rbd - Bug #4827

librbd: use after free of ceph context or something in it

04/26/2013 09:21 AM - Josh Durgin

Status:	Resolved	**% Done:**	0%
Priority:	Urgent	**Spent time:**	0.00 hour
Assignee:	Josh Durgin		
Category:			
Target version:	v0.61 - Cuttlefish		
Source:	Q/A	**Reviewed:**	
Tags:		**Affected Versions:**	
Backport:		**ceph-qa-suite:**	
Regression:	No	**Pull request ID:**	
Severity:	3 - minor	**Crash signature:**	

Description

From teuthology:/a/teuthology-2013-04-26_02:29:00-rbd-next-testing-basic/1393/teuthology.log:

```
2013-04-26T03:26:37.988 INFO:teuthology.orchestra.run.out:4970 read     0x979e331 thru  0x97a2149
      (0x3e19 bytes)
2013-04-26T03:26:37.989 INFO:teuthology.orchestra.run.out:4972 clone    41 order 21 su 32768 sc 12
2013-04-26T03:26:40.144 INFO:teuthology.orchestra.run.out:checking clone #39, image image_client.0
-clone39 against file /home/ubuntu/cephtest/1393/archive/fsx-image_client.0-parent40
2013-04-26T03:27:12.305 INFO:teuthology.orchestra.run.err:common/Mutex.cc: In function 'Mutex::~Mu
tex()' thread 7f8d33ee1780 time 2013-04-26 03:27:14.526172
2013-04-26T03:27:12.305 INFO:teuthology.orchestra.run.err:common/Mutex.cc: 71: FAILED assert(nlock
 == 0)
2013-04-26T03:27:12.313 INFO:teuthology.orchestra.run.err: ceph version 0.60-666-ga5cade1 (a5cade1
fe7338602fb2bbfa867433d825f337c87)
2013-04-26T03:27:12.313 INFO:teuthology.orchestra.run.err: 1: (()+0x18f076) [0x7f8d32e4f076]
2013-04-26T03:27:12.313 INFO:teuthology.orchestra.run.err: 2: (librbd::ImageCtx::~ImageCtx()+0x133
) [0x7f8d33a6e093]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 3: (librbd::close_image(librbd::ImageCt
x*)+0x86) [0x7f8d33a7c036]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 4: (librbd::close_image(librbd::ImageCt
x*)+0x61) [0x7f8d33a7c011]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 5: (rbd_close()+0x9) [0x7f8d33a5bc59]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 6: (check_clone()+0x126) [0x404fa6]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 7: (do_clone()+0x1c5) [0x405265]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 8: (test()+0x375) [0x405915]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 9: (main()+0x7cf) [0x40318f]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 10: (__libc_start_main()+0xed) [0x7f8d3
262676d]
2013-04-26T03:27:12.314 INFO:teuthology.orchestra.run.err: 11: ceph_test_librbd_fsx() [0x403519]
```

History

#1 - 04/26/2013 03:44 PM - Josh Durgin

- Subject changed from librbd: mutex held during destruction to librbd: use after free of ceph context or something in it

Segfaults with different backtraces occurred with and without caching enabled. Unfortunately the first core file is corrupt, but the latter two show different signs of corruption (accessing an invalid ceph::log::SubsystemMap in one and accessing PerfCounters with an invalide CephContext in another).

#2 - 04/29/2013 10:18 AM - Josh Durgin

It didn't reproduce with log_max_recent = 1, but without that setting it happened after just 3 tries.

Unfortunately, since the corruption is in CephContext, there's no crash dump at the end of the log.

