问题线上描述

接触问题得到的初步信息：

- 线上环境使用gluster volume heal volxxxx info总是卡顿
- 线上环境用户的流量并不大
- 最开始搭建环境的时候并不卡，后面使用一段时间之后就不卡可
- 还有就是一般不卡，偶尔使用会卡

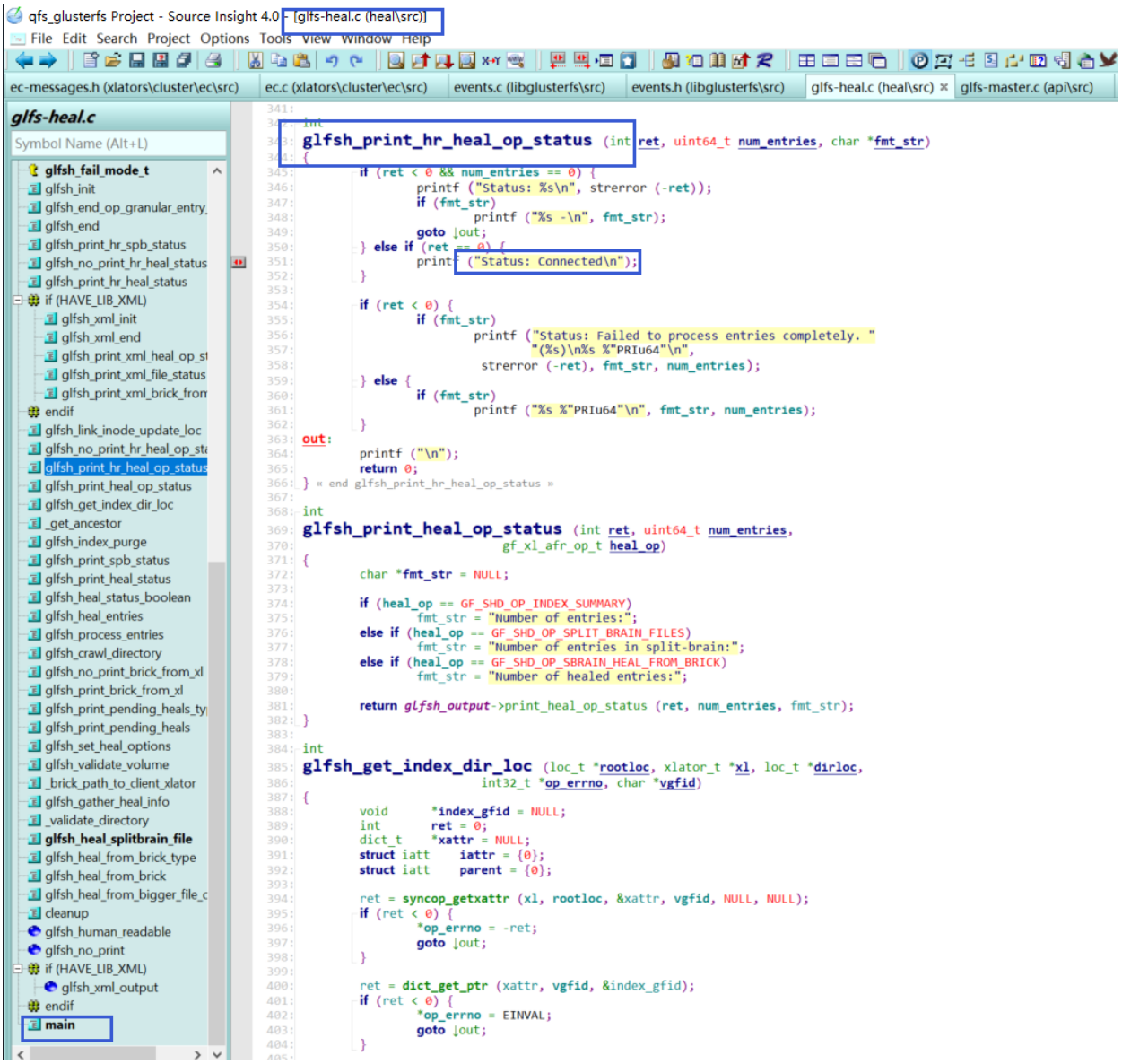
heal命令的主逻辑文件

启动查询命令的时候，如果卡顿可以发现 有两个heal相关的进程

```
Status: Connected
Number of entries: 0

Brick storage0-cluster0:/data/brk05/src
Status: Connected
Number of entries: 0

Brick storage1-cluster0:/data/brk05/src
Status: Connected
Number of entries: 0

Brick storage2-cluster0:/data/brk05/src
Status: Connected
Number of entries: 0

root@storage1-cluster0:~#
```

```
root@storage1-cluster0:~# ps axu | grep heal
root        3396  0.0  0.0      0     0 ?        S<   Sep17   0:00 [mlx5_health0000]
root        6732  0.0  0.0      0     0 ?        S<   Sep17   0:00 [mlx5_health0000]
root        8279  0.0  0.0      0     0 ?        S<   Sep17   0:00 [mlx5_health0000]
root        8591  0.0  0.0      0     0 ?        S<   Sep17   0:00 [mlx5_health0000]
root       40067  0.0  0.0 414024  4152 pts/4    Sl+  14:58   0:00 gluster vol heal vol0 info
root       40073  0.2  0.0 2879676 17176 pts/4   SLl+ 14:58   0:00 /usr/sbin/glfsheal vol0
root       43558  0.0  0.0 112584   956 pts/5    S+   14:59   0:00 grep --color=auto heal
root@storage1-cluster0:~#
```

