  Use HTTPS protocol:
  Use Copy The Obj To Another Zos Cluster:

Test access with supplied credentials? [Y/n] y
Please wait, attempting to list all buckets...
Success. Your access key and secret key worked fine :-)

Now verifying that encryption works...
Not configured. Never mind.

Save settings? [y/N] y
Configuration saved to '/root/.zosCfg'
[root@localhost centos]#
```

2.2 桶管理

2.2.1 桶的展示

（1）命令：./zosutil ls
功能：展示当前用户下所有的桶
示例：

```
root@localhost centos]# ./zosutil ls
2024-05-30 07:48  zos://test
```

（2）命令：./zosutil info zos://test
功能：展示 test 桶的详细信息
示例：

```
[root@localhost centos]# ./zosutil info zos://test
zos://test/ (bucket):
   Versioning:none
   Policy:    none
   CORS:      none
   ACL:       zyh: FULL_CONTROL
```

（3）命令：./zosutil du zos://test
功能：展示当前桶内存储的文件数量和总文件大小
示例：查询 test 桶内存储的文件数量和总文件大小

```
[root@localhost centos]# ./zosutil du zos://test
      102400     100 objects zos://test/
```

2.2.2 创建桶

命令：./zosutil mb zos://bucket
功能：创建一个名为 bucket 的桶
示例：

```
[root@localhost centos]# ./zosutil mb zos://bucket
Bucket 'zos://bucket/' created
```

注：（1）默认创建的桶 ACL 为 private（私有）。

（2）通过添加 ACL 相关参数，创建不同权限的桶，具体为：--acl-private（私有）、--acl-public（公共读），--acl-public-read-write（公共读写） 。

示例：创建一个公共读的桶：

```
[root@localhost centos]# ./zosutil mb zos://public --acl-public
Bucket 'zos://public/' created
```

2.2.3 删除桶

命令：./zosutil rb zos://bucket
功能：删除 bucket 桶
示例：

```
[root@localhost centos]# ./zosutil rb zos://bucket
Bucket 'zos://bucket/' removed
```

注：删除桶之前需要清空桶内对象，不能删除非空的桶。

2.2.4 设置桶 policy

命令：./zosutil setpolicy FILE zos://bucket
功能：设置 bucket 桶 policy，具体 policy 规则写入 FILE 文件中
文件内容示例：
{"Version": "2012-10-17", "Statement": [{ "Sid": "id-1","Effect": "Allow", "Action": ["s3:*"] }]}

各字段描述如下：

字段	描述	是否必须
Version	保持与 Amazon S3 一致，当前支持"2012-10-17"	否
Statement	桶策略描述，定义完整的权限控制。每条桶策略的 Statement 可由多条描述组成，每条描述是一个 dict，每条描述可包含以下	是

	字段： Sid Effect Principal Action ReSource Condition	
Sid	本条桶策略描述的 ID	否
Effect	桶策略的效果，即指定本条桶策略描述的权限是接受请求还是拒绝请求。 接受请求：配置为“Allow”， 拒绝请求：配置为“Deny”	是
Principal	被授权人，即指定本条桶策略描述所作用的用户，支持通配符“*”，表示所有用户。当对某个 user 进行授权时，Principal 格式为"AWS": "arn:aws:s3:::user/userId"	否
Action	操作，即指定本条桶策略描述所作用的 ZOS 操作。以列表形式表示，可配置多条操作，以逗号间隔。支持通配符”*“，表示该资源能进行的所有操作。常用的 Action 有"s3:GetObject"，"s3:GetObjectAcl"，"s3:PutObject"，"s3:PutObjectAcl"等	否
Condition	条件语句，指定本条桶策略所限制的条件。可以通过 Condition 对 ZOS 资源设置防盗链，形如： "Condition": {"StringEquals":{“aws:Referer":["www.example.com"]}，此时如果 Effect 为“Allow”，则允许来自"www.example.com"的请求；如果为“Deny”，则拒绝。	否

示例：

