RADOS - Bug #20684

pg refs leaked when osd shutdown

07/19/2017 03:08 AM - Honggang Yang

Status:	Resolved	**Start date:**	07/14/2017
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Source:		**Reviewed:**	
Tags:		**Affected Versions:**	v10.2.2
Backport:		**ceph-qa-suite:**	
Regression:	No	**Component(RADOS):**	OSD
Severity:	1 - critical	**Pull request ID:**	

Description

1. summary

When kicking a pg, its ref count is great than 1, this cause assert failed.

When osd is in process of shutdown, sparse-read ops are dequeued.
The corresponding obc is not found in object_context. So a obc
is created from disk. Then a OpContext is created with this obc.
And MOSDECSubOpReads requests are sent to peers.

When shutdown process reached ReplicatedPG::on_shutdown's
pgbackend->on_change() line, these read ops are cancelled, but
OpContext are not cleared.

Then when the related pg is removed, it ref count is 2, and assert failed.

```
// bt info
    6895 2017-07-07 14:00:46.039360 7f84994f0700 20 osd.4 39610  kicking pg 69
.es0
    6896 2017-07-07 14:00:46.039369 7f84994f0700 30 osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0
'0,39606'89] local-les=39523 n=2 ec=38641 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1
]       r=0 lpr=39519 luod=0'0 crt=39489'84 lcod 3
    6897 2017-07-07 14:00:46.039383 7f84994f0700 -1 osd.4 39610 pgid 69.es0 has ref count of 2

    ...

    6930 2017-07-07 14:00:46.063526 7f84994f0700 -1 osd/OSD.cc: In function 'int OSD::shutdown()'
thread 7f84994f0700 time 2017-07-07 14:00:46.039424
    6931 osd/OSD.cc: 2728: FAILED assert(0
)
    6932

    6933  ceph version 10.2.2-33-g353eb22 (353
eb2284b024c6d024667bc9eb04e7ebc06c293)
    6934  1: (ceph::__ceph_assert_fail(char const*, char const*, int, char const*)+0x95) [
0x7f84c42899ad]
    6935  2: (OSD::shutdown()+0x21c2) [0x7f84c38b1716
]
    6936  3: (OSD::handle_signal(int)+0x1aa) [0x7f84c38a6dfe
]
    6937  4: (handle_osd_signal(int)+0x2b) [0x7f84c388728a
]
    6938  5: (SignalHandler::entry()+0x23e) [0x7f84c40dae9a
```

```
]
    6939  6: (Thread::entry_wrapper()+0xc1) [0x7f84c426a065
]
    6940  7: (Thread::_entry_func(void*)+0x18) [0x7f84c4269f9a
]
    6941  8: (()+0x7dc5) [0x7f84c0be5dc5
]
    6942  9: (clone()+0x6d) [0x7f84bef6e28d
]
    6943  NOTE: a copy of the executable, or `objdump -rdS <executable>` is needed to interpret
this.
```

2. figure out unreleased pg reference

My ceph version 10.2.2 and with PG_DEBUG_REFS enabled.