最后发现输出存在“Status: Connected”这样的日志，搜索代码发现文件和方法



该文件glfs-heal.c是heal命令的主逻辑文件

解决过程：

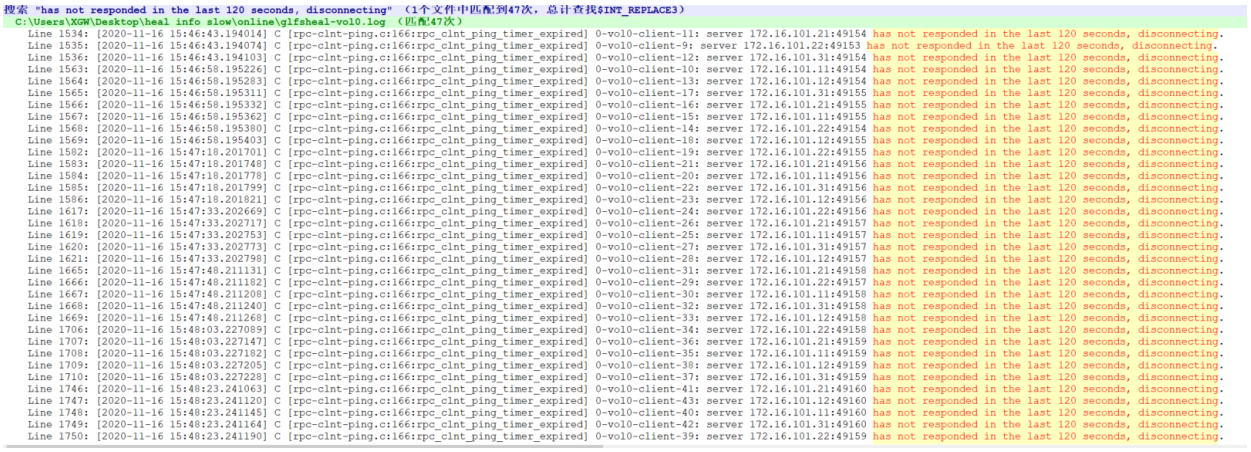
侥幸解决方法：

由于是用户环境不太敢随意操作，debug日志没有打卡开，使用"gluster volume heal volxxxx info"输出一些日志

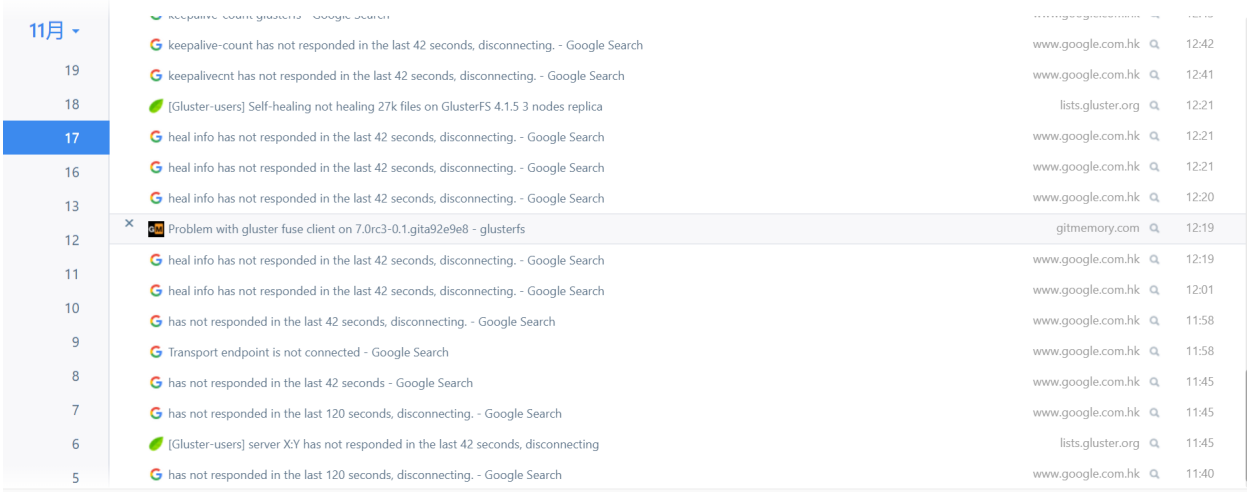
```
root@storage1-cluster0:~# gluster vol heal vol0 info
```

```
total 968612
drwxr-xr-x 3 root root        18 Sep  3 19:42 snaps
drwxr-xr-x 2 root root      8192 Nov 16 03:14 bricks
-rw------- 1 root root         0 Nov 18 15:29 cmd_history.log-20201115
-rw------- 1 root root         0 Nov 18 15:29 cli.log-20201115
-rw------- 1 root root         0 Nov 18 15:29 glusterd.log-20201115
-rw------- 1 root root         0 Nov 18 15:29 vol0-rebalance.log
-rw------- 1 root root         0 Nov 18 15:29 glustershd.log-20201115
-rw------- 1 root root       224 Nov 18 18:06 cmd_history.log
-rw------- 1 root root    377527 Nov 18 18:07 glfsheal-vol0.log-20201119
-rw------- 1 root root     17948 Nov 20 02:44 glusterd.log
-rw------- 1 root root       325 Nov 20 14:58 glfsheal-vol.log
-rw------- 1 root root 459676946 Nov 20 15:31 glustershd.log
-rw------- 1 root root      7099 Nov 20 15:31 cli.log
-rw------- 1 root root    969876 Nov 20 15:31 glfsheal-vol0.log
root@storage1-cluster0:~# ls -lrt /var/log/glusterfs/
[0] 0:root@storage1-cluster0:~*
```

发现该日志文件glfsheal-vol0.log有最新输出，看出该日志文件有一下CRITICAL输出，并且线上有总共有12*(4+1)个brick，前0-8个brick连接没有出现rpc_clnt_ping_timer_expired的日志



然后在google上搜索了一下关键词



出现了两个有代表性的表述:

Troubleshooting Issues

> Though this will likely match, sometimes there could be a bug leading to stale port usage. A quick workaround would be to restart glusterd on that node and check if things match. Report the issue to the devs if you see this problem.