It looks like there may be reads still being processed by the ObjectCacher after the image is closed:

```
2013-04-29 08:29:06.611159 7fcbd5d2f780 10 objectcacher flush_set on 0x13e9b00 dne
2013-04-29 08:29:06.611161 7fcbd5d2f780 20 librbd::ImageCtx: finished flushing cache
2013-04-29 08:29:06.611162 7fcbd5d2f780 10 objectcacher release_set on 0x13e9b00 dne
2013-04-29 08:29:06.611179 7fcb56ffc700 10 objectcacher flusher finish
2013-04-29 08:29:06.611224 7fcbd5d2f780 20 librbd: close_image 0x1417610
2013-04-29 08:29:06.611563 7fcbcaefc700  7 objectcacher bh_read_finish rbd_data.101649202a25.000000000000023c/
e padding 0~32768 with 32768 bytes of zeroes
2013-04-29 08:29:06.611572 7fcbcaefc700 20 objectcacher finishing waiters
2013-04-29 08:29:06.611578 7fcbcaefc700 20 librbdwriteback: aio_cb finished
2013-04-29 08:29:06.611582 7fcbcaefc700 20 librbdwriteback: aio_cb completing
2013-04-29 08:29:06.611584 7fcbcaefc700  7 objectcacher bh_read_finish rbd_data.101649202a25.000000000000023d/
e 360448~32768 (bl is 0) returned -2
2013-04-29 08:29:06.611611 7fcbcaefc700  7 objectcacher bh_read_finish rbd_data.101649202a25.000000000000023d/
e padding 360448~32768 with 32768 bytes of zeroes
2013-04-29 08:29:06.611617 7fcbcaefc700 20 objectcacher finishing waiters
2013-04-29 08:29:06.611620 7fcbcaefc700 20 librbdwriteback: aio_cb finished
2013-04-29 08:29:06.611623 7fcbcaefc700 20 librbdwriteback: aio_cb completing
2013-04-29 08:29:06.611624 7fcbcaefc700  7 objectcacher bh_read_finish rbd_data.101649202a25.000000000000023c/
e 360448~32768 (bl is 0) returned -2
2013-04-29 08:29:06.611631 7fcbcaefc700  7 objectcacher bh_read_finish rbd_data.101649202a25.000000000000023c/
e padding 360448~32768 with 32768 bytes of zeroes
2013-04-29 08:29:06.611635 7fcbcaefc700 20 objectcacher finishing waiters
2013-04-29 08:29:06.611643 7fcbcaefc700 20 librbdwriteback: aio_cb finished
2013-04-29 08:29:06.611646 7fcbcaefc700 20 librbdwriteback: aio_cb completing
2013-04-29 08:29:06.611648 7fcbd5d2f780 10 objectcacher release_set 0x13fdad0
2013-04-29 08:29:06.611672 7fcbd5d2f780 10 objectcacher release trimming object[rbd_data.101649202a25.00000000
00000025/e oset 0x13fdad0 wr 0/0]
2013-04-29 08:29:06.611675 7fcbd5d2f780 10 objectcacher close_object object[rbd_data.101649202a25.000000000000
0025/e oset 0x13fdad0 wr 0/0]
2013-04-29 08:29:06.611728 7fcbd5d2f780 10 objectcacher release trimming object[rbd_data.101649202a25.00000000
00000041/e oset 0x13fdad0 wr 0/0]
2013-04-29 08:29:06.611732 7fcbd5d2f780 10 objectcacher close_object object[rbd_data.101649202a25.000000000000
0041/e oset 0x13fdad0 wr 0/0]
... a lot more close_object() calls until the end of the log
```

The first case of the corruption we saw was without caching enabled, so it may be a race at a higher level.

#3 - 04/29/2013 03:29 PM - Josh Durgin

The wip-rbd-close-image branch contains a potential fix. Running the test in a loop to see if it'll happen again.

#4 - 04/29/2013 04:44 PM - Sage Weil

- Status changed from 12 to 7

#5 - 04/29/2013 05:01 PM - Josh Durgin

- Status changed from 7 to In Progress

Failed on the 8th try, in a similar way, although without logs.

