fs - Bug #27051

client: cannot list out files created by another ceph-fuse client

08/22/2018 01:22 PM - Peng Xie

Status:	Resolved	**Start date:**	08/22/2018
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Correctness/Safety	**Estimated time:**	0.00 hour
Target version:	v14.0.0		
Source:	Community (dev)	**Affected Versions:**	v14.0.0
Tags:		**ceph-qa-suite:**	fs
Backport:	mimic,luminous	**Component(FS):**	Client
Regression:	No	**Labels (FS):**	
Severity:	2 - major	**Pull request ID:**	
Reviewed:			

Description

recently, in our cephfs (ceph-fuse client) online production environment, i found several rmdir fail due to not empty directory. after further dig into, the problem is one client can not list the files
created by other client.

our code has incorporated the fix https://github.com/ceph/ceph/pull/21712 , but it still happend due to other reasons.

after analysing the log , here is my step to reproduce the problem, the root cause of the problem is also attached below as well as my proposes fix pull request.

reproduce steps (ceph-fuse mount) :
a. there are two clients: client1, client2. first mount client1 under /mnt/peng1/, which is empty
b. run 'mkdir -p /mnt/peng1/yunfei/god/' under client1
c. mount client2 under path /mnt/peng2
d. run 'touch /mnt/peng2/yunfei/god' through client2, and the run 'stat /mnt/peng2/yunfei/god' under the same client2
e. gdb attach client2 ceph-fuse daemon, let mds find out 'new stale session' for client2. after mds log print "new stale session", and gdb continue and restore client2 session
f. run my own written c program (listed below), which will create a new file "pengsynctest" under /mnt/peng1/yunfei/god/ through client1, and lookup "/mnt/peng2/yunfei/god" through client2.
g. finally , you will never list out the "pengsynctest" under /mnt/peng2/yunfei/god through client2