```
[root@localhost centos]# ./zosutil setpolicy FILE zos://bucket
[root@localhost centos]# ./zosutil info zos://test
zos://test/ (bucket):
   Versioning:none
   Policy:    {"Version": "2012-10-17", "Statement": [{ "Sid": "id-1","Effect": "Allow", "Action": [ "s3:*"] } ]}
   CORS:      none
   ACL:       zyh: FULL_CONTROL
[root@localhost centos]#
```

2.2.5 删除桶 policy

命令：./zosutil delpolicy zos://bucket
功能：删除 bucket 桶的 policy
示例：

```
[root@localhost centos]# ./zosutil delpolicy zos://bucket
zos://bucket/: Policy deleted
```

2.2.6 设置桶生命周期规则

命令：zosutil setlifecycle FILE zos://bucket
功能：为 bucket 桶设置生命周期规则
文件内容示例：

```
<?xml version="1.0" ?>
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Rule>
        <ID>Lifecycle Test</ID>
        <Filter>
            <Prefix>prefix</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Expiration>
            <Days>10</Days>
        </Expiration>
    </Rule>
    <Rule>
        <ID>id-test-023</ID>
        <Filter>
            <Tag>
                <Key>obj1</Key>
                <Value>value1</Value>
            </Tag>
        </Filter>
        <Status>Enabled</Status>
        <NoncurrentVersionTransition>
            <NoncurrentDays>1</NoncurrentDays>
            <StorageClass>GLACIER</StorageClass>
        </NoncurrentVersionTransition>
    </Rule>
</LifecycleConfiguration>
```

各字段描述：

字段	描述	是否必须
LifecycleConfiguration	生命周期规则的容器	是

Rule	生命周期规则	是
Expiration	用日期或天数指定对象的过期时间	否
Date	标识对象的过期日期,日期为 ISO8601 格式，必须为 UTC 午夜 0 时	Date 与 Days 二选一
Days	标识对象受规则约束的天数	Date 与 Days 二选一
ID	标识唯一的规则	是
Filter	过滤应用规则的对象	否
Prefix	标识应用规则的对象前缀	是
Tag	应用规则到拥有指定标签的对象	否
Key	标签的名称	否
Value	标签的值	否
Status	标识是否应用规则 ， 可选值：Enabled，Disabled	是
Transtions	标识对象何时转存到指定的 Storage Class	否
StorageClass	标识要转存储到哪种存储类别,如 STANDARD/INFREQUENT-ACCESS/ARCHIVE(ARCHIVE 级别暂未启用)	如设置转存储规则，该字段必选
NoncurrentVersionTransitions	标识历史版本的转存储规则	否
NoncurrentDays	标识对象的历史版本受规则约束的天数	否
NoncurrentVersionExpiration	标识历史版本的过期规则	否

示例：

```
[root@localhost centos]# ./zosutil setlifecycle FILE zos://bucket
zos://bucket/: Lifecycle Policy updated
[root@localhost centos]#
```

2.2.7 获取桶生命周期规则

命令：./zosutil getlifecycle zos://bucket
功能：获取 bucket 桶的生命周期规则
示例：

```
[root@localhost centos]# ./zosutil getlifecycle zos://bucket
<?xml version="1.0" ?>
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
        <Rule>
                <ID>Lifecycle Test</ID>
                <Filter>
                        <Prefix>prefix</Prefix>
                </Filter>
                <Status>Enabled</Status>
                <Expiration>
                        <Days>10</Days>
                </Expiration>
        </Rule>
        <Rule>
                <ID>id-test-023</ID>
                <Filter>
                        <Tag>
                                <Key>obj1</Key>
                                <Value>value1</Value>
                        </Tag>
                </Filter>
                <Status>Enabled</Status>
                <NoncurrentVersionTransition>
                        <NoncurrentDays>1</NoncurrentDays>
                        <StorageClass>GLACIER</StorageClass>
                </NoncurrentVersionTransition>
        </Rule>
</LifecycleConfiguration>
```

2.2.8 删除桶生命周期规则

命令：./zosutil dellifecycle zos://bucket
功能：删除桶 bucket 的生命周期规则
示例：