seeing socket disconnects and transport endpoint not connected frequently on systemic setup

Doc Text: Cause: Due to heavy load, ping requests are not read fast enough so that they can be responded within the timeout.

Consequence: When ping requests are not responded within a timeout value, transport is disconnected.

Fix:
Two fixes:
1. Event threads no longer execute any code of Glusterfs program. Instead requests are queued to be consumed by threads of Glusterfs program. Since no code of Glusterfs program is executed by event-threads, they are not slowed down.
2. Even with queuing, under still high load default number of event threads are not sufficient to read requests. Hence the number of event-threads are suggested to be increased. In testing of this bug, client.event-threads and server.event-threads were set to 8. But this value can be experimented and we suggest to try a value of 4 and use 8 threads only if it doesn't resolve the issue.

Result:

确认restart glusterd和以及设置event-threads风险较小, 设置之后，使用查询命令依旧卡顿，将修改的配置修改回去

```
root@storage1-cluster0:~# gluster vol get vol0 all | grep event
client.event-threads                    32
server.event-threads                    128
root@storage1-cluster0:~# gluster^C
root@storage1-cluster0:~# gluster vol set vol0 client.event-threads 32
volume set: success
root@storage1-cluster0:~# gluster vol set vol0 server.event-threads 128
volume set: success
root@storage1-cluster0:~# service glusterd restart
Redirecting to /bin/systemctl restart glusterd.service
root@storage1-cluster0:~#
```