step-f 's c program:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

#include <memory.h>
#include <fcntl.h>
#include <sys/types.h>

int main(void) {
int fd;
int ret = 0;
struct stat buf1;
fd = open("/mnt/peng1/yunfei/god/pengsynctest", O_RDWR|O_CREAT);
ret = stat("/mnt/peng2/yunfei/god", buf1);
if (ret >= 0)
printf("stat file size1 %d\n", buf1.st_size);
printf("will sleep");
sleep(300000);
```

close(fd);
exit(0);
}

according to the above reproduce step, root cause is :
after step-d, client2 will bring back pAsLsXsFs for dir "god". in step-2, during client2 session stale,
in the client2 side, cap for "god" still pAsLsXsFs, but not valid. in the mds side, the
corresponding stale cap for client2 will be revoked from pAsLsXsFs to only P.
however, after the stale session restored, after the step-f 's create "god/pengsynctest" from
client1 (early reply, journal has not been persisted), the following step-f's client2 lookup
'god/pengsynctest' will bring back pAsLsXs and 0 numfiles under "god"
(since client1's create op write journal has not been completed).
finally, during, client2 handle the lookup "god" reply, in its add_update_cap(), it merge the reply issued
cap (pAsLsXs) with the invalid, out-dataed pFsLsXsFs and finally issue client2 pFsLsXsFs by mistake.

the detailed log pasted below:

client.745629 log :

2018-08-16 12:30:53.336204 7fa4b1fb7700 3 client.745629 ll_lookup 0xc6c4300 god
....
2018-08-16 12:30:53.336214 7fa4b1fb7700 10 client.745629 _do_lookup on #10000032ea8/god
....
2018-08-16 12:30:53.336240 7fa4b1fb7700 10 client.745629 send_request client_request(unknown.0:8 lookup #10000032ea8/god
2018-08-16 12:30:53.336215) v3 to mds.0
2018-08-16 12:30:53.336243 7fa4b1fb7700 1 -- 192.168.61.100:0/3244889987 --> 192.168.61.100:6808/12461 --
client_request(client.745629:8 lookup #10000032ea8/god 2018-08-16 12:30:53.336215) v3 -- ?+0 0xc7662c0 con 0xc658780
<<--- send to mds ....

mds.log :

<<<---mds received the lookup
2018-08-16 12:30:53.336437 7fdf9d75f700 1 -- 192.168.61.100:6808/12461 <== client.745629 192.168.61.100:0/3244889987 29
==== client_request(client.745629:8 lookup #10000032ea8/god 2018-08-16 12:30:53.336215) v3 ==== 125+0+0 (3252943613 0 0)
0xca34c00 con 0xc270900

2018-08-16 12:30:53.336480 7fdf9d75f700 20 mds.0.server handle_client_request client_request(client.745629:8 lookup
#10000032ea8/god 2018-08-16 12:30:53.336215) v3

2018-08-16 12:30:53.336512 7fdf9d75f700 10 mds.0.server rdlock_path_pin_ref request(client.745629:8 cr=0xca34c00)
#10000032ea8/god
2018-08-16 12:30:53.336524 7fdf9d75f700 7 mds.0.cache traverse: opening base ino 10000032ea8 snap head
2018-08-16 12:30:53.336527 7fdf9d75f700 12 mds.0.cache traverse: path seg depth 0 'god' snapid head
2018-08-16 12:30:53.336530 7fdf9d75f700 20 mds.0.cache.dir(10000032ea8) lookup (head, 'god')
2018-08-16 12:30:53.336533 7fdf9d75f700 20 mds.0.cache.dir(10000032ea8) hit ~~> (god,head)~~
~~2018-08-16 12:30:53.336536 7fdf9d75f700 10 mds.0.cache path_traverse finish on snapid head~~
~~2018-08-16 12:30:53.336537 7fdf9d75f700 10 mds.0.server ref is [inode 10000032eb3 [...2,head] /yunfei/god/ auth v45 pv47 ap=1+1~~
~~dirtyparent f() n(v0 1=0+1) (inest lock w=1) (ifile lock w=1) (iversion lock w=1 last_client=745628) caps={745628=pAsLsXs/~~
@8,745629=p/-@4} | request=0 lock=3 dirfrag=1 caps=1 dirtyparent=1 dirty=1 waiter=0 authpin=1 0xd284d20 mode: 16877]
<<<--- "ifile lock w=1" means the previous create god/pengsynctest op writing to journal not come
back yet .
....