The ObjectCacher looks like it's been destroyed already again though, as its name is no longer valid.

```
INFO:teuthology.orchestra.run.out:1469 clone    7 order 20 su 32768 sc 11
INFO:teuthology.orchestra.run.out:checking clone #5, image image_client.0-clone5 against file /home/ubuntu/cep
htest/archive/fsx-image_client.0-parent6

gdb) bt
#0  ceph::log::SubsystemMap::should_gather (this=0x311cc90, sub=<optimized out>, level=7) at ./log/SubsystemMa
p.h:64
#1  0x00007f83352a18e4 in ObjectCacher::bh_read_finish (this=0x12fce30, poolid=3, oid=..., start=140288, lengt
h=1024, bl=..., r=-2, trust_enoent=true)
    at osdc/ObjectCacher.cc:617
#2  0x00007f83352b008f in ObjectCacher::C_ReadFinish::finish (this=0x7f819b12bd60, r=-2) at osdc/ObjectCacher.
h:475
#3  0x00007f833526680a in Context::complete (this=0x7f819b12bd60, r=<optimized out>) at ./include/Context.h:41
#4  0x00007f8335297105 in librbd::C_Request::finish (this=0x7f819b12be20, r=-2) at librbd/LibrbdWriteback.cc:5
5
#5  0x00007f8335295b24 in librbd::context_cb (c=<optimized out>, arg=0x7f819b12be20) at librbd/LibrbdWriteback
.cc:35
#6  0x00007f83345ffe5d in librados::C_AioComplete::finish (this=0x7f8320c73b30, r=<optimized out>) at ./librad
os/AioCompletionImpl.h:171
#7  0x00007f833466d8e0 in Finisher::finisher_thread_entry (this=0x123f168) at common/Finisher.cc:56
#8  0x00007f8333bf7e9a in start_thread (arg=0x7f832a7fc700) at pthread_create.c:308
#9  0x00007f8333f00cbd in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:112
#10 0x0000000000000000 in ?? ()
(gdb) up
#1  0x00007f83352a18e4 in ObjectCacher::bh_read_finish (this=0x12fce30, poolid=3, oid=..., start=140288, lengt
h=1024, bl=..., r=-2, trust_enoent=true)
    at osdc/ObjectCacher.cc:617
617     osdc/ObjectCacher.cc: No such file or directory.
(gdb) p *this
$1 = {perfcounter = 0x12fd260, cct = 0x7f83341c5778, writeback_handler = @0x12423b0, name = {static npos = <op
timized out>,
    _M_dataplus = {<std::allocator<char>> = {<__gnu_cxx::new_allocator<char>> = {<No data fields>}, <No data f
ields>}, _M_p = 0x1313b78 "P;1\001"}}, lock = @0x13135f8,
  max_dirty = 25165824, target_dirty = 16777216, max_size = 33554432, max_objects = 266, max_dirty_age = {tv =
 {tv_sec = 1, tv_nsec = 0}}, block_writes_upfront = false,
  flush_set_callback = 0, flush_set_callback_arg = 0x0,
  objects = {<std::_Vector_base<__gnu_cxx::hash_map<sobject_t, ObjectCacher::Object*, __gnu_cxx::hash<sobject_
t>, std::equal_to<sobject_t>, std::allocator<ObjectCacher::Object*> >, std::allocator<__gnu_cxx::hash_map<sobj
ect_t, ObjectCacher::Object*, __gnu_cxx::hash<sobject_t>, std::equal_to<sobject_t>, std::allocator<ObjectCache
r::Object*> > > >> = {
      _M_impl = {<std::allocator<__gnu_cxx::hash_map<sobject_t, ObjectCacher::Object*, __gnu_cxx::hash<sobject
_t>, std::equal_to<sobject_t>, std::allocator<ObjectCacher::Object*> > >> = {<__gnu_cxx::new_allocator<__gnu_c
