# 3. 第三章 glusterfs 的核心概念

从本章开始，有了前面的关于 linux 的一些文件系统的简单认识，那么从这里开始真正去认识一下 glusterfs 这种无中心架构的特点，因为无中心架构，那么必然在元数据存储与节点之间通信 ,数据恢复等方面会与常见的架构有着非常大的不同，而这也是 glusterfs 的魅力之处，下面开始将一点点地深入去理解 glusterfs 是如何运作的。另外这里先做个简单的约定，下面的代码内容如无特殊说明，则基本是在虚拟机环境中进行的实验操作，对于集群节点则是以 gfs01,gfs02 这样来命令规范的，而客户端机器则是 gfsclient01 这样的，因此阅读代码的时候，可以进行区别开来。

关于实验的虚拟机的 ip 与 hostname 对应关系如下所示..

```
# cat /etc/hosts
...
192.168.0.110 gfs01
192.168.0.111 gfs02
192.168.0.112 gfs03
```

## 3.1. 有趣的扩展属性 gfid

### 3.1.1. 文件的 gfid 属性

因为 glusterfs 是无中心架构的，这点在前面的内容中也多次提到过，当然可能会有困惑，那么到底这个文件是存放到哪里的，由什么来决定呐？对于 glusterfs 来说，这里巧妙地利用了 linux 文件的扩展属性，而一个文件除了有基本的元数据以外，还支持扩展属性的，而其中有一个叫做 gfid 的扩展属性是

关键的作用，下面来感受一下。

```
# gluster volume info test-replica

Volume Name: test-replica
Type: Replicate
Volume ID: 4568c063-5b75-4304-98f6-21f3955cc138
Status: Started
Snapshot Count: 0
Number of Bricks: 1 x 3 = 3
Transport-type: tcp
Bricks:
Brick1: 192.168.0.110:/glusterfs/test-replica
Brick2: 192.168.0.111:/glusterfs/test-replica
Brick3: 192.168.0.112:/glusterfs/test-replica
Options Reconfigured:
cluster.granular-entry-heal: on
storage.fips-mode-rchecksum: on
transport.address-family: inet
nfs.disable: on
performance.client-io-threads: off
```

首先这里有一个 3 副本的复制卷，然后在一个客户端虚拟机上面进行挂载，接着查看一下扩展属性。

```
[root@gfsclient01 ~]# mount -t glusterfs  -o aux-gfid-mount   192.168.0.110:test-replica /mnt/test-replica
[root@gfsclient01 ~]# cd /mnt/test-replica/
[root@gfsclient01 test-replica]# getfattr -n glusterfs.gfid.string  a.txt
# file: a.txt
glusterfs.gfid.string="71b9fb49-53ae-42f8-bb1b-b53af17521fc"
```

从这里可以看到 glusterfs 的文件是有一个叫做 gfid 的扩展属性的，那么这里还可以到 brick 所在的目录下面进行查看一下信息。

```
[root@gfs01 test-replica]# getfattr -d -m . -e hex a.txt
# file: a.txt
trusted.gfid=0x71b9fb4953ae42f8bb1bb53af17521fc
```

```
4.  trusted.gfid2path.640ca896bde02acc=0x30303030303030302d303030302d303030302d303030302d3030303030303030303030312f612e747874
5.  trusted.glusterfs.mdata=0x010000000000000000000000060ba3cdf000000001ca8c3c90000000060ba3cdf000000001ca8c3c90000000060ba3cdf000000001ca8c3c9
6.
7.  [root@gfs02 ~]# cd /glusterfs/test-replica
8.  [root@gfs02 test-replica]# getfattr -d -m . -e hex a.txt
9.  # file: a.txt
10. trusted.gfid=0x71b9fb4953ae42f8bb1bb53af17521fc
11. trusted.gfid2path.640ca896bde02acc=0x30303030303030302d303030302d303030302d303030302d3030303030303030303030312f612e747874
12. trusted.glusterfs.mdata=0x010000000000000000000000060ba3cdf000000001ca8c3c90000000060ba3cdf000000001ca8c3c90000000060ba3cdf000000001ca8c3c9
```