```
[root@localhost centos]# ./zosutil dellifecycle zos://bucket
zos://bucket/: Lifecycle Policy deleted
```

2.2.9 设置桶跨域规则

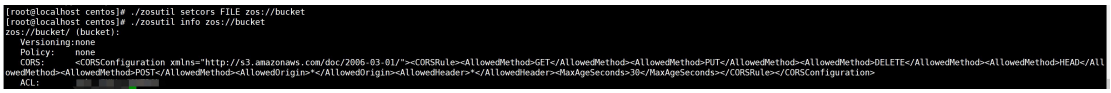
命令：./zosutil setcors FILE zos://bucket
功能：为 bucket 桶设置跨域规则
文件内容示例：

<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/"><CORSRule><AllowedMethod>GET</AllowedMethod><AllowedMethod>PUT</AllowedMethod><AllowedMethod>DELETE</AllowedMethod><AllowedMethod>HEAD</AllowedMethod><AllowedMethod>POST</AllowedMethod><AllowedOrigin>*</AllowedOrigin><AllowedHeader>*</AllowedHeader><MaxAgeSeconds>30</MaxAgeSeconds></CORSRule></CORSConfiguration>

各字段描述：

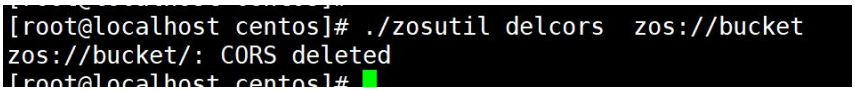
参数名称	参数描述	是否必须
CORSConfiguration	描述存储桶中对象的跨源访问配置。	是
CORSRule	为指定 bucket 配置的所有跨域规则的集合，允许配置 100 条规则	是
ID	跨域规则的 ID,最大长度 255	否
AllowedHeaders	允许浏览器发送 CORS 请求时携带的自定义 HTTP 请求头部，不区分英文大小写，单条 CORSRule 可以配置多个 AllowedHeader。	否
AllowedMethods	允许该源执行的 HTTP 方法列表，包括 GET , PUT , HEAD , POST , DELETE。单挑规则可以配置多个方法。	是
AllowedOrigins	允许能够访问该 bucket 的一个或多个源,支持 * 通配符，表示所有域名都允许访问，不推荐。 一条 CORSRule 可以配置多个 allowedorigins	是
ExposeHeaders	允许浏览器获取的 CORS 请求响应中的头部，不区分英文大小写,单条 CORSRule 可以配置多个 ExposeHeader。	否
MaxAgeSeconds	跨域资源共享配置的有效时间，单位为秒，对应 CORS 请求响应中的 Access-Control-Max-Age 头部，单条 CORSRule 只能配置一个 MaxAgeSeconds	否

示例：



2.2.10 删除桶跨域规则

命令：./zosutil delcors zos://bucket
功能：删除 bucket 桶的跨域规则
示例：



2.2.11 开启/暂停桶多版本

命令：zosutil setversioning zos://bucket enable|disable
功能：开启（enable）/暂停（disable）多版本
示例：

2.3 对象管理

2.3.1 对象的展示

（1）列举桶内对象

命令：./zosutil ls zos://bucket

功能：列举 bucket 桶内的对象（如果桶内存在目录，则会列举目录）

示例：

```
[root@localhost centos]# ./zosutil ls zos://bucket
                           DIR  zos://bucket/obj/
2024-05-31 05:54          1024  zos://bucket/object0
2024-05-31 05:54          1024  zos://bucket/object1
2024-05-31 05:54          1024  zos://bucket/object2
2024-05-31 05:54          1024  zos://bucket/object3
[root@localhost centos]#
```

（2）获取对象的具体信息

命令：./zosutil info zos://bucket/object0

功能：获取 bucket 桶内 object0 的详细信息

示例：

```
[root@localhost centos]# ./zosutil info zos://bucket/object0
zos://bucket/object0 (object):
   File size: 1024
   Last mod:  Fri, 31 May 2024 05:54:13 GMT
   MIME type: application/octet-stream
   Storage:   STANDARD
   MD5 sum:   bde587b69cf9ea422fe7d2776fcdebab
   SSE:       none
   Policy:    none
   CORS:      none
   ACL:
   x-amz-meta-zosutil-attrs: atime:1717057961/ctime:1717057937/gid:0/gname:root/md5:bde587b69cf9ea422fe7d2776fcdebab/mode:33188/mtime:1717057937/uid:0/uname:root
```

2.3.2 上传对象

（1）上传小对象

命令：./zosutil put obj1 zos://bucket

功能：向 bucket 桶内上传一个对象名为 obj1 的对象

示例：