没有办法只能线下搭建一个环境，看代码逻辑，对照日志查问题

线下调试解决

本地搭建了一个测试环境，没有写任何数据，使用查询命令依旧卡顿，从而可以排除是数据量太多导致，当然数据量太多而卡顿有可能是另外一种情况，暂时没有遇到不去考虑

由于本地环境是35*(2+1)的配置，修改日志级别导致一次查询，出现太多干扰日志

```
root@storage1-cluster0:~# gluster vol set diaget vol0 all | grep level
diagnostics.brick-log-level             INFO
diagnostics.client-log-level            INFO
diagnostics.brick-sys-log-level         CRITICAL
diagnostics.client-sys-log-level        CRITICAL
network.compression.mem-level           8
network.compression.compression-level   -1
features.worm-file-level                off
root@storage1-cluster0:~# glusterclear^C
root@storage1-cluster0:~# gluster vol set vol0 diagnostics.client-log-level TRACE; gluster vol set vol0 diagnostics.brick-log-level TRACE
volume set: success
volume set: success
root@storage1-cluster0:~#
```

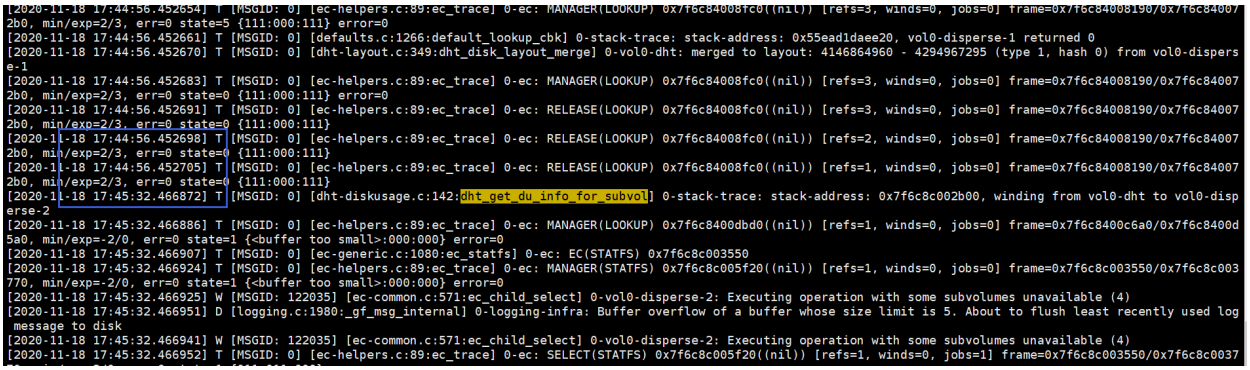
删除过多的brick，减少日志的数量

```
for((num=10;num<=35;num++)); do gluster volume remove-brick vol0 storage0-
cluster0:/data/brk$num/src  storage1-cluster0:/data/brk$num/src storage2-
cluster0:/data/brk$num/src force; done

for((num=6;num<=9;num++)); do gluster volume remove-brick vol0 storage0-
cluster0:/data/brk0$num/src  storage1-cluster0:/data/brk0$num/src storage2-
cluster0:/data/brk0$num/src force; done
```

最后只剩下6个(2+1)subvolume

```
root@storage0-cluster0:~# gluster vol info

Volume Name: vol0
Type: Distributed-Disperse
Volume ID: 9e85838f-d4ae-48af-a451-de336bb22161
Status: Started
Snapshot Count: 0
Number of Bricks: 6 x (2 + 1) = 18
Transport-type: tcp
Bricks:
Brick1: storage0-cluster0:/data/brk00/src
Brick2: storage1-cluster0:/data/brk00/src
Brick3: storage2-cluster0:/data/brk00/src
Brick4: storage0-cluster0:/data/brk01/src
Brick5: storage1-cluster0:/data/brk01/src
Brick6: storage2-cluster0:/data/brk01/src
Brick7: storage0-cluster0:/data/brk02/src
Brick8: storage1-cluster0:/data/brk02/src
Brick9: storage2-cluster0:/data/brk02/src
Brick10: storage0-cluster0:/data/brk03/src
Brick11: storage1-cluster0:/data/brk03/src
Brick12: storage2-cluster0:/data/brk03/src
Brick13: storage0-cluster0:/data/brk04/src
Brick14: storage1-cluster0:/data/brk04/src
Brick15: storage2-cluster0:/data/brk04/src
Brick16: storage0-cluster0:/data/brk05/src
Brick17: storage1-cluster0:/data/brk05/src
Brick18: storage2-cluster0:/data/brk05/src
Options Reconfigured:
diagnostics.client-log-level: TRACE
```

