

# 数据加速器 GooseFS

## GooseFS-Lite



腾讯云

## 【 版权声明 】

©2013–2024 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

## 【 商标声明 】



及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

## 【 服务声明 】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

## 【 联系我们 】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或 95716。

---

## 文档目录

GooseFS-Lite  
GooseFS-Lite 工具

# GooseFS-Lite

## GooseFS-Lite 工具

最近更新时间：2024-04-10 15:18:21

### 功能说明

GooseFS-Lite 工具支持将对象存储（Cloud Object Storage, COS）存储桶挂载到本地，像使用本地文件系统一样直接操作腾讯云对象存储中的对象，相比于 COSFS 工具，GooseFS-Lite 可提供更高的大文件读写速度，不受本地磁盘的性能限制。GooseFS-Lite 支持 POSIX 文件系统的主要功能，例如文件顺序/随机读，顺序写、目录操作等功能。

### 局限性

GooseFS-Lite 仅适合挂载后对文件进行简单的管理，不支持本地文件系统的一些功能用法。请注意以下不适用的场景：

- 不支持对文件进行随机写和 truncate 操作。
- 多个客户端挂载同一个 COS 存储桶时，依赖用户自行协调多个客户端的行为。例如避免多个客户端写同一个文件等。
- 文件/文件夹的 rename 操作非原子操作。
- 不支持读取和 rename 当前挂载点正在写入的文件。
- 元数据操作。例如 list directory，性能较差，因为需要远程访问 COS 服务器。
- 不支持 soft/hard link。
- 追加写性能较差，涉及服务端数据拷贝和下载被追加文件。

#### ⚠ 注意

外网挂载和非低频存储的追加写操作，会产生下载流量费用。

### 使用环境

- JDK 11。
- Linux 系统：libfuse 2.9.3及以上。

### 使用方法

#### 步骤1：安装依赖

centos 可以执行如下命令，安装依赖。

```
yum install -y fuse-devel java-11-openjdk-devel
```

如果不希望修改默认 Java 环境，可以参考此文档[手动安装JDK](#)。

## 步骤2：下载 GooseFS-Lite 安装包

目前最新版本为1.0.4，请按照以下步骤安装使用。

1. 执行如下命令，获取 GooseFS-Lite 安装包：

```
curl -LO https://downloads.tencentgoosefs.cn/goosefs-lite/goosefs-  
lite-{version}.tar.gz  
  
## 执行时可以将版本号替换为公开版本号，如1.0.4，示例如下  
curl -LO https://downloads.tencentgoosefs.cn/goosefs-lite/goosefs-lite-1.0.4.tar.gz
```

2. 执行如下命令，获取 GooseFS-Lite 的 Md5 文件：

```
curl -LO https://downloads.tencentgoosefs.cn/goosefs-lite/goosefs-lite-{version}-  
md5.txt  
  
## 执行时可以将版本号替换为公开版本号，如1.0.4，示例如下  
curl -LO https://downloads.tencentgoosefs.cn/goosefs-lite/goosefs-lite-1.0.4-  
md5.txt
```

3. 执行如下命令，验证文件的完整性。

```
md5sum --check goosefs-lite-{version}-md5.txt  
  
## 执行时可以将版本号替换为公开版本号，如1.0.4，示例如下  
md5sum --check goosefs-lite-1.0.4-md5.txt
```

执行以上三步后，当看到如下图所示，则表示文件完整性验证通过。

```
[root@VM-0-11-tencentos ~]# curl -LO https://downloads.tencentgoosefs.cn/goosefs-lite/goosefs-lite-1.0.4.tar.gz  
% Total % Received % Xferd Average Speed Time Time Time Current  
Dload Upload Total Spent Left Speed  
100 118M 100 118M 0 0 12.1M 0 0:00:09 0:00:09 --:--:-- 11.6M  
[root@VM-0-11-tencentos ~]# curl -LO https://downloads.tencentgoosefs.cn/goosefs-lite/goosefs-lite-1.0.4-md5.txt  
% Total % Received % Xferd Average Speed Time Time Time Current  
Dload Upload Total Spent Left Speed  
100 66 100 66 0 0 71 0 --:--:-- --:--:-- --:--:-- 71  
[root@VM-0-11-tencentos ~]# md5sum --check goosefs-lite-1.0.4-md5.txt  
goosefs-lite-1.0.4.tar.gz: OK
```