那么这里可以看到 gfid 是相同的，在复制卷里面，因为都是相同的文件，因此这里的 gfid 显示一致,那么这个 gfid 是不是 inode 呢？其实不是的，gfid 是客户端计算出来的一个独一无二的类似 uuid 这样的字符串，另外下面还可以查看 inode 和其他信息。

```
1.  [root@gfsclient01 test-replica]# ls -i a.txt
2.  13482569174227427836 a.txt
3.  [root@gfsclient01 test-replica]# ls -l a.txt
4.  -rw-r--r--. 1 root root 0 Jun  4 10:46 a.txt
5.
6.  [root@gfs01 test-replica]# ls -l a.txt
7.  -rw-r--r-- 2 root root 0 Jun  4 10:46 a.txt
8.  [root@gfs01 test-replica]# ls -i a.txt
9.  68224979 a.txt
```

这里从客户端的与其中一个 brick 目录下对比 inode 就发现，首先和 gfid 是不一样的，同时两者的 inode 也不是相同的。另外这里还可以观察到文件中的链接文件的数量是不一样的，客户端中的是 1，而 brick 目录中的文件是 2，说明这里还有其他文件引用了该文件，而引用的地方就是 glusterfs 的一个隐藏目录，这个地方就是每个 brick 记录源数据信息的一个重要目录了。

```
[root@gfs01 test-replica]# ls -i  .glusterfs/71/b9/71b9fb49-53ae-42f8-bb1b-b53af17521fc
68224979 .glusterfs/71/b9/71b9fb49-53ae-42f8-bb1b-b53af17521fc
[root@gfs01 test-replica]# ls -i a.txt
68224979 a.txt
```

从这里就可以知道 ,在这个.glusterfs 隐藏目录下面 ,有一个该文件的硬链接，而其中 gfid 的前四位，每两位就是隐藏目录下面的目录名称。因此这里就是非常重要的地方了，因此对于 glusterfs 来说，所有的操作，对文件的增删改查，都是需要先找到该文件的 ,而寻找该文件 ,也就是 lookup 操作 ,都是基于 inode 的，而不是直接去找 path 的。

### 3.1.2. 目录的 gfid 属性

前面看了文件的 gfid 属性，那么下面来看看目录的该属性是否有不一样的地方。首先还是在客户端挂载，然后创建一个目录，然后查看一下属性。

```
[root@gfsclient01 test-replica]# getfattr -n glusterfs.gfid.string  dir01
# file: dir01
glusterfs.gfid.string="a0e974d1-0902-40ba-ad21-7436f8bf31cf"


[root@gfs03 test-replica]# ls -l
total 0
-rw-r--r-- 2 root root  0 Jun  4 10:46 a.txt
drwxr-xr-x 2 root root 19 Jun  4 11:20 dir01
drwxr-xr-x 2 root root  6 Jun  4 11:20 dir02
[root@gfs03 test-replica]# getfattr  -d -m . -e hex dir01

# file: dir01
trusted.gfid=0xa0e974d1090240baad217436f8bf31cf
trusted.glusterfs.dht=0x00000000000000000000000ffffffff
```

```
trusted.glusterfs.mdata=0x010000000000000000000000060ba3cd80000000024ef360f0000000060ba3cd80000000024ef360f0000000060ba3ccc000000000a37a017

[root@gfs03 test-replica]# ls -l .glusterfs/a0/e9/
total 0
lrwxrwxrwx 1 root root 54 Jun  4 11:20 a0e974d1-0902-40ba-ad21-7436f8bf31cf -> ../../00/00/00000000-0000-0000-0000-000000000001/dir01
[root@gfs03 test-replica]# ls -i dir01
67605992 1.txt
[root@gfs03 test-replica]# ls -i .glusterfs/a0/e9/a0e974d1-0902-40ba-ad21-7436f8bf31cf
67605992 1.txt

[root@gfs03 test-replica]# ls -l .glusterfs/00/00/
total 0
lrwxrwxrwx 1 root root 8 May 25 12:11 00000000-0000-0000-0000-000000000001 -> ../../..
```