再次使用查询命令发现日志会在个时间点出现没有输出，这个有可能就是阻塞点



安装gdb和debuginfo-install方便调试

yum install gdb yum-utils -y debuginfo-install glusterfs-server-3.12.14-qs2.x86_64

```
root@storage1-cluster0:~# gdb /usr/sbin/glfsheal
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-114.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /usr/sbin/glfsheal...Reading symbols from /usr/sbin/glfsheal...(no
debugging symbols found)...done.
(no debugging symbols found)...done.
Missing separate debuginfos, use: debuginfo-install glusterfs-server-3.12.14-qs2.x86_64
```

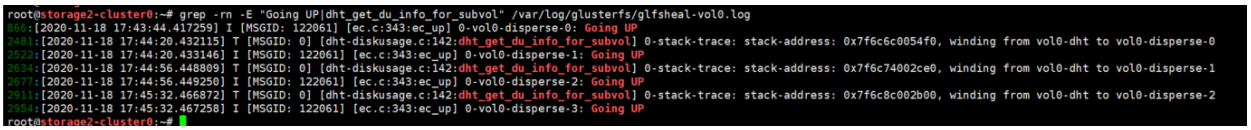
```
(gdb) r vol0
Starting program: /usr/sbin/glfsheal vol0
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
[New Thread 0x7ffff4ef3700 (LWP 15146)]
[New Thread 0x7ffff3bf2700 (LWP 15147)]
[New Thread 0x7ffff33f1700 (LWP 15148)]
[New Thread 0x7ffff1ff0700 (LWP 15149)]
[New Thread 0x7ffff17ef700 (LWP 15151)]
[New Thread 0x7fffeaac4700 (LWP 15152)]
[New Thread 0x7fffea0c0700 (LWP 15153)]
[New Thread 0x7fffe98bf700 (LWP 15154)]
[New Thread 0x7fffe80b6700 (LWP 15157)]
[New Thread 0x7fffe0ed4700 (LWP 15158)]
[New Thread 0x7fffd7fff700 (LWP 15159)]
[New Thread 0x7fffd77fe700 (LWP 15160)]
[New Thread 0x7fffd6ffd700 (LWP 15161)]
[New Thread 0x7fffd67fc700 (LWP 15162)]
[Switching to Thread 0x7fffe0ed4700 (LWP 15158)]

Breakpoint 1, dht_get_du_info_for_subvol (this=this@entry=0x7fffdc03df80, subvol_idx=0)
at dht-diskusage.c:112
112     {
```

```
Missing separate debuginfos, use: debuginfo-install bzip2-libs-1.0.6-13.el7.x86_64
elfutils-libelf-0.172-2.el7.x86_64 elfutils-libs-0.172-2.el7.x86_64 keyutils-libs-1.5.8-
3.el7.x86_64 krb5-libs-1.15.1-37.el7_6.x86_64 libacl-2.2.51-14.el7.x86_64 libattr-2.4.46-
13.el7.x86_64 libcap-2.22-9.el7.x86_64 libcom_err-1.42.9-13.el7.x86_64 libselinux-2.5-
14.1.el7.x86_64 libuuid-2.23.2-59.el7_6.1.x86_64 libxml2-2.9.1-6.el7_2.3.x86_64 ncurses-
libs-5.9-14.20130511.el7_4.x86_64 openssl-libs-1.0.2k-16.el7_6.1.x86_64 pcre-8.32-
17.el7.x86_64 readline-6.2-10.el7.x86_64 scylla-libgcc73-7.3.1-1.2.el7.centos.x86_64
systemd-libs-219-62.el7_6.6.x86_64 xz-libs-5.2.2-1.el7.x86_64 zlib-1.2.7-18.el7.x86_64

(gdb) bt
#0  dht_get_du_info_for_subvol (this=this@entry=0x7fffdc03df80, subvol_idx=0) at dht-
diskusage.c:112
#1  0x00007fffe303d1fc in dht_notify (this=0x7fffdc03df80, event=5, data=<optimized out>)
at dht-common.c:9287
#2  0x00007ffff7b06b62 in xlator_notify (xl=0x7fffdc03df80, event=event@entry=5,
data=data@entry=0x7fffdc036290) at xlator.c:566
#3  0x00007ffff7ba6d14 in default_notify (this=0x7fffdc036290, event=5,
data=0x7fffdc00f3c0) at defaults.c:3113
#4  0x00007fffe328d036 in ec_notify (this=<optimized out>, event=<optimized out>, data=
<optimized out>, data2=<optimized out>) at ec.c:577
#5  0x00007fffe328d2b9 in notify (this=<optimized out>, event=<optimized out>, data=
<optimized out>) at ec.c:594
#6  0x00007ffff7b06b62 in xlator_notify (xl=0x7fffdc036290, event=event@entry=5,
data=data@entry=0x7fffdc00f3c0) at xlator.c:566
#7  0x00007ffff7ba6d14 in default_notify (this=this@entry=0x7fffdc00f3c0,
event=event@entry=5, data=data@entry=0x0) at defaults.c:3113
#8  0x00007fffe3511e39 in client_notify_dispatch (this=0x7fffdc00f3c0,
event=event@entry=5, data=data@entry=0x0) at client.c:90
#9  0x00007fffe3511e9a in client_notify_dispatch_uniq (this=<optimized out>,
event=event@entry=5, data=data@entry=0x0) at client.c:68
#10 0x00007fffe3535406 in client_notify_parents_child_up (this=this@entry=0x7fffdc00f3c0)
at client-handshake.c:138
#11 0x00007fffe353753e in client_post_handshake (frame=frame@entry=0x7fffd0000b00,
this=0x7fffdc00f3c0) at client-handshake.c:1060
#12 0x00007fffe3537d43 in client_setvolume_cbk (req=<optimized out>, iov=<optimized out>,
count=<optimized out>, myframe=0x7fffd0000b00) at client-handshake.c:1244
#13 0x00007ffff701fe60 in rpc_clnt_handle_reply (clnt=clnt@entry=0x7fffdc0d7990,
pollin=pollin@entry=0x7fffd0005830) at rpc-clnt.c:778
#14 0x00007ffff7020147 in rpc_clnt_notify (trans=<optimized out>, mydata=0x7fffdc0d79c0,
event=<optimized out>, data=0x7fffd0005830) at rpc-clnt.c:971
#15 0x00007ffff701bf73 in rpc_transport_notify (this=this@entry=0x7fffdc0d7b60,
event=event@entry=RPC_TRANSPORT_MSG_RECEIVED, data=data@entry=0x7fffd0005830)
    at rpc-transport.c:538
#16 0x00007fffe8ead516 in socket_event_poll_in (this=this@entry=0x7fffdc0d7b60,
notify_handled=<optimized out>) at socket.c:2322
#17 0x00007fffe8eafabc in socket_event_handler (fd=19, idx=5, gen=1, data=0x7fffdc0d7b60,
poll_in=1, poll_out=0, poll_err=0) at socket.c:2474
#18 0x00007ffff7b66424 in event_dispatch_epoll_handler (event=0x7fffe0ed3e80,
event_pool=0x5555557d3ba0) at event-epoll.c:583
#19 event_dispatch_epoll_worker (data=0x7fffdc080ef0) at event-epoll.c:659
#20 0x00007ffff5b35163 in start_thread (arg=0x7fffe0ed4700) at pthread_create.c:309
#21 0x00007ffff5405f6d in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:111
```