In osd.4's log, it dumps which ref of pg is not released.

```
/// ref id is 2515
    6895 2017-07-07 14:00:46.039360 7f84994f0700 20 osd.4 39610  kicking pg 69
.es0
    6896 2017-07-07 14:00:46.039369 7f84994f0700 30 osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0
'0,39606'89] local-les=39523 n=2 ec=38641 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1
]       r=0 lpr=39519 luod=0'0 crt=39489'84 lcod 3
    6897 2017-07-07 14:00:46.039383 7f84994f0700 -1 osd.4 39610 pgid 69.es0 has ref count of 2

    6898 2017-07-07 14:00:46.039386 7f84994f0700  0 osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0
'0,39606'89] local-les=39523 n=2 ec=38641 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1
]       r=0 lpr=39519 luod=0'0 crt=39489'84 lcod 3
    6899 2017-07-07 14:00:46.039395 7f84994f0700  0 osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0
'0,39606'89] local-les=39523 n=2 ec=38641 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1
]       r=0 lpr=39519 luod=0'0 crt=39489'84 lcod 3
    6900  1: (ceph::BackTrace::BackTrace(int)+0x2d) [0x7f84c3a971db
]
    6901  2: (PG::get_with_id()+0xa2) [0x7f84c3a4bdac
]
    6902  3: (get_with_id(ReplicatedPG*)+0x18) [0x7f84c3bb7b72
]
    6903  4: (TrackedIntPtr<ReplicatedPG>::TrackedIntPtr(ReplicatedPG*)+0x2e) [0x7f84c3bd75da
]
    6904  5: (ReplicatedPG::C_PG_ObjectContext::C_PG_ObjectContext(ReplicatedPG*, ObjectContext*)+
0x4a) [0x7f84c3bcba58]
    6905  6: (ReplicatedPG::get_object_context(hobject_t const&, bool, std::map<std::string
, ceph::buffer::list, std::less<std::string>, std::allocator<std::pair<std::string const
, ceph::bu      ffer::list> > >*)+0xbe6) [0x7f84c3b8a59a]
    6906  7
: (ReplicatedPG::find_object_context(hobject_t const&, std::shared_ptr<ObjectContext>*, bool, bool
, hobject_t*)+0xde) [0x7f84c3b8af5e]
    6907  8: (ReplicatedPG::do_op(std::shared_ptr<OpRequest>&)+0x27a6) [0x7f84c3b41ae6
]
    6908  9: (ReplicatedPG::do_request(std::shared_ptr<OpRequest>&, ThreadPool::TPHandle&)+0x98b
) [0x7f84c3b3ea89]
    6909  10
: (OSD::dequeue_op(TrackedIntPtr<PG>, std::shared_ptr<OpRequest>, ThreadPool::TPHandle&)+0x4f0) [
0x7f84c38ecd4c]
    6910  11: (PGQueueable::RunVis::operator()(std::shared_ptr<OpRequest>&)+0x6c) [0x7f84c3894c16
]
```

After checked the log, I find this stack is generated when service client request 'client.88411.0:946'.

Because obc is not found in cache, a new one is created. And C_PG_ObjectContext add pg's reference.
Then two MOSDECSubOpRead requests are sent to peers. But these two ops are cancelled in pg 69.es0's ECBackend::on_change function. But its obc is not cleared. So the pg reference is not decreased.