```
[root@localhost centos]# ./zosutil put obj1 zos://bucket
upload: 'obj1' -> 'zos://bucket/obj1'  [1 of 1]
 1024 of 1024   100% in    0s    26.37 KB/s  done
```

（2）上传小对象，指定存储级别为低频，访问权限为公共读

命令：./zosutil put obj zos://bucket --acl-public

功能：向 bucket 桶内上传一个对象名为 obj 的公共读对象（注：（1）storage-class 和 acl 可分开指定 （2）存储级别支持：STANDARD（标准）/STANDARD_IA（低频）/GLACIER （归档）(3)访问控制权限支持：--acl-private（私有）/--acl-public（公共读）

示例：

```
[root@localhost centos]#
[root@localhost centos]# zosutil put obj zos://bucket --acl-public
upload: 'obj' -> 'zos://bucket/obj'  [1 of 1]
 4096 of 4096   100% in    0s   292.48 KB/s  done
Public URL of the object is: http://127.0.0.1:8000/bucket/obj
[root@localhost centos]#
```

（3）上传大对象

说明：在默认设置下，超过 15MB 的对象将自动进行分段上传，默认分段大小为 15MB，可通过--multipart-chunk-size-mb=SIZE 自行设置段大小。

命令：./zosutil put 1G zos://bucket --multipart-chunk-size-mb=128

功能：上传 1G 大小对象到 bucket 桶内，设置分段大小为 128MB

示例：

```
[root@localhost centos]# ./zosutil put 1G zos://bucket --multipart-chunk-size-mb=1
upload: '1G' -> 'zos://bucket/1G'  [part 1 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    3s    41.20 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 2 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    6s    18.52 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 3 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    6s    20.01 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 4 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    5s    22.23 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 5 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    1s    91.55 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 6 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    1s   106.66 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 7 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    1s   127.54 MB/s  done
upload: '1G' -> 'zos://bucket/1G'  [part 8 of 8, 128MB] [1 of 1]
 134217728 of 134217728   100% in    1s    95.17 MB/s  done
```

2.3.3 批量上传对象

（1）命令：./zosutil put obj* zos://bucket

功能：上传当前目录下以 obj 为前缀的对象到 bucket 桶内（注：该功能在 windows 环境需启用 gitbash）

示例：

```
[root@localhost centos]# ls
dir  FILE  obj0  obj1  obj2  obj3  test0  test1  test2  test3  zosutil
[root@localhost centos]# ./zosutil put obj* zos://bucket
upload: 'obj0' -> 'zos://bucket/obj0'  [1 of 4]
 1024 of 1024   100% in    0s    25.42 KB/s  done
upload: 'obj1' -> 'zos://bucket/obj1'  [2 of 4]
 1024 of 1024   100% in    0s    24.26 KB/s  done
upload: 'obj2' -> 'zos://bucket/obj2'  [3 of 4]
 1024 of 1024   100% in    0s    23.42 KB/s  done
upload: 'obj3' -> 'zos://bucket/obj3'  [4 of 4]
 1024 of 1024   100% in    0s    25.31 KB/s  done
```

（2）命令：./zosutil put obj* zos://bucket --concurrent-threads=4

功能：上传当前目录下以 obj 为前缀的对象到 bucket 桶内，并开启多线程并发上传，并发线程数为 4（注： --concurrent-threads 支持自定义并发线程数，需根据运行主机配置自行设置线程数，建议使用并发数不超过 10）

示例：