并且在逻辑dht_get_du_info_for_subvol中对应的event type一定是"GF_EVENT_CHILD_UP" 检查调用栈会打印日志的地方：

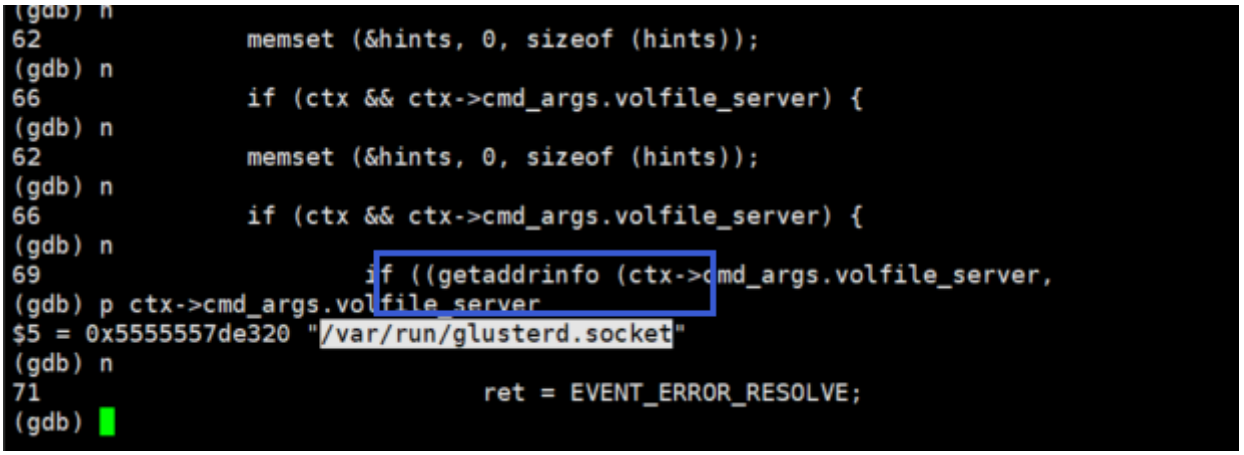
```
gf_msg_trace (this->name, 0, "NOTIFY(%d): %p, %p", event, data, data2); -> (ec_notify:
ec.c:511行)
gf_msg (this->name, GF_LOG_INFO, 0, EC_MSG_EC_UP, "Going UP"); -> (ec_up: ec.c:351行)
```