2018-08-16 12:30:53.336772 7fdf9d75f700 7 mds.0.locker rdlock_start on (dn sync) on [dentry #1/yunfei/god [2,head] auth (dversion
lock) pv=47 v=45 ap=1+3 inode=0xd284d20 | request=0 lock=0 inodepin=1 dirty=1 authpin=1 0xd52c6a0] <<<--- lookup will not
touch god ifile lock
2018-08-16 12:30:53.336778 7fdf9d75f700 10 mds.0.locker got rdlock on (dn sync r=1) [dentry #1/yunfei/god [2,head] auth (dn sync
r=1) (dversion lock) pv=47 v=45 ap=1+3 inode=0xd284d20 | request=0 lock=1 inodepin=1 dirty=1 authpin=1 0xd52c6a0]
....
2018-08-16 12:30:53.336887 7fdf9d75f700 10 mds.0.server reply to stat on client_request(client.745629:8 lookup
#10000032ea8/god 2018-08-16 12:30:53.336215) v3 ref: [inode 10000032eb3 [...2,head] /yunfei/god/ auth v45 pv47 ap=2+1
dirtyparent f() n(v0 1=0+1) (isnap sync r=1) (inest lock w=1) (ifile lock w=1) (iversion lock w=1 last_client=745628)
caps={745628=pAsLsXs/-@8,745629=p/-@4} | request=1 lock=4 dirfrag=1 caps=1 dirtyparent=1 dirty=1 waiter=0 authpin=1
0xd284d20 mode: 16877]
2018-08-16 12:30:53.336897 7fdf9d75f700 10 mds.0.server reply_client_request 0 ((0) Success) client_request(client.745629:8
lookup #10000032ea8/god 2018-08-16 12:30:53.336215) v3
2018-08-16 12:30:53.336916 7fdf9d75f700 10 mds.0.server apply_allocated_inos 0 / [] / 0
2018-08-16 12:30:53.336918 7fdf9d75f700 20 mds.0.server lat 0.000596

2018-08-16 12:30:53.336922 7fdf9d75f700 20 mds.0.server set_trace_dist snapid head
2018-08-16 12:30:53.336925 7fdf9d75f700 10 mds.0.server set_trace_dist snaprealm snaprealm(1 seq 1 lc 0 cr 0 cps 1 snaps={} 0xc1ab840) len=48
2018-08-16 12:30:53.336928 7fdf9d75f700 20 mds.0.cache.ino(10000032ea8) pfile 0 pauth 0 plink 0 pxattr 0 plocal 0 ctime 2018-08-16 12:24:36.135012 valid=1
2018-08-16 12:30:53.336941 7fdf9d75f700 10 mds.0.cache.ino(10000032ea8) encode_inodestat issuing pAsLsXsFs seq 5
2018-08-16 12:30:53.336943 7fdf9d75f700 10 mds.0.cache.ino(10000032ea8) encode_inodestat caps pAsLsXsFs seq 5 mseq 0 xattrv 0 len 0
2018-08-16 12:30:53.336951 7fdf9d75f700 20 mds.0.server set_trace_dist added diri [inode 10000032ea8 [...2,head] /yunfei/ auth v2488 pv2490 ap=2+1 dirtyparent f(v0 m2018-08-16 12:24:36.135012 6=0+6) n(v3 rc2018-08-16 12:24:36.135012 11=4+7) (isnap sync r=1) (inest lock w=1) (iversion lock w=1 last_client=745628) caps={745628=pAsLsXsFs/-@3,745629=pAsLsXsFs/-@5} | request=0 lock=3 dirfrag=1 caps=1 dirtyparent=1 dirty=1 authpin=1 0xd27aa60 mode: 16877]
2018-08-16 12:30:53.336962 7fdf9d75f700 20 mds.0.server set_trace_dist added dir [dir 10000032ea8 /yunfei/ [2,head] auth pv=48 v=46 cv=0/0 ap=1+3+4 state=1610612738|complete f(v0 m2018-08-16 12:24:36.135012 6=0+6) n(v3 rc2018-08-16 12:24:36.135012 10=4+6) hs=6+0,ss=0+0 dirty=6 | child=1 dirty=1 authpin=1 0xc879b00]
2018-08-16 12:30:53.336968 7fdf9d75f700 20 mds.0.locker issue_client_lease no/null lease on [dentry #1/yunfei/god [2,head] auth (dn sync r=1) (dversion lock) pv=47 v=45 ap=1+3 inode=0xd284d20 | request=0 lock=1 inodepin=1 dirty=1 authpin=1 0xd52c6a0]
2018-08-16 12:30:53.336983 7fdf9d75f700 20 mds.0.server set_trace_dist added dn head [dentry #1/yunfei/god [2,head] auth (dn sync r=1) (dversion lock) pv=47 v=45 ap=1+3 inode=0xd284d20 | request=0 lock=1 inodepin=1 dirty=1 authpin=1 0xd52c6a0]
2018-08-16 12:30:53.336989 7fdf9d75f700 20 mds.0.cache.ino(10000032eb3) pfile 0 pauth 0 plink 0 pxattr 0 plocal 0 ctime 2018-08-16 12:29:17.739113 valid=1
<<<--- pfile 0 means choose inode itself rather than projected reply to clients
2018-08-16 12:30:53.336992 7fdf9d75f700 10 mds.0.cache.ino(10000032eb3) encode_inodestat issuing pAsLsXs seq 5 <<--- allow pAsLsXs, no Fs since the previous write journal not comback