```
[root@localhost centos]# zosutil put obj* zos://bucket --concurrent-threads=4
upload: 'obj0' -> 'zos://bucket/obj0' (1024 bytes in 0.0 seconds, 44.30 KB/s) 1
upload: 'obj11' -> 'zos://bucket/obj11' (1024 bytes in 0.0 seconds, 53.36 KB/s) 2
upload: 'obj10' -> 'zos://bucket/obj10' (1024 bytes in 0.0 seconds, 66.11 KB/s) 3
upload: 'obj1' -> 'zos://bucket/obj1' (1024 bytes in 0.0 seconds, 43.46 KB/s) 4
upload: 'obj15' -> 'zos://bucket/obj15' (1024 bytes in 0.0 seconds, 62.99 KB/s) 5
upload: 'obj13' -> 'zos://bucket/obj13' (1024 bytes in 0.0 seconds, 56.22 KB/s) 6
upload: 'obj14' -> 'zos://bucket/obj14' (1024 bytes in 0.0 seconds, 51.71 KB/s) 7
upload: 'obj12' -> 'zos://bucket/obj12' (1024 bytes in 0.0 seconds, 47.46 KB/s) 8
upload: 'obj17' -> 'zos://bucket/obj17' (1024 bytes in 0.0 seconds, 58.25 KB/s) 9
upload: 'obj18' -> 'zos://bucket/obj18' (1024 bytes in 0.0 seconds, 57.90 KB/s) 10
upload: 'obj16' -> 'zos://bucket/obj16' (1024 bytes in 0.0 seconds, 49.82 KB/s) 11
upload: 'obj19' -> 'zos://bucket/obj19' (1024 bytes in 0.0 seconds, 55.13 KB/s) 12
upload: 'obj21' -> 'zos://bucket/obj21' (1024 bytes in 0.0 seconds, 65.16 KB/s) 13
upload: 'obj22' -> 'zos://bucket/obj22' (1024 bytes in 0.0 seconds, 66.68 KB/s) 14
upload: 'obj20' -> 'zos://bucket/obj20' (1024 bytes in 0.0 seconds, 62.50 KB/s) 15
upload: 'obj2' -> 'zos://bucket/obj2' (1024 bytes in 0.0 seconds, 51.69 KB/s) 16
upload: 'obj25' -> 'zos://bucket/obj25' (1024 bytes in 0.0 seconds, 55.55 KB/s) 17
upload: 'obj23' -> 'zos://bucket/obj23' (1024 bytes in 0.0 seconds, 52.46 KB/s) 18
upload: 'obj24' -> 'zos://bucket/obj24' (1024 bytes in 0.0 seconds, 52.41 KB/s) 19
upload: 'obj26' -> 'zos://bucket/obj26' (1024 bytes in 0.0 seconds, 49.83 KB/s) 20
upload: 'obj27' -> 'zos://bucket/obj27' (1024 bytes in 0.0 seconds, 55.33 KB/s) 21
upload: 'obj28' -> 'zos://bucket/obj28' (1024 bytes in 0.0 seconds, 55.88 KB/s) 22
upload: 'obj3' -> 'zos://bucket/obj3' (1024 bytes in 0.0 seconds, 61.95 KB/s) 23
upload: 'obj29' -> 'zos://bucket/obj29' (1024 bytes in 0.0 seconds, 55.43 KB/s) 24
upload: 'obj33' -> 'zos://bucket/obj33' (1024 bytes in 0.0 seconds, 69.55 KB/s) 25
upload: 'obj30' -> 'zos://bucket/obj30' (1024 bytes in 0.0 seconds, 62.62 KB/s) 26
upload: 'obj32' -> 'zos://bucket/obj32' (1024 bytes in 0.0 seconds, 57.65 KB/s) 27
upload: 'obj31' -> 'zos://bucket/obj31' (1024 bytes in 0.0 seconds, 60.55 KB/s) 28
upload: 'obj36' -> 'zos://bucket/obj36' (1024 bytes in 0.0 seconds, 60.13 KB/s) 29
upload: 'obj34' -> 'zos://bucket/obj34' (1024 bytes in 0.0 seconds, 52.52 KB/s) 30
upload: 'obj37' -> 'zos://bucket/obj37' (1024 bytes in 0.0 seconds, 59.74 KB/s) 31
upload: 'obj35' -> 'zos://bucket/obj35' (1024 bytes in 0.0 seconds, 46.93 KB/s) 32
upload: 'obj40' -> 'zos://bucket/obj40' (1024 bytes in 0.0 seconds, 54.71 KB/s) 33
upload: 'obj39' -> 'zos://bucket/obj39' (1024 bytes in 0.0 seconds, 50.53 KB/s) 34
upload: 'obj38' -> 'zos://bucket/obj38' (1024 bytes in 0.0 seconds, 45.07 KB/s) 35
upload: 'obj4' -> 'zos://bucket/obj4' (1024 bytes in 0.0 seconds, 51.59 KB/s) 36
upload: 'obj44' -> 'zos://bucket/obj44' (1024 bytes in 0.0 seconds, 50.61 KB/s) 37
upload: 'obj43' -> 'zos://bucket/obj43' (1024 bytes in 0.0 seconds, 49.38 KB/s) 38
upload: 'obj42' -> 'zos://bucket/obj42' (1024 bytes in 0.0 seconds, 47.28 KB/s) 39
upload: 'obj41' -> 'zos://bucket/obj41' (1024 bytes in 0.0 seconds, 43.79 KB/s) 40
upload: 'obj46' -> 'zos://bucket/obj46' (1024 bytes in 0.0 seconds, 64.01 KB/s) 41
upload: 'obj45' -> 'zos://bucket/obj45' (1024 bytes in 0.0 seconds, 58.71 KB/s) 42
upload: 'obj47' -> 'zos://bucket/obj47' (1024 bytes in 0.0 seconds, 64.12 KB/s) 43
upload: 'obj48' -> 'zos://bucket/obj48' (1024 bytes in 0.0 seconds, 56.26 KB/s) 44
upload: 'obj49' -> 'zos://bucket/obj49' (1024 bytes in 0.0 seconds, 54.15 KB/s) 45
upload: 'obj6' -> 'zos://bucket/obj6' (1024 bytes in 0.0 seconds, 56.89 KB/s) 46
upload: 'obj7' -> 'zos://bucket/obj7' (1024 bytes in 0.0 seconds, 54.07 KB/s) 47
upload: 'obj5' -> 'zos://bucket/obj5' (1024 bytes in 0.0 seconds, 45.90 KB/s) 48
upload: 'obj8' -> 'zos://bucket/obj8' (1024 bytes in 0.0 seconds, 64.11 KB/s) 49
upload: 'obj9' -> 'zos://bucket/obj9' (1024 bytes in 0.0 seconds, 65.86 KB/s) 50
```

2.3.4 上传文件夹

命令：./zosutil put dir zos://bucket -r
功能：上传 dir 文件夹到 bucket 桶内
示例：