搜索日志



调用ec_up之后，接着就是dht_get_du_info_for_subvol，大概率是阻塞在ec_up中

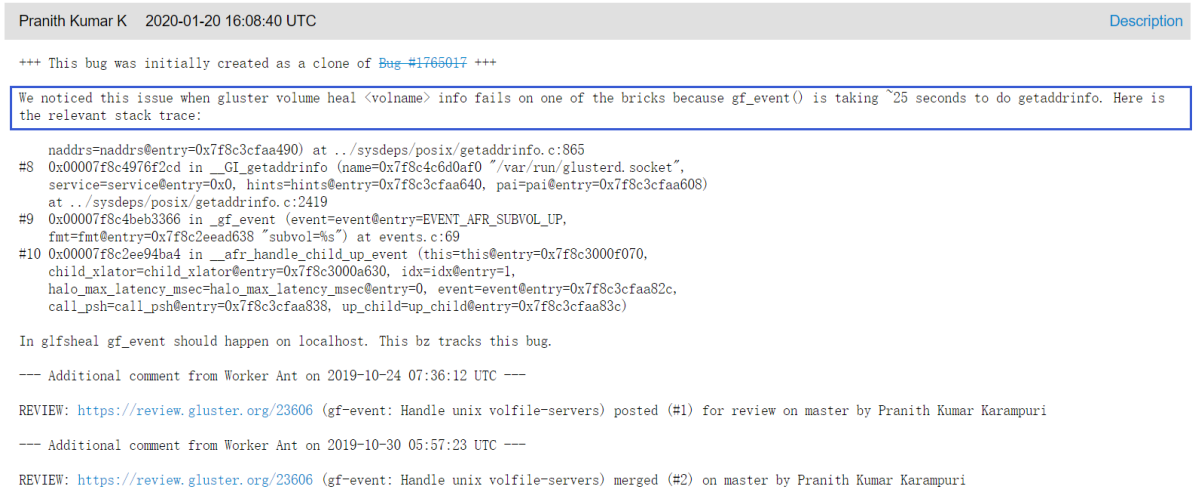
使用gdb调试ec_up发现阻塞在gf_event中，最终发现是阻塞在getaddrinfo方法中，并且该方法调用失败



然后在/etc/hosts下添加，gluster vol heal vol0 info能很快结束查询

```
127.0.0.1 /var/run/glusterd.socket
```

搜索关键字 gf_event发现Red Hat Bugzilla – Bug 1793085



Pranith Kumar K 2020-01-20 16:08:40 UTC Description

+++ This bug was initially created as a clone of ~~Bug #1765017~~ +++

We noticed this issue when gluster volume heal <volname> info fails on one of the bricks because gf_event() is taking ~25 seconds to do getaddrinfo. Here is the relevant stack trace:

naddrs=naddrs@entry=0x7f8c3cfaa490) at ../sysdeps/posix/getaddrinfo.c:865
#8 0x00007f8c4976f2cd in __GI_getaddrinfo (name=0x7f8c4c6d0af0 "/var/run/glusterd.socket", service=service@entry=0x0, hints=hints@entry=0x7f8c3cfaa640, pai=pai@entry=0x7f8c3cfaa608) at ../sysdeps/posix/getaddrinfo.c:2419
#9 0x00007f8c4beb3366 in _gf_event (event=event@entry=EVENT_AFR_SUBVOL_UP, fmt=fmt@entry=0x7f8c2eead638 "subvol=%s") at events.c:69
#10 0x00007f8c2ee94ba4 in __afr_handle_child_up_event (this=this@entry=0x7f8c3000f070, child_xlator=child_xlator@entry=0x7f8c3000a630, idx=idx@entry=1, halo_max_latency_msec=halo_max_latency_msec@entry=0, event=event@entry=0x7f8c3cfaa82c, call_psh=call_psh@entry=0x7f8c3cfaa838, up_child=up_child@entry=0x7f8c3cfaa83c)

In glfsheal gf_event should happen on localhost. This bz tracks this bug.