2018-08-16 12:30:53.336998 7fdf9d75f700 10 mds.0.cache.ino(10000032eb3) encode_inodestat caps pAsLsXs seq 5 mseq 0 xattrv 0 len 0
2018-08-16 12:30:53.337010 7fdf9d75f700 20 mds.0.server set_trace_dist added in [inode 10000032eb3 [...2,head] /yunfei/god/ auth v45 pv47 ap=2+1 dirtyparent f() n(v0 1=0+1) (isnap sync r=1) (inest lock w=1) (ifile lock w=1) (iversion lock w=1 last_client=745628) caps={745628=pAsLsXs/-@8,745629=pAsLsXs/-@5} | request=1 lock=4 dirfrag=1 caps=1 dirtyparent=1 dirty=1 waiter=0 authpin=1 0xd284d20 mode: 16877]
2018-08-16 12:30:53.337019 7fdf9d75f700 1 -- 192.168.61.100:6808/12461 --> 192.168.61.100:0/3244889987 -- client_reply(???:8 = 0 (0) Success) v1 -- ?+0 0xc9f6dc0 con 0xc270900
<<---

Client.745629 log:
...
2018-08-16 12:30:53.337135 7fa4b6ac3700 1 -- 192.168.61.100:0/3244889987 <== mds.0 192.168.61.100:6808/12461 31 ==== client_reply(???:8 = 0 (0) Success) v1 ==== 654+0+0 (3948990430 0 0) 0xc7662c0 con 0xc658780
2018-08-16 12:30:53.337150 7fa4b6ac3700 20 client.745629 handle_client_reply got a reply. Safe:1 tid 8
2018-08-16 12:30:53.337151 7fa4b6ac3700 10 client.745629 insert_trace from 2018-08-16 12:30:53.336238 mds.0 is_target=1 is_dentry=1
2018-08-16 12:30:53.337154 7fa4b6ac3700 10 client.745629 features 0x7fffffefdfbfff
2018-08-16 12:30:53.337155 7fa4b6ac3700 10 client.745629 update_snap_trace len 48
2018-08-16 12:30:53.337157 7fa4b6ac3700 20 client.745629 get_snap_realm 1 0xc6ba0d0 3 ~~> 4~~
~~2018-08-16 12:30:53.337159 7fa4b6ac3700 10 client.745629 update_snap_trace snaprealm(1 nref=4 c=0 seq=1 parent=0 my_snaps=[] cached_snapc=1=[]) seq 1 <= 1 and same parent, SKIPPING~~
~~2018-08-16 12:30:53.337161 7fa4b6ac3700 20 client.745629 put_snap_realm 1 0xc6ba0d0 4 -> 3~~
~~2018-08-16 12:30:53.337162 7fa4b6ac3700 10 client.745629 hrm is_target=1 is_dentry=1~~
~~2018-08-16 12:30:53.337171 7fa4b6ac3700 12 client.745629 add_update_inode had 10000032eb3.head(faked_ino=0 ref=2 ll_ref=3 cap_refs={} open={} mode=40755 size=0/0 mtime=2018-08-16 12:29:17.738215 caps=~~(0=pAsLsXsFs) COMPLETE parents=0xc675340 0xc6c4800) caps pAsLsXs <<-- since session stale happened , so the client side caps pAsLsXsFs is not valid .
2018-08-16 12:30:53.337178 7fa4b6ac3700 20 client.745629 dir hash is 2
2018-08-16 12:30:53.337179 7fa4b6ac3700 10 client.745629 update_inode_file_bits 10000032eb3.head(faked_ino=0 ref=2 ll_ref=3 cap_refs={} open={} mode=40755 size=0/0 mtime=2018-08-16 12:29:17.738215 caps=-(0=pAsLsXsFs) COMPLETE parents=0xc675340 0xc6c4800) - mtime 2018-08-16 12:29:17.738215
<<<<--- since mds issue pAsLsXs, the function check_cap_issue() will not clear the COMPLETE flag here,
2018-08-16 12:30:53.337187 7fa4b6ac3700 10 client.745629 add_update_cap issued pAsLsXsFs ~~> pAsLsXsFs from mds.0 on 10000032eb3.head(faked_ino=0 ref=2 ll_ref=3 cap_refs={} open={} mode=40755 size=0/0 mtime=2018-08-16 12:29:17.738215 caps=pAsLsXsFs(0=pAsLsXsFs) COMPLETE parents=0xc675340 0xc6c4800)~~
~~<<<<~~ the problem happened here, take a look at the code:
void Client::add_update_cap(...) {
.....
cap->issued |= issued; //<<<-- finally cap->issued = pAsLsXsFs .
cap->implemented |= issued;
cap->seq = seq;
cap->issue_seq = seq;

cap->mseq = mseq;
cap->gen = mds_session->cap_gen; <<--- make the cap->issued valid .
}

2018-08-16 12:30:53.337193 7fa4b6ac3700 12 client.745629 add_update_inode had 10000032ea8.head(faked_ino=0 ref=4 ll_ref=5 cap_refs={} open={} mode=40755 size=0/0 mtime=2018-08-16 12:24:36.135012 caps=pAsLsXsFs(0=pAsLsXsFs) parents=0xc675030 0xc6c4300) caps pAsLsXsFs