```
[root@localhost centos]# ls
dir  FILE  obj0  obj1  obj2  obj3  test0  test1  test2  test3  zosutil
[root@localhost centos]# ./zosutil put dir zos://bucket -r
upload: 'dir/obj0' -> 'zos://bucket/dir/obj0'  [1 of 11]
 1024 of 1024   100% in    0s    25.84 KB/s  done
upload: 'dir/obj1' -> 'zos://bucket/dir/obj1'  [2 of 11]
 1024 of 1024   100% in    0s    26.92 KB/s  done
upload: 'dir/obj10' -> 'zos://bucket/dir/obj10'  [3 of 11]
 1024 of 1024   100% in    0s    24.59 KB/s  done
upload: 'dir/obj2' -> 'zos://bucket/dir/obj2'  [4 of 11]
 1024 of 1024   100% in    0s    27.23 KB/s  done
upload: 'dir/obj3' -> 'zos://bucket/dir/obj3'  [5 of 11]
 1024 of 1024   100% in    0s    21.32 KB/s  done
upload: 'dir/obj4' -> 'zos://bucket/dir/obj4'  [6 of 11]
 1024 of 1024   100% in    0s    22.65 KB/s  done
upload: 'dir/obj5' -> 'zos://bucket/dir/obj5'  [7 of 11]
 1024 of 1024   100% in    0s    21.92 KB/s  done
upload: 'dir/obj6' -> 'zos://bucket/dir/obj6'  [8 of 11]
 1024 of 1024   100% in    0s    17.13 KB/s  done
upload: 'dir/obj7' -> 'zos://bucket/dir/obj7'  [9 of 11]
 1024 of 1024   100% in    0s    18.59 KB/s  done
upload: 'dir/obj8' -> 'zos://bucket/dir/obj8'  [10 of 11]
 1024 of 1024   100% in    0s    22.08 KB/s  done
upload: 'dir/obj9' -> 'zos://bucket/dir/obj9'  [11 of 11]
 1024 of 1024   100% in    0s    23.30 KB/s  done
```

2.3.5 下载对象

命令：./zosutil get zos://bucket/obj1 object
功能：下载 bucke 桶内名为 obj1 的对象
示例：