--- Additional comment from Worker Ant on 2019-10-24 07:36:12 UTC ---

REVIEW: https://review.gluster.org/23606 (gf-event: Handle unix volfile-servers) posted (#1) for review on master by Pranith Kumar Karampuri

--- Additional comment from Worker Ant on 2019-10-30 05:57:23 UTC ---

REVIEW: https://review.gluster.org/23606 (gf-event: Handle unix volfile-servers) merged (#2) on master by Pranith Kumar Karampuri

Change 23606 - Merged

Change 23606 - **Merged**

```
gf-event: Handle unix volfile-servers

Problem:
glfsheal program uses unix-socket-based volfile server.
volfile server will be the path to socket in this case.
gf_event expects this to be hostname in all cases. So getaddrinfo
will fail on the unix-socket path, events won't be sent in this case.

Fix:
In case of unix sockets, default to localhost

fixes: bz#1765017
Change-Id: I60d27608792c29d83fb82beb5fde5ef4754bece8
Signed-off-by: Pranith Kumar K <pkarampu@redhat.com>
```

打patch: 由于libglustefs/src/events.c每次提交并不是单独提交，没法git patch-format和git am，最后将v7.7版本的整个文件同步到qingstor的glusterfs-3.12.14.qs的分支上

```
commit 81c1b4d3c87c3bdbb69abab55892cdbbeeec2208 (HEAD -> v3.12-qs, origin/v3.12-qs)
Author: wenwenxiao <wenwenxiao@yunify.com>
Date:   Fri Nov 20 00:34:25 2020 +0800

    add more log for ec_up and ec_down

commit b0a573e6e0915c86466050f92740093c6dc3ed90
Author: wenwenxiao <wenwenxiao@yunify.com>
Date:   Thu Nov 19 12:12:14 2020 +0800

    the cmd of "gluster vol heal info vol0" will be blocked so long time

    ec_up()->gf_event will be blocked in getaddrinfo

    the commit f4f24c8c782bf4fa601f7ef14bbf2e2b6583cd90 will reslove the problem

    related commit:

commit 670c6c229a1ca4d457d613d1ae0ac1948800b788
Author: Subha sree Mohankumar <smohanku@redhat.com>
Date:   Sat Nov 11 00:12:10 2017 +0530

    libglusterfs:UNUSED VALUE coverity fix

    Problem : Overwriting previous write to "ret" with value "EVENT_SEND_OK"
    before itr can be used.

    Fix: The value of ret is used in out.

    Change-Id: I2cdb32e441c85c94de30de89a7a4121fd54d1acd
    BUG: 789278
    Signed-off-by: Subha sree Mohankumar <smohanku@redhat.com>
```

qingstor的glusterfs-3.12.14.qs的分支上的这个commit b0a573e6e0915c86466050f92740093c6dc3ed90是解决问题的commit，

open source的glusterfs v7.7的这个commit f4f24c8c782bf4fa601f7ef14bbf2e2b6583cd90可以解决问题

getaddrinfo工作原理分析

实例分析过程

要想知道getaddrinfo是如何查询信息的，可以用strace工具，追踪getaddrinfo函数 在执行时打开了哪些文件。

利用**APUE 第三版 图16-9**的程序，来分析

```
$ gcc 16-9.c -o getaddr
$ strace -e trace=file -o file ./getaddr google.com http
$ cat file
```

截取file中与本次目的相关输出

```
open("/etc/nsswitch.conf", O_RDONLY|O_CLOEXEC) = 3
open("/etc/services", O_RDONLY|O_CLOEXEC) = 3
open("/etc/host.conf", O_RDONLY|O_CLOEXEC) = 3
open("/etc/resolv.conf", O_RDONLY|O_CLOEXEC) = 3
open("/etc/hosts", O_RDONLY|O_CLOEXEC) = 3
```

现在来逐个分析这些文件

/etc/services

该文件是记录网络服务名和它们对应使用的端口号及协议。

/etc/host.conf