## 步骤3：解压 GooseFS-Lite 安装包

```
tar -xvf goosefs-lite-{version}.tar.gz
```

## 执行时可以将版本号替换为公开版本号，如1.0.4，示例如下

```
tar -xvf goosefs-lite-1.0.4.tar.gz
```

您可以将上述命令中的 `{version}` 替换为使用的 GooseFS-Lite 版本，例如 1.0.4。当看到如下图所示，则表示解压成功，并生成了一个 `goosefs-lite-{version}` 的目录。

```
[root@VM-0-11-tencentos ~]# tar -xvf goosefs-lite-1.0.4.tar.gz
goosefs-lite-1.0.4/
goosefs-lite-1.0.4/bin/
goosefs-lite-1.0.4/target/
goosefs-lite-1.0.4/lib/
goosefs-lite-1.0.4/conf/
goosefs-lite-1.0.4/conf/goosefs-lite.properties
goosefs-lite-1.0.4/conf/goosefs-site.properties
goosefs-lite-1.0.4/conf/core-site.xml
goosefs-lite-1.0.4/conf/core-site.xml.template
goosefs-lite-1.0.4/conf/log4j.properties.template
goosefs-lite-1.0.4/conf/log4j.properties
goosefs-lite-1.0.4/lib/fuse-tools-install.sh
goosefs-lite-1.0.4/lib/fuse-tools.tar.gz
goosefs-lite-1.0.4/target/goosefs-lite-1.0.4-jar-with-dependencies.jar
goosefs-lite-1.0.4/bin/create-client-conf.py
goosefs-lite-1.0.4/bin/init.sh
goosefs-lite-1.0.4/bin/._goosefs-lite-keepalive.sh
tar: Ignoring unknown extended header keyword 'LIBARCHIVE.xattr.com.apple.quarantine'
goosefs-lite-1.0.4/bin/goosefs-lite-keepalive.sh
goosefs-lite-1.0.4/bin/enable-goosefs-lite-service.sh
goosefs-lite-1.0.4/bin/goosefs-lite
goosefs-lite-1.0.4/bin/readme.md
```

## 步骤4：初始化

将 `goosefs-lite` 软链接(ln)到 `/usr/bin/goosefs-lite`，方便后续可以直接使用 `goosefs-lite` 命令，而不需要进入到 `goosefs-lite` 目录。使用 `sudo` 或者 `root` 用户执行如下命令。

```
sudo bash bin/init.sh
```

```
~/test/goosefs-lite-1.0.3 » sudo bash bin/init.sh
ln -sf /data/home/xaopan/test/goosefs-lite-1.0.3/bin/goosefs-lite /usr/bin/goosefs-lite
fuse-2.9.7/
fuse-2.9.7/doc/
fuse-2.9.7/doc/ulockmgr_server.1
fuse-2.9.7/doc/kernel.txt
fuse-2.9.7/doc/fusermount.1
fuse-2.9.7/doc/Makefile.in
fuse-2.9.7/doc/html/
fuse-2.9.7/doc/html/fuse_8h.html
fuse-2.9.7/doc/html/fuse__lowlevel_8h.html
fuse-2.9.7/doc/html/files.html
fuse-2.9.7/doc/html/globals_enum.html
fuse-2.9.7/doc/html/nav_g.png
fuse-2.9.7/doc/html/structfuse__lowlevel__ops.html
fuse-2.9.7/doc/html/structfuse__session__ops.html
fuse-2.9.7/doc/html/tab_h.png
fuse-2.9.7/doc/html/globals_func.html
fuse-2.9.7/doc/html/structfuse__entry__param.html
fuse-2.9.7/doc/html/globals_type.html
fuse-2.9.7/doc/html/nav_f.png
fuse-2.9.7/doc/html/structfuse__context.html
```

备注：部分较老的操作系统（如CentOS6），可能出现 jemalloc 库编译失败的问题。对于64GB内存以内的机器，可以通过在 bin/goosefs-lite 的第二行加入 `export GOOSEFS_DISABLE_CALLOC=true` 跳过此错误。对于64GB 内存以上的机器，建议升级 gcc 版本，重新编译。