```
[root@localhost centos]# ./zosutil get zos://bucket/obj1 object
download: 'zos://bucket/obj1' -> 'object'  [1 of 1]
 1024 of 1024   100% in    0s    19.84 KB/s  done
```

2.3.6 批量下载对象

命令：./zosutil get zos://bucket/dir download -r
功能：下载 bucket 桶内 dir 目录到 download 目录下
示例：

```
[root@localhost centos]# ./zosutil put dir zos://bucket -r
upload: 'dir/obj4' -> 'zos://bucket/dir/obj4'  [1 of 6]
 1024 of 1024   100% in    0s    14.28 KB/s  done
upload: 'dir/obj5' -> 'zos://bucket/dir/obj5'  [2 of 6]
 1024 of 1024   100% in    0s    14.13 KB/s  done
upload: 'dir/obj6' -> 'zos://bucket/dir/obj6'  [3 of 6]
 1024 of 1024   100% in    0s    15.69 KB/s  done
upload: 'dir/obj7' -> 'zos://bucket/dir/obj7'  [4 of 6]
 1024 of 1024   100% in    0s    16.80 KB/s  done
upload: 'dir/obj8' -> 'zos://bucket/dir/obj8'  [5 of 6]
 1024 of 1024   100% in    0s    13.91 KB/s  done
upload: 'dir/obj9' -> 'zos://bucket/dir/obj9'  [6 of 6]
 1024 of 1024   100% in    0s    13.00 KB/s  done
```

2.3.7 删除对象

命令：./zosutil rm zos://bucket/obj0
功能：删除 bucket 桶内 obj0 的对象
示例：

```
[root@localhost centos]# ./zosutil rm zos://bucket/obj0
delete: 'zos://bucket/obj0'
```

2.3.8 批量删除对象

命令：./zosutil rm zos://bucket/dir -r
功能：删除 bucket 桶内 dir 目录下的对象
示例：

```
[root@localhost centos]# ./zosutil rm zos://bucket/dir -r
delete: 'zos://bucket/dir/obj4'
delete: 'zos://bucket/dir/obj5'
delete: 'zos://bucket/dir/obj6'
delete: 'zos://bucket/dir/obj7'
delete: 'zos://bucket/dir/obj8'
delete: 'zos://bucket/dir/obj9'
```

2.3.9 移动对象

命令：./zosutil mv zos://bucket/obj1 zos://test/
功能：将对象 obj1 从 bucket 桶内移动到 test 桶
示例：

```
[root@localhost centos]# ./zosutil mv zos://bucket/obj1 zos://test/
move: 'zos://bucket/obj1' -> 'zos://test/obj1'  [1 of 1]
```

2.3.10 批量移动对象

命令：./zosutil mv zos://bucket/obj* zos://test/
功能：将 bucket 桶内以 obj 为前缀的对象移动到 test 桶内
示例：

```
[root@localhost centos]# ./zosutil mv zos://bucket/obj* zos://test/
move: 'zos://bucket/obj2' -> 'zos://test/obj2'  [1 of 2]
move: 'zos://bucket/obj3' -> 'zos://test/obj3'  [2 of 2]
```

2.3.11 拷贝对象（本集群）

命令：./zosutil cp zos://bucket/object0 zos://test
功能：拷贝 bucket 桶下 object0 对象到 test 桶
示例：

```
[root@localhost centos]# ./zosutil cp zos://bucket/object0 zos://test
remote copy: 'zos://bucket/object0' -> 'zos://test/object0'  [1 of 1]
```

2.3.12 批量拷贝对象（本集群）

命令：./zosutil cp zos://bucket/obj* zos://test
功能：拷贝 bucket 桶下 obj 为前缀的对象到 test 桶
示例：

