fs - Feature #36585

allow nfs-ganesha to export named cephfs filesystems

10/24/2018 11:46 AM - Jeff Layton

Status:	Resolved	**Start date:**	10/24/2018
Priority:	High	**Due date:**	
Assignee:	Jeff Layton	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	v14.0.0		
Source:	Development	**Affected Versions:**	
Tags:		**Component(FS):**	Ganesha FSAL, libcephfs
Backport:		**Labels (FS):**	multifs
Reviewed:		**Pull request ID:**	

Description

Recently, libcephfs grew a new ceph_select_filesystem call that allows the caller to select a particular filesystem to mount prior to issuing the ceph_mount call. Add a new parameter to nfs-ganesha's FSAL_CEPH export config block, that allows the admin to specify the fsname to mount.

This however exposes a second issue. We currently generate filehandles in FSAL_CEPH using a snapid+inode number tuple. With multiple filesystems, this is no longer sufficient to uniquely ID the inode. We need to be able to specify the filesystem too.

What we'll probably use is the fscid that is tracked in the MDS. It's a 32-bit value that should be unique, even if the filesystem is destroyed and recreated. We should be able to mix that in with filehandle to generate enough uniqueness.

History

#1 - 10/24/2018 01:48 PM - Jeff Layton

I may have misinterpreted the problems I was seeing while attempting this yesterday. Ganesha actually embeds the export id inside the filehandle so there should be no collisions. I am seeing some crashes down in libcephfs though when attempting to access multiple filesystems via the same ganesha daemon. Here's one:

```
#0  0x00007ffff540ceab in raise () from /lib64/libc.so.6
#1  0x00007ffff53f75b9 in abort () from /lib64/libc.so.6
#2  0x00007fffeb4f0ee9 in ceph::__ceph_assert_fail (assertion=<optimized out>, file=<optimized out>, line=98,
func=0x7ffff5adf050 <Cond::Signal()::__PRETTY_FUNCTION__> "int Cond::Signal()")
    at /usr/src/debug/ceph-14.0.0-4481.g1cebaacc8f5b.fc28.x86_64/src/common/assert.cc:74
#3  0x00007fffeb4f10b4 in ceph::__ceph_assert_fail (ctx=...) at /usr/src/debug/ceph-14.0.0-4481.g1cebaacc8f5b.
fc28.x86_64/src/common/assert.cc:79
#4  0x00007fffe01d7210 in Cond::Signal (this=<optimized out>) at /usr/src/debug/ceph-14.0.0-4481.g1cebaacc8f5b
.fc28.x86_64/src/common/Cond.h:101
#5  Client::signal_cond_list (this=this@entry=0xb3d1c0, ls=std::__cxx11::list = {...}) at /usr/src/debug/ceph-
14.0.0-4481.g1cebaacc8f5b.fc28.x86_64/src/client/Client.cc:3717
#6  0x00007fffe01d74d5 in Client::wake_inode_waiters (this=this@entry=0xb3d1c0, s=s@entry=0x8377b8) at /usr/sr
c/debug/ceph-14.0.0-4481.g1cebaacc8f5b.fc28.x86_64/src/client/Client.cc:3741
#7  0x00007fffe02187e4 in Client::handle_client_session(MClientSession*) () at /usr/src/debug/ceph-14.0.0-4481
.g1cebaacc8f5b.fc28.x86_64/src/client/Client.cc:2098
#8  0x00007fffe024fcb7 in Client::ms_dispatch(Message*) () at /usr/src/debug/ceph-14.0.0-4481.g1cebaacc8f5b.fc
28.x86_64/src/client/Client.cc:2542
#9  0x00007fffe028014a in Dispatcher::ms_dispatch2 (this=0xb3d1c0, m=...) at /usr/src/debug/ceph-14.0.0-4481.g
1cebaacc8f5b.fc28.x86_64/src/msg/Dispatcher.h:126
#10 0x00007fffeb7485aa in Messenger::ms_deliver_dispatch (m=..., this=0xb31aa0) at /usr/src/debug/ceph-14.0.0-
4481.g1cebaacc8f5b.fc28.x86_64/src/msg/DispatchQueue.cc:197
#11 DispatchQueue::entry() () at /usr/src/debug/ceph-14.0.0-4481.g1cebaacc8f5b.fc28.x86_64/src/msg/DispatchQue
ue.cc:196
#12 0x00007fffeb7fc411 in DispatchQueue::DispatchThread::entry (this=<optimized out>) at /usr/src/debug/ceph-1
4.0.0-4481.g1cebaacc8f5b.fc28.x86_64/src/msg/DispatchQueue.h:102
#13 0x00007ffff5f41594 in start_thread () from /lib64/libpthread.so.0
#14 0x00007ffff54cfe6f in clone () from /lib64/libc.so.6
```

The assertion is inside Signal:

```
int Signal() {
  // make sure signaler is holding the waiter's lock.
  ceph_assert(waiter_mutex == NULL ||
         waiter_mutex->is_locked());

  int r = pthread_cond_broadcast(&_c);
  return r;
}
```

I'm going to see if I can simulate this with a cephfs testcase, and get ganesha out of the way.

#2 - 10/24/2018 08:36 PM - Jeff Layton

Nope, I was right the first time. The issue was with filehandle hash key collisions in ganesha due to the fact that they were being generated from the vinodeno_t.

#3 - 10/26/2018 12:20 PM - Jeff Layton

New ceph interface here. I also have a set of ganesha patches that will use this to generate filehandles when it's dealing with a named filesystem:

https://github.com/ceph/ceph/pull/24747

#4 - 11/01/2018 09:54 PM - Patrick Donnelly

- Status changed from New to Resolved

- Target version set to v14.0.0

- Source set to Development