## 步骤5：配置密钥文件

进入 `goosefs-lite--${version}` 目录下，修改配置文件 `conf/core-site.xml` 中的配置项；执行时可以将版本号替换为公开的版本号，如1.0.4，示例如下：

```
[root@VM-0-11-tencentos goosefs-lite-1.0.4]# ll conf
total 20
-rw-r--r-- 1 501 games 1050 Nov 13 20:37 core-site.xml
-rw-r--r-- 1 501 games 450 Nov 13 20:37 core-site.xml.template
-rw-r--r-- 1 501 games 0 Nov 13 20:37 goosefs-lite.properties
-rw-r--r-- 1 501 games 32 Nov 13 20:37 goosefs-site.properties
-rw-r--r-- 1 501 games 1134 Nov 13 20:37 log4j.properties
-rw-r--r-- 1 501 games 977 Nov 13 20:37 log4j.properties.template
[root@VM-0-11-tencentos goosefs-lite-1.0.4]# vim conf/core-site.xml
```

- 将 fs.cosn.userinfo.secretKey 配置为腾讯云密钥 Key。
- 将 fs.cosn.userinfo.secretId 配置为腾讯云密钥 ID。
- 将 fs.cosn.bucket.region 配置为存储桶地域。

#### ⚠ 注意

- 建议用户尽量避免在配置中使用永久密钥，采取配置子账号密钥或者临时密钥的方式有助于提升业务安全性。为子账号授权时建议按需授权子账号可执行的操作和资源，避免发生预期外的数据泄露。
- 如果您一定要使用永久密钥，建议对永久密钥的权限范围进行限制，可通过限制永久密钥的可执行操作、资源范围和条件（访问 IP 等），提升使用安全性。

#### 示例：

```
<configuration>
  <!--账户的 API 密钥信息。可登录 [访问管理控制台]
  (https://console.cloud.tencent.com/capi) 查看云 API 密钥。-->
  <!--建议使用子账号密钥或者临时密钥的方式完成配置，提升配置安全性。为子账号授权时建议
  按需授权子账号可执行的操作和资源-->
  <property>
    <name>fs.cosn.impl</name>
    <value>org.apache.hadoop.fs.CosFileSystem</value>
  </property>
  <property>
    <name>fs.cosn.userinfo.secretId</name>
    <value>*****</value>
  </property>
  <property>
    <name>fs.cosn.userinfo.secretKey</name>
    <value>****</value>
  </property>
  <property>
    <name>fs.cosn.bucket.region</name>
    <value>ap-guangzhou</value>
  </property>
  <property>
    <name>fs.cosn.read.ahead.queue.size</name>
    <value>16</value>
  </property>
  <property>
    <name>fs.cosn.upload_thread_pool</name>
    <value>32</value>
  </property>
```

```

<property>
  <name>fs.cosn.tmp.dir</name>
  <value>/data/goosefs-lite-tmp/</value>
</property>
<property>
  <name>fs.cosn.posix_extension.enabled</name>
  <value>>false</value>
</property>
<property>
  <name>fs.cosn.posix_extension.tmp.dir.quota</name>
  <value>10737418240</value>
</property>
<property>
  <name>fs.cosn.filestatus.list.recursive.enabled</name>
  <value>>true</value>
</property>

</configuration>
    
```

如果您挂载的是 [元数据加速](#) 存储桶，则需要您先对元数据加速存储桶进行如下配置：

1. 性能配置 > HDFS 权限配置 > HDFS 用户配置，添加超级用户，例如 root。
2. 性能配置 > HDFS 权限配置 > HDFS 权限配置，添加访问 COS 存储桶的 VPC 和 IP 地址信息。
3. 在本机执行如下命令，添加 hadoop 用户和 supergroup 组。

```

useradd hadoop
groupadd supergroup
    
```

4. 修改配置信息：

```

<configuration>
  <!--账户的 API 密钥信息。可登录 [访问管理控制台]
  (https://console.cloud.tencent.com/capi) 查看云 API 密钥。-->
  <!--建议使用子账号密钥或者临时密钥的方式完成配置，提升配置安全性。为子账号授权时建议
  按需授权子账号可执行的操作和资源-->
  <property>
    <name>fs.cosn.impl</name>
    <value>org.apache.hadoop.fs.CosFileSystem</value>
  </property>
  <property>
    <name>fs.cosn.userinfo.secretId</name>
    <value>AKIDnQxxxxx</value>
  </property>
  <property>
    
```

```
<name>fs.cosn.userinfo.secretKey</name>
<value>YYYY</value>
</property>
<property>
<name>fs.cosn.bucket.region</name>
<value>ap-guangzhou</value>
</property>
<property>
<name>fs.cosn.trsf.fs ofs.bucket.region</name>
<value>ap-guangzhou</value>
</property>
<property>
<name>fs.cosn.trsf.fs ofs.plugin.info.log</name>
<value>>true</value>
</property>
<property>
<name>fs.cosn.trsf.fs ofs.user.appid</name>
<value>1250000000</value>
</property>
<property>
<name>fs.cosn.trsf.fs ofs.tmp.cache.dir</name>
<value>/data/tmp/ofs</value>
</property>
</configuration>
```