从代码中可以看到，这里隐藏目录下，也是有一个链接的，而且 inode 都是相同的，但是注意的是，这里的链接文件的路径是很多个 0，然后最后是 1，这个就是表示在 brick 下的根目录下 ,那么如果该目录不是在根目录下呢？我们可以在 dir01 下面再创建一个目录进行查看。

```
//在客户端下创建一个目录
[root@gfsclient01 test-replica]# mkdir -p dir01/dir001


//在 brick 下再次查看属性
[root@gfs03 test-replica]# getfattr  -d -m . -e hex dir01/dir001
# file: dir01/dir001
trusted.gfid=0xa1064e3fa66e4663a69a112ccad4eaaa
trusted.glusterfs.dht=0x00000000000000000000000ffffffff
```

```
trusted.glusterfs.mdata=0x010000000000000000000000060bc7b0b0000000026031064000000060bc7b0b0000000026031064000000060bc7b0b0000000026031064

[root@gfs03 test-replica]# ls -l .glusterfs/a1/06/a1064e3f-a66e-4663-a69a-112ccad4eaaa
lrwxrwxrwx 1 root root 55 Jun  6 03:36 .glusterfs/a1/06/a1064e3f-a66e-4663-a69a-112ccad4eaaa -> ../../a0/e9/a0e974d1-0902-40ba-ad21-7436f8bf31cf/dir001

[root@gfs03 test-replica]# ls .glusterfs/a0/e9/a0e974d1-0902-40ba-ad21-7436f8bf31cf -l
lrwxrwxrwx 1 root root 54 Jun  4 11:20 .glusterfs/a0/e9/a0e974d1-0902-40ba-ad21-7436f8bf31cf -> ../../00/00/00000000-0000-0000-0000-000000000001/dir01
```

从这里的信息显示，很明显，不管是创建文件还是目录，这里会在隐藏目录下面创建一个链接文件，而这里的 dir001 因为是在目录 dir01 下面的，所以隐藏目录下的上一层路径就是父目录的 gfid，然后再通过父目录的 gfid 就可以找到父目录的路径了。接着观察一下那个很多个 0 最后是 1 的目录隐藏路径下的内容吧。

```
[root@gfs03 test-replica]# ls -l .glusterfs/00/00/00000000-0000-0000-0000-000000000001/
total 0
-rw-r--r-- 2 root root  0 Jun  4 10:46 a.txt
drwxr-xr-x 3 root root 33 Jun  6 03:36 dir01
drwxr-xr-x 2 root root  6 Jun  4 11:20 dir02
drwxr-xr-x 2 root root  6 Jun  6 03:43 dir03

[root@gfs03 test-replica]# ls -l /glusterfs/test-replica/
total 0
-rw-r--r-- 2 root root  0 Jun  4 10:46 a.txt
drwxr-xr-x 3 root root 33 Jun  6 03:36 dir01
drwxr-xr-x 2 root root  6 Jun  4 11:20 dir02
drwxr-xr-x 2 root root  6 Jun  6 03:43 dir03
```

通过这里的一些例子的观察，当然这里只是两层目录，如果多层目录下面，

大家可以测试一下效果会是怎样的。

最后这里简单总结一下 gfid 这个扩展属性的特点:

1. 在复制卷中，每一个 brick 的文件和目录的 gfid 都是相同的。

2. 创建文件和目录，都会在隐藏目录下面.glusterfs 创建对应的链接文件

3. gfid 属性的这一串字符里面，前四位是用于标识隐藏目录名称的。其中每两位是一个隐藏目录下的目录名称。

4. 隐藏目录中，对文件创建是硬链接，目录是软连接。

那么最后了解一下，这个 gfid 到底是怎么来的呢？这里可以在代码中找到相关的内容，libglusterfs/src/inode.c 中有一个 hash_gfid 的函数。

```
1.  static int
2.  hash_gfid(uuid_t uuid, int mod)
3.  {
4.      return ((uuid[15] + (uuid[14] << 8)) % mod);
5.  }
```