xx::hash_map<sobject_t, ObjectCacher::Object*, __gnu_cxx::hash<sobject_t>, std::equal_to<sobject_t>, std::allo
cator<ObjectCacher::Object*> > >> = {<No data fields>}, <No data fields>}, _M_start = 0x7f82b004eb00, _M_finis
h = 0x7f82b004eba0,
        _M_end_of_storage = 0x7f82b004eba0}}, <No data fields>}, dirty_bh = {_M_t = {
      _M_impl = {<std::allocator<std::_Rb_tree_node<ObjectCacher::BufferHead*> >> = {<__gnu_cxx::new_allocator
<std::_Rb_tree_node<ObjectCacher::BufferHead*> >> = {<No data fields>}, <No data fields>},
        _M_key_compare = {<std::binary_function<ObjectCacher::BufferHead*, ObjectCacher::BufferHead*, bool>> =
 {<No data fields>}, <No data fields>}, _M_header = {
          _M_color = std::_S_red, _M_parent = 0x0, _M_left = 0x12fceb8, _M_right = 0x12fceb8}, _M_node_count =
 0}}}, bh_lru_dirty = {lru_top = {head = 0x0, tail = 0x0,
      len = 0}, lru_bot = {head = 0x0, tail = 0x0, len = 0}, lru_pintail = {head = 0x0, tail = 0x0, len = 0},
lru_num = 0, lru_num_pinned = 0, lru_max = 0,
    lru_midpoint = 0.59999999999999998}, bh_lru_rest = {lru_top = {head = 0x0, tail = 0x0, len = 0}, lru_bot =
 {head = 0x0, tail = 0x0, len = 0}, lru_pintail = {
      head = 0x0, tail = 0x0, len = 0}, lru_num = 0, lru_num_pinned = 0, lru_max = 0, lru_midpoint = 0.5999999
```

```
9999999998}, ob_lru = {lru_top = {head = 0x0, tail = 0x0,
      len = 0}, lru_bot = {head = 0x0, tail = 0x0, len = 0}, lru_pintail = {head = 0x0, tail = 0x0, len = 0},
lru_num = 0, lru_num_pinned = 0, lru_max = 0,
    lru_midpoint = 0.59999999999999998}, flusher_cond = {_vptr.Cond = 0x7f83354d9050, _c = {__data = {__lock =
 1, __futex = 134, __total_seq = 18446744073709551615,
        __wakeup_seq = 67, __woken_seq = 67, __mutex = 0x1313608, __nwaiters = 0, __broadcast_seq = 1},
      __size = "\001\000\000\000\206\000\000\000\377\377\377\377\377\377\377\377C\000\000\000\000\000\000\000C
\000\000\000\000\000\000\000\b61\001\000\000\000\000\000\000\000\000\001\000\000", __align = 575525617665}, wa
iter_mutex = 0x13135f8}, flusher_stop = true, flusher_thread = {<Thread> = {_vptr.Thread = 0x7f8334a9cdd0,
      thread_id = 0}, oc = 0x12fce30}, finisher = {cct = 0x123a940, finisher_lock = {name = 0x7f83352c50b7 "Fi
nisher::finisher_lock", id = -1, recursive = false,
      lockdep = true, backtrace = false, _m = {__data = {__lock = 0, __count = 0, __owner = 0, __nusers = 0, _
_kind = -1, __spins = 0, __list = {__prev = 0x0,
            __next = 0x0}}, __size = '\000' <repeats 16 times>"\377, \377\377\377", '\000' <repeats 19 times>,
 __align = 0}, nlock = 0, locked_by = 0, cct = 0x0,
      logger = 0x0}, finisher_cond = {_vptr.Cond = 0x7f83354d9050, _c = {__data = {__lock = 1, __futex = 2, __
total_seq = 18446744073709551615, __wakeup_seq = 1,
          __woken_seq = 1, __mutex = 0x12fd078, __nwaiters = 0, __broadcast_seq = 1},
        __size = "\001\000\000\000\002\000\000\000\377\377\377\377\377\377\377\377\001\000\000\000\000\000\000