## 步骤5: 运行工具

执行如下命令，将密钥文件中配置的存储桶挂载到指定目录：

```
./bin/goosefs-lite mount <MountPoint> cosn://<BucketName>/
```

其中：

- <MountPoint> 为本地挂载目录（例如 /mnt/goosefs-lite-mnt-dir）。
- <BucketName> 为存储桶名称（例如 examplebucket-1250000000）。

示例：

```
mkdir -p /mnt/goosefs-lite-mnt
./bin/goosefs-lite mount /mnt/goosefs-lite-mnt/ cosn://examplebucket-1250000000/
```

查看本地挂载点和对应的 COS 存储桶，输出信息依次为进程 ID、本地挂载点和 COS 路径：

```
$ ./bin/goosefs-lite stat
```

```
pid mount_point cos_path
13815 /mnt/goosefs-lite-mnt/ cosn://examplebucket-1250000000/
```

如果您需要在命令行中，同时指定多个挂载参数，可以使用逗号分隔多个参数，例如，下面的命令设置挂载点只读，且允许除其他用户访问挂载点：

```
./bin/goosefs-lite mount -o"ro,allow_other" mnt/ cosn://examplebucket-1250000000/
```

其中：

- `-oallow_other`：如果要允许其他用户访问挂载文件夹，可以在运行 GooseFS-Lite 的时候指定该参数。
- `-oro`：将挂载点设置为只读，不允许写入和删除操作。

### ⓘ 说明

单个参数可通过 `-o` 指定，例如 `-oro`；多个参数可通过逗号分割，例如 `-o"ro,allow_other"`。

## 步骤6：卸载存储桶

卸载存储桶示例：

```
./bin/goosefs-lite umount /mnt/goosefs-lite-mnt
Unmount fuse at /mnt/goosefs-lite-mnt/ (PID: 17206).
```

## 步骤7：参数调优

GooseFS-Lite 包含两个配置文件，分别为 `conf/core-site.xml` 及 `conf/goosefs-lite.properties`。您可以通过修改 `conf/core-site.xml` 调优上传下载带宽。常用参数如下，更多参数可参见 [Hadoop-COS](#) 文档。

属性键	说明	默认值	必填项
<code>fs.cosn.useHttps</code>	配置是否使用 HTTPS 作为与 COS 后端的传输协议。	true	否
<code>fs.cosn.upload.part.size</code>	分块上传的每个 part size 的大小。由于 COS 的分块上传最多只能支持10000块，因此需要预估最大可能使用到的单文件大小。例如，part size 为8MB时，最大能够支持78GB的单文件上传。part size 最大可以支持到2GB，即单文件最大可支持19TB。	8388608 (8MB)	否
<code>fs.cosn.upload_thread_pool</code>	文件流式上传到 COS 时，并发上传的线程数目。	10	否

fs.cosn.read.ahead.block.size	预读块的大小。	1048576 (1MB)	否
fs.cosn.read.ahead.queue.size	预读队列的长度。	8	否
fs.cosn.trsf.fs ofs.tmp.cache.dir	元数据加速存储桶的临时文件目录。	无	是
fs.cosn.trsf.fs ofs.user.appid	元数据加速存储桶的 Appid。	无	是
fs.cosn.trsf.fs ofs.bucket.region	元数据加速存储桶所在的地域，如 ap-shanghai, ap-beijing。	无	是

您可以通过修改 `conf/goosefs-lite.properties` 对 GooseFS-Lite 的行为进行调整。常用参数如下：

属性	说明	默认值	必填项
goosefs.fuse.list.entries.cache.enabled	是否开启客户端 List 缓存	true	否
goosefs.fuse.list.entries.cache.max.size	客户端 List 最大缓存的条目数，单位：条	100000	否
goosefs.fuse.list.entries.cache.max.expiration.time	客户端 List 缓存的有效时间，单位：ms	15000	否
goosefs.fuse.async.release.max.wait.time	open 和 rename 操作的文件正在被写入时，等待写入操作完成的时间，单位：ms	5000	否
goosefs.fuse.umount.timeout	卸载文件系统时，等待未完成操作的时间，单位：ms	120000	否

当您的读取和写入并发度较大，您可以通过如下方式，调整 GooseFS-Lite 最大 JVM 运行内存，避免 FullGC 和 OutOfMemoryError。JVM 默认值为