3. whole story

```
    ###
: OSD shutdown
    @@@: client request 1

    $$$: client request 2

    &&&: another new
 request

    ---------------- copied from osd.4's
 log ordered by timestamp ----------------------------------
    ###1
. OSD shutdown begin
          osd.4 39610
 *** Got signal Terminated ***
          osd.4 39610
 prepare_to_stop telling mon we are shutting down
          osd.4 39610
 got_stop_ack starting shutdown
          osd.4 39610
 prepare_to_stop starting shutdown
          osd.4 39610
 shutdown

    @@@1. Client request#946
 dequeue begin
          osd.4 39610 dequeue_op 0x7f84d1849900 prio 63 cost 0 latency 0.848527 osd_op(client
.884116.0:946 69.1703329e (undecoded) ack+read+known_if_redirected e39609) v7 pg pg[69.es0( v
39606'89 (0'0,39606'89] local-les=39523 n=2 ec=386
          ...

          osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0'0,39606'89] local-les=39523 n=2 ec=38641
 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1]       r=0 lpr=39519 crt=39489'84 lcod 39598'86
 mlcod 39598'86 active+clean] do_op osd_op(c

    ###2. Kicking pg 69.4
s0 begin
          osd.4 39610  kicking pg 69.4
s0
          ...

          osd.4 pg_epoch: 39610 pg[69.4s0( v 39598'73 (0'0,39598'73] local-les=39398 n=3 ec=38641
 les/c/f 39398/39402/0 39390/39395/39395) [4,1,3]       r=0 lpr=39395 crt=39590'66 lcod 39598'71
 mlcod 39598'71 active+clean] on_shutdown

    @@@2. Client request#946
 dequeue done
          osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0'0,39606'89] local-les=39523 n=2 ec=38641
 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1]       r=0 lpr=39519 crt=39489'84 lcod 39598'86
 mlcod 39598'86 active+clean] get_object_con
          ...

          ******** Create a new obc and C_PG_ObjectContext
// (ref id is 2515) *********
          ...

          osd.4 39610 dequeue_op 0x7f84d1849900
 finish

    &&&1. A new request #948
```

```
 arrived
          10.0.0.186:6809/17396 <== client.884116 10.0.0.94:0/1528694612 205 ==== osd_op(client
.884116.0:948 69.1703329e (undecoded) ack+read+kn      own_if_redirected e39609) v7 ==== 198+0+0 (
1196714097 0 0) 0x7f84d0095c80 con 0x7f84cfa

    ###3
. Continue kicking pgs
          osd.4 39610  kicking pg 69.4
s0
          osd.4 39610  kicking pg 69.9
s0
          osd.4 39610  kicking pg 69
.bs2
          osd.4 39610  kicking pg 69
.fs2
          osd.4 39610  kicking pg 69
.es0

    ###4. Begin kicking 69
.es0
          osd.4 39610  kicking pg 69
.es0

    $$$1. client request #947
 handling
          osd.4 39610 dequeue_op 0x7f84d1849a00 prio 63 cost 0 latency 1.017226 osd_op(client
.884116.0:947 69.1703329e (undecoded) ack+read+known_i      f_redirected e39609) v7 pg pg[69
.es0( v 39606'89 (0'0,39606'89] local-les=39523 n=2
          ...

          osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0'0,39606'89] local-les=39523 n=2 ec=38641
 les/c/f 39523/39534/0 39514/39519/39401) [4,5,1]       r=0 lpr=39519 crt=39489'84 lcod 39598'86
 mlcod 39598'86 active+clean] get_object_con
          ...

          osd.4 39610 dequeue_op 0x7f84d1849a00
 finish
    ###5. Continue kicking 69
.es0
          pg[69
.es0(... lock
          pg[69
.es0(... on_shutdown
          pg[69
.es0(... requeue_ops
          pg[69
.es0(... cancel_copy_ops
          pg[69
.es0(... cancel_flush_ops
          pg[69
.es0(... cancel_proxy_ops
          pg[69
.es0(... on_change
          ************* Read ops are cancelled and
 obc is leaked! ********************
          osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0'0,39606'89
]... on_change: cancelling ReadOp(tid=61, to_read={69:794cc0e8:::yhg-116127-2
:head=read_request_t(to_read=[0,405504      ,0], need=4(0),5(1), want_attrs=0)}, complete={69
          osd.4 pg_epoch: 39610 pg[69.es0( v 39606'89 (0'0,39606'89
]... on_change: cancelling ReadOp(tid=62, to_read={69:794cc0e8:::yhg-116127-2
:head=read_request_t(to_read=[0,405504      ,0], need=4(0),5(1), want_attrs=0)}, complete={69
          ...

          pg[69
.es0... clear_primary_state
          pg[69
.es0... agent_stop
```

```
          pg[69
.es0... cancel_recovery
          pg[69
.es0... clear_recovery_state

    ###6
. pg ref assert failed
          osd.4 39610
 Store synced
          ...

          osd.4 39610  kicking pg 69
.es0
          osd.4 39610 pgid 69.es0 has ref count of 2
```

History

#1 - 07/19/2017 03:12 AM - Honggang Yang

https://github.com/ceph/ceph/pull/16408

#2 - 07/19/2017 03:11 PM - Sage Weil

- Status changed from New to Need Review

#3 - 07/19/2017 05:55 PM - Josh Durgin

Nice debugging and presentation of your analysis! That's my favorite kind of bug report!

#4 - 07/21/2017 04:43 PM - Yuri Weinstein

Honggang Yang wrote:

> https://github.com/ceph/ceph/pull/16408

merged

#5 - 07/21/2017 05:20 PM - Sage Weil

- Status changed from Need Review to Resolved