\000\001\000\000\000\000\000\000\000x\320/\001\000\000\000\000\000\000\000\000\001\000\000", __align = 8589934
593}, waiter_mutex = 0x12fd068}, finisher_empty_cond = {_vptr.Cond = 0x7f83354d9050, _c = {__data = {__lock =
1,
          __futex = 0, __total_seq = 18446744073709551615, __wakeup_seq = 0, __woken_seq = 0, __mutex = 0x0, _
_nwaiters = 0, __broadcast_seq = 0},
        __size = "\001\000\000\000\000\000\000\000\377\377\377\377\377\377\377\377", '\000' <repeats 31 times>
, __align = 1}, waiter_mutex = 0x0}, finisher_stop = true,
    finisher_running = false, finisher_queue = {<std::_Vector_base<Context*, std::allocator<Context*> >> = {
        _M_impl = {<std::allocator<Context*>> = {<__gnu_cxx::new_allocator<Context*>> = {<No data fields>}, <N
o data fields>}, _M_start = 0x0, _M_finish = 0x0,
          _M_end_of_storage = 0x0}}, <No data fields>},
    finisher_queue_rval = {<std::_List_base<std::pair<Context*, int>, std::allocator<std::pair<Context*, int>
> >> = {
        _M_impl = {<std::allocator<std::_List_node<std::pair<Context*, int> > >> = {<__gnu_cxx::new_allocator<
std::_List_node<std::pair<Context*, int> > >> = {<No data fields>}, <No data fields>}, _M_node = {_M_next = 0x
12fd160, _M_prev = 0x12fd160}}}, <No data fields>}, logger = 0x0, finisher_thread = {<Thread> = {
        _vptr.Thread = 0x7f8334a9cdd0, thread_id = 0}, fin = 0x12fd060}}, stat_cond = {_vptr.Cond = 0x7f83354d
9050, _c = {__data = {__lock = 1, __futex = 0,
        __total_seq = 18446744073709551615, __wakeup_seq = 0, __woken_seq = 0, __mutex = 0x0, __nwaiters = 0,
__broadcast_seq = 0},
      __size = "\001\000\000\000\000\000\000\000\377\377\377\377\377\377\377\377", '\000' <repeats 31 times>,
__align = 1}, waiter_mutex = 0x0}, stat_clean = 0,
  stat_zero = 0, stat_dirty = 0, stat_rx = 0, stat_tx = 0, stat_missing = 0, stat_error = 0, stat_dirty_waitin
g = 0}
(gdb) p this->lock
$2 = (Mutex &) @0x13135f8: {name = 0x7f83352c2761 "librbd::ImageCtx::cache_lock", id = -1, recursive = false,
lockdep = true, backtrace = false, _m = {__data = {
      __lock = 1, __count = 0, __owner = 649, __nusers = 1, __kind = 2, __spins = 0, __list = {__prev = 0x0, _
_next = 0x0}},
    __size = "\001\000\000\000\000\000\000\000\211\002\000\000\001\000\000\000\002", '\000' <repeats 22 times>
, __align = 1}, nlock = 1, locked_by = 140201330460416,
  cct = 0x0, logger = 0x0}
(gdb) p this->name
$3 = {static npos = <optimized out>, _M_dataplus = {<std::allocator<char>> = {<__gnu_cxx::new_allocator<char>>
 = {<No data fields>}, <No data fields>},
    _M_p = 0x1313b78 "P;1\001"}}
```

#6 - 04/30/2013 11:22 AM - Josh Durgin

- Status changed from In Progress to 7

The fix might work after all. The test was still running against the next branch since I had specified it in the ceph task instead of the install task. 6 passes so far against the correct branch...

#7 - 04/30/2013 01:48 PM - Sage Weil

- Status changed from 7 to Resolved

commit:c2bcc2a60c2c1f66c757c01ed6bcc6778821f81d