`-Xms2G -Xmx2G -XX:MaxDirectMemorySize=4G -XX:+UseG1GC`，调整方式如下：

```
export JAVA_OPTS="-Xms16G -Xmx16G -XX:MaxDirectMemorySize=16G -XX:+UseG1GC"
./bin/goosefs-lite mount /mnt/goosefs-lite-mnt/ cosn://examplebucket-1250000000/
ps -ef|grep goosefs-lite|grep -v grep
```

## 常见问题

### 缺少 libfuse 库文件，该如何处理？

```
java.lang.UnsatisfiedLinkError: /tmp/libjnihfuse2359305659114946750.so: libfuse.so.2: cannot open shared object file: No such file or directory
    at java.base/java.lang.ClassLoader$NativeLibrary.load0(Native Method)
    at java.base/java.lang.ClassLoader$NativeLibrary.load(ClassLoader.java:2445)
    at java.base/java.lang.ClassLoader$NativeLibrary.loadLibrary(ClassLoader.java:2501)
    at java.base/java.lang.ClassLoader.loadLibrary0(ClassLoader.java:2700)
    at java.base/java.lang.ClassLoader.loadLibrary(ClassLoader.java:2630)
    at java.base/java.lang.Runtime.load0(Runtime.java:768)
    at java.base/java.lang.System.load(System.java:1837)
    at com.qcloud.cos.goosefs.jnihfuse.utils.NativeLibraryLoader.loadLibraryFromJar(NativeLibraryLoader.java:105)
    at com.qcloud.cos.goosefs.jnihfuse.utils.NativeLibraryLoader.loadLibrary(NativeLibraryLoader.java:83)
    at com.qcloud.cos.goosefs.jnihfuse.LibFuse.loadLibrary(LibFuse.java:48)
    at com.qcloud.cos.goosefs.jnihfuse.LibFuse.<clinit>(LibFuse.java:32)
    at com.qcloud.cos.goosefs.jnihfuse.AbstractFuseFileSystem.<clinit>(AbstractFuseFileSystem.java:41)
    at com.qcloud.cos.goosefs.fuse.GooseFSLiteFuse.main(GooseFSLiteFuse.java:64)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.base/java.lang.reflect.Method.invoke(Method.java:566)
    at org.springframework.boot.loader.MainMethodRunner.run(MainMethodRunner.java:53)
    at java.base/java.lang.Thread.run(Thread.java:829)
```

需要安装 libfuse，请参照以下方式进行操作：

- 方式一

1. 执行如下命令，安装 fuse-devel。

- 如果是 CentOS 或者 TencentOS 系统，执行如下命令：

```
yum install fuse-devel
```

- 如果是 Ubuntu 系统，执行如下命令：

```
apt install libfuse-dev
```

2. 安装完成后，执行如下命令，查看安装是否成功。

```
find / -name libfuse.so*
```

- 方式二

更新旧版本 libfuse.so.2.9.2，安装步骤如下：

① 说明

CentOS 7 默认安装的是 libfuse.so.2.9.2。

1. 下载 [libfuse 源码](#)，并编译生成 libfuse.so.2.9.7。

```
tar -zxvf fuse-2.9.7.tar.gz
cd fuse-2.9.7/ && ./configure && make && make install
echo -e '\n/usr/local/lib' >> /etc/ld.so.conf
ldconfig
```

2. 编译及生成 libfuse.so.2.9.7 后，可按照以下步骤进行替换：

2.1 执行以下命令，查找旧版本 libfuse.so.2.9.2 库链接。

```
find / -name libfuse.so*
```

2.2 执行以下命令，将 libfuse.so.2.9.7 拷贝至旧版本库 libfuse.so.2.9.2 所在位置。

```
cp /usr/local/lib/libfuse.so.2.9.7 /usr/lib64/
```

2.3 执行以下命令，删除旧版本 libfuse.so 库的所有链接。

```
rm -f /usr/lib64/libfuse.so
rm -f /usr/lib64/libfuse.so.2
```

2.4 执行以下命令，建立与被删除旧版本链接类似的 libfuse.so.2.9.7 库链接。

```
ln -s /usr/lib64/libfuse.so.2.9.7 /usr/lib64/libfuse.so
ln -s /usr/lib64/libfuse.so.2.9.7 /usr/lib64/libfuse.so.2
```