2018-08-16 12:30:53.337197 7fa4b6ac3700 20 client.745629 dir hash is 2
.....
.....
<<--- the following readdir arrived :

2018-08-16 12:31:12.506072 7fa4b27b8700 10 client.745629 readdir_r_cb 10000032eb3.head(faked_ino=0 ref=3 ll_ref=6 cap_refs={} open={} mode=40755 size=0/0 mtime=2018-08-16 12:29:17.738215 caps=pAsLsXsFs(0=pAsLsXsFs) COMPLETE parents=0xc675340 0xc6c4800) offset 0 at_end=0 hash_order=0
<<<-- pay attention that the mds does not issue Fs to the client .
2018-08-16 12:31:12.506091 7fa4b27b8700 15 client.745629 including .
2018-08-16 12:31:12.506092 7fa4b27b8700 10 client.745629 fill_stat on 10000032eb3 snap/devhead mode 040755 mtime 2018-08-16 12:29:17.738215 ctime 2018-08-16 12:29:17.739113
2018-08-16 12:31:12.506106 7fa4b27b8700 10 client.745629 fill_dirent '.' -> 10000032eb3 type 4 w/ next_off 1
2018-08-16 12:31:12.506110 7fa4b27b8700 15 client.745629 including ..
2018-08-16 12:31:12.506121 7fa4b27b8700 10 client.745629 fill_stat on 10000032eb3 snap/devhead mode 040755 mtime 2018-08-16 12:29:17.738215 ctime 2018-08-16 12:29:17.739113
2018-08-16 12:31:12.506123 7fa4b27b8700 10 client.745629 fill_dirent '..' -> 10000032eb3 type 4 w/ next_off 2
2018-08-16 12:31:12.506124 7fa4b27b8700 10 client.745629 offset 2 snapid head (complete && ordered) 1 issued pAsLsXsFs
2018-08-16 12:31:12.506128 7fa4b27b8700 10 client.745629 _readdir_cache_cb 0xc689200 on 10000032eb3 last_name offset 2
2018-08-16 12:31:12.506129 7fa4b27b8700 10 client.745629 dir is empty .
<<<---since the cap issued was set pAsLsXsFs by mistake, and the dir "god" inode flag
is completed and ordered, it will try to read from the readdir cache
int Client::_readdir_cache_cb() {
// at the very beginning
Dir *dir = dirp->inode->dir;
if (!dir) {
ldout(cct, 10)<<" dir is empty " << dendl; <<-- no chance to fetch dentries from mds
dir->set_end();
return 0;
}
.....
}
2018-08-16 12:31:12.506146 7fa4b1fb7700 3 client.745629 seekdir(0xc689200, 2)
2018-08-16 12:31:12.506159 7fa4b1fb7700 10 client.745629 readdir_r_cb 10000032eb3.head(faked_ino=0 ref=3 ll_ref=6 cap_refs={} open={} mode=40755 size=0/0 mtime=2018-08-16 12:29:17.738215 caps=pAsLsXsFs(0=pAsLsXsFs) COMPLETE parents=0xc675340 0xc6c4800) offset 2 at_end=0 hash_order=0

Related issues:

Copied to fs - Backport #35933: luminous: client: cannot list out files creat...	**Resolved**
Copied to fs - Backport #35934: mimic: client: cannot list out files created ...	**Resolved**

History

#1 - 08/22/2018 01:33 PM - Peng Xie

my fix pull request is
https://github.com/ceph/ceph/pull/23691

#2 - 08/23/2018 09:12 PM - Nathan Cutler

- Project changed from Ceph to fs

#3 - 08/23/2018 09:15 PM - Nathan Cutler

- Status changed from New to Need Review

#4 - 08/23/2018 11:30 PM - Patrick Donnelly

- Subject changed from cephfs ceph-fuse client cannot list out files created by another ceph-fuse client to client: cannot list out files created by another ceph-fuse client

- Category set to Correctness/Safety

- Source set to Community (dev)

- Backport set to mimic,luminous

- Component(FS) Client added

#5 - 09/11/2018 06:26 PM - Patrick Donnelly

- Status changed from Need Review to Pending Backport

#6 - 09/11/2018 07:15 PM - Nathan Cutler

- Copied to Backport #35933: luminous: client: cannot list out files created by another ceph-fuse client added

#7 - 09/11/2018 07:15 PM - Nathan Cutler

- Copied to Backport #35934: mimic: client: cannot list out files created by another ceph-fuse client added

#8 - 10/19/2018 09:46 PM - Nathan Cutler

- Status changed from Pending Backport to Resolved