```
[root@localhost centos]# ./zosutil cp zos://bucket/obj* zos://test
remote copy: 'zos://bucket/object0' -> 'zos://test/object0'  [1 of 11]
remote copy: 'zos://bucket/object1' -> 'zos://test/object1'  [2 of 11]
remote copy: 'zos://bucket/object10' -> 'zos://test/object10'  [3 of 11]
remote copy: 'zos://bucket/object2' -> 'zos://test/object2'  [4 of 11]
remote copy: 'zos://bucket/object3' -> 'zos://test/object3'  [5 of 11]
remote copy: 'zos://bucket/object4' -> 'zos://test/object4'  [6 of 11]
remote copy: 'zos://bucket/object5' -> 'zos://test/object5'  [7 of 11]
remote copy: 'zos://bucket/object6' -> 'zos://test/object6'  [8 of 11]
remote copy: 'zos://bucket/object7' -> 'zos://test/object7'  [9 of 11]
remote copy: 'zos://bucket/object8' -> 'zos://test/object8'  [10 of 11]
remote copy: 'zos://bucket/object9' -> 'zos://test/object9'  [11 of 11]
```

2.3.13 拷贝对象（跨集群）

命令：./zosutil cp zos://bucket/object0 zos://bucket-4d15 --crr
功能：拷贝本集群 bucket 桶内的对象 object0 到远端集群 bucket-4d15 桶
示例：

```
[root@localhost centos]# ./zosutil cp zos://bucket/object0 zos://bucket-4d15 --crr
remote copy: 'zos://bucket/object0' -> 'zos://bucket-4d15/object0'  [1 of 1]
[root@localhost centos]#
```

注：如果要使用跨集群拷贝功能，需要先配置远端集群的 AK、SK、Endpoint 等参数。参数配置方式：使用--configure 配置集群，将 Use Copy The Obj To Another Zos Cluster 参数输入 yes，进入远端集群的参数配置流程，配置相应参数。

2.3.14 批量拷贝对象（跨集群）

命令：./zosutil cp zos://bucket/obj* zos://bucket-4d15 --crr
功能：拷贝本集群 bucket 桶内前缀为 obj 的对象到远端集群 bucket-4d15 桶
示例：

```
[root@localhost centos]# ./zosutil cp zos://bucket/obj* zos://bucket-4d15 --crr
remote copy: 'zos://bucket/object0' -> 'zos://bucket-4d15/object0'  [1 of 11]
remote copy: 'zos://bucket/object1' -> 'zos://bucket-4d15/object1'  [2 of 11]
remote copy: 'zos://bucket/object10' -> 'zos://bucket-4d15/object10'  [3 of 11]
remote copy: 'zos://bucket/object2' -> 'zos://bucket-4d15/object2'  [4 of 11]
remote copy: 'zos://bucket/object3' -> 'zos://bucket-4d15/object3'  [5 of 11]
remote copy: 'zos://bucket/object4' -> 'zos://bucket-4d15/object4'  [6 of 11]
remote copy: 'zos://bucket/object5' -> 'zos://bucket-4d15/object5'  [7 of 11]
remote copy: 'zos://bucket/object6' -> 'zos://bucket-4d15/object6'  [8 of 11]
remote copy: 'zos://bucket/object7' -> 'zos://bucket-4d15/object7'  [9 of 11]
remote copy: 'zos://bucket/object8' -> 'zos://bucket-4d15/object8'  [10 of 11]
remote copy: 'zos://bucket/object9' -> 'zos://bucket-4d15/object9'  [11 of 11]
```

2.3.15 获取对象预签名下载链接

命令：./zosutil signurl zos://bucket/object0
功能：生成对象预签名下载链接
参数说明：

参数	描述	是否必须
--signature-v4	生成 v4 版本预签名下载链接，不加默认是生成 v2 版本	否
--expiry-days=EXPIRY_DAYS	预签名过期天数，默认为 1（v4 签名只能生成 7 天内的下载链接）	否
--expiry-date=EXPIRY_DATE	预签名过期时间，需要填写时间戳（例如 1704038400）	否

示例：

```
[root@localhost centos]# ./zosutil signurl zos://bucket/object0
http://127.0.0.1:8000/bucket/object0?[redacted]
[root@localhost centos]#
```

2.3.16 修改对象访问权限

命令：./zosutil setacl zos://bucket/object0 --acl-public
功能：修改 bucket 桶内 object0 的访问权限为公共读
示例：

```
[root@localhost centos]# ./zosutil setacl zos://bucket/object0 --acl-public
zos://bucket/object0: ACL set to Public  [1 of 1]
```