## 如何配置开机挂载？

1. 编辑文件 /usr/lib/systemd/system/goosefs-lite.service，追加如下内容，您可以将 examplebucket-1250000000 换为您的存储桶：注意，以下 JAVA\_OPTS 中 -Xms 和 -Xmx 配置的内存值不能超过所在节点的物理内存上限的 50%。如节点物理内存 16GB，那么建议最多配置 -Xms8G -Xmx8G。

```
[Unit]
Description=The Tencent Cloud GooseFS Lite for COS
Requires=network-online.target
After=network-online.target

[Service]
Type=forking
User=root
```

```
Environment="JAVA_OPTS=-Xms16G -Xmx16G -XX:MaxDirectMemorySize=16G -
XX:+UseG1GC"
ExecStart=/usr/local/goosefs-lite-1.0.0/bin/goosefs-lite mount /mnt/goosefs-mnt
cosn://examplebucket-1250000000/
ExecStop=/usr/local/goosefs-lite-1.0.0/bin/goosefs-lite umount /mnt/goosefs-mnt
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
```

2. 执行如下命令，执行挂载命令和查看后台 Daemon 进程状态：

```
# 让 goosefs-lite 的 systemd 配置生效
systemctl daemon-reload
# 启动后台 Fuse 进程
systemctl start goosefs-lite
# 查看后台 Daemon 进程状态
systemctl status goosefs-lite
# 查看挂载点列表
/usr/local/goosefs-lite-1.0.0/bin/goosefs-lite stat
# 如果是修改systemd配置，则在修改后reload并restart。
```

设置为开机启动时尝试挂载：

```
systemctl enable goosefs-lite
```

3. 卸载挂载点，重启机器，并查看 Fuse 进程状态：

```
# 执行卸载，注意：请勿在数据写入的时卸载，否则会导致数据不完整
systemctl stop goosefs-lite
# 重启操作系统，请谨慎操作，不要影响业务
reboot -h now
# 查看后台 Daemon 进程状态
systemctl status goosefs-lite
# 查看挂载点列表
/usr/local/goosefs-lite-1.0.0/bin/goosefs-lite stat
```

**GooseFS-Lite 每天在某个时间段里 CPU 使用率较高，且向 COS 发出大量 Head、List 请求，产生大量请求次数费用，该怎么处理？**

这通常是由于您机器上存在定时扫盘任务导致的，Linux 系统上常见的扫盘程序是 updatedb，您可以将 GooseFS-Lite 挂载点目录，添加到 updatedb 的配置文件 /etc/updatedb.conf 文件的 PRUNEPATHS 配置项中，避免该程序的扫盘行为。此外，您可以使用 Linux 工具 auditd，查找访问 GooseFS-Lite 挂载点的程序。

操作步骤如下：

### 1. 安装 auditd

Ubuntu：

```
apt-get install auditd -y
```

CentOS：

```
yum install audit audit-libs
```

### 2. 启动 auditd 服务

```
systemctl start auditd  
systemctl enable auditd
```

### 3. 监控挂载目录

#### ⓘ 说明

`-w` 指定 GooseFS-Lite 挂载目录，`-k` 为输出在 audit 日志中的 key。

```
auditctl -w /usr/local/service/mnt/ -k gosefs_lite_mnt
```

### 4. 根据日志确定访问程序

audit 的日志目录： /var/log/audit，查询命令如下：

```
ausearch -i|grep 'gosefs_lite_mnt'
```

### 5. 停止 auditd 服务

如果您需要停止 auditd 服务，可以使用如下命令：

```
/sbin/service auditd stop
```

#### ⚠ 注意

如果访问挂载点的程序一直在运行，新启动的 auditd，并不会监控到该程序的访问行为；程序中关于挂载目录的多次调用，只会记录第一次。

## GooseFS-Lite 安装时报 “cannot allocate memory” 错误

```
OpenJDK 64-Bit Server VM warning: INFO: os::commit_memory(0x0000000080000000, 2147483648, 0) failed; error='Cannot allocate memory' (errno=12)
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Native memory allocation (mmap) failed to map 2147483648 bytes for committing reserved memory.
# An error report file with more information is saved as:
# /usr/goosefs-lite/goosefs-lite-1.0.2/hs_err_pid2860293.log
~
```

该报错主要是因为 GooseFS-Lite 运行过程时分配内存异常报错，一般为申请内存小于实际可用内存。您可以进入 `./bin/goosefs-lite` 文件中修改 `JAVA_OPT` 参数配置为合理内存数值，至少需要保证申请的内存数量小于实例可用内存数量。