Ceph - Bug #1116

RecoveryWQ assert failure

05/27/2011 09:48 AM - Greg Farnum

Status:	Resolved	**% Done:**	0%
Priority:	Normal	**Spent time:**	0.00 hour
Assignee:	Greg Farnum		
Category:	OSD		
Target version:	v0.29		
Source:		**Reviewed:**	
Tags:		**Affected Versions:**	
Backport:		**ceph-qa-suite:**	
Regression:	No	**Pull request ID:**	
Severity:	3 - minor	**Crash signature:**	

Description

From Fyodor:

```
2011-05-27 02:35:22.046798 7fa8ff058700 journal check_for_full at 837623808 : JOURNAL FULL 8376238
08 >= 147455 (max_size 996147200 start 837771264)
2011-05-27 02:35:23.479379 7fa8f7f49700 journal throttle: waited for bytes
2011-05-27 02:35:34.730418 7fa8ff058700 journal check_for_full at 836984832 : JOURNAL FULL 8369848
32 >= 638975 (max_size 996147200 start 837623808)
2011-05-27 02:35:36.050384 7fa8f7f49700 journal throttle: waited for bytes
2011-05-27 02:35:47.226789 7fa8ff058700 journal check_for_full at 836882432 : JOURNAL FULL 8368824
32 >= 102399 (max_size 996147200 start 836984832)
2011-05-27 02:35:48.937259 7fa8f874a700 journal throttle: waited for bytes
2011-05-27 02:35:59.985040 7fa8ff058700 journal check_for_full at 836685824 : JOURNAL FULL 8366858
24 >= 196607 (max_size 996147200 start 836882432)
2011-05-27 02:36:01.654955 7fa8f874a700 journal throttle: waited for bytes
2011-05-27 02:36:12.362896 7fa8ff058700 journal check_for_full at 835723264 : JOURNAL FULL 8357232
64 >= 962559 (max_size 996147200 start 836685824)
2011-05-27 02:36:14.375435 7fa8f7f49700 journal throttle: waited for bytes
./include/xlist.h: In function 'void xlist<T>::remove(xlist<T>::item*) [with T = PG*]', in thread
'0x7fa8f7748700'
./include/xlist.h: 107: FAILED assert(i->_list == this)
 ceph version 0.28.1 (commit:d66c6ca19bbde3c363b135b66072de44e67c6632)
 1: (xlist<PG*>::pop_front()+0xbb) [0x54f28b]
 2: (OSD::RecoveryWQ::_dequeue()+0x73) [0x56bcc3]
 3: (ThreadPool::worker()+0x10a) [0x65799a]
 4: (ThreadPool::WorkThread::entry()+0xd) [0x548c8d]
 5: (()+0x6d8c) [0x7fa904294d8c]
 6: (clone()+0x6d) [0x7fa90314704d]
 ceph version 0.28.1 (commit:d66c6ca19bbde3c363b135b66072de44e67c6632)
 1: (xlist<PG*>::pop_front()+0xbb) [0x54f28b]
 2: (OSD::RecoveryWQ::_dequeue()+0x73) [0x56bcc3]
 3: (ThreadPool::worker()+0x10a) [0x65799a]
 4: (ThreadPool::WorkThread::entry()+0xd) [0x548c8d]
 5: (()+0x6d8c) [0x7fa904294d8c]
 6: (clone()+0x6d) [0x7fa90314704d]
*** Caught signal (Aborted) **
 in thread 0x7fa8f7748700
 ceph version 0.28.1 (commit:d66c6ca19bbde3c363b135b66072de44e67c6632)
 1: /usr/bin/cosd() [0x6729f9]
 2: (()+0xfc60) [0x7fa90429dc60]
 3: (gsignal()+0x35) [0x7fa903094d05]
 4: (abort()+0x186) [0x7fa903098ab6]
 5: (__gnu_cxx::__verbose_terminate_handler()+0x11d) [0x7fa90394b6dd]
 6: (()+0xb9926) [0x7fa903949926]
```

```
 7: (()+0xb9953) [0x7fa903949953]
 8: (()+0xb9a5e) [0x7fa903949a5e]
 9: (ceph::__ceph_assert_fail(char const*, char const*, int, char const*)+0x362) [0x655e32]
 10: (xlist<PG*>::pop_front()+0xbb) [0x54f28b]
 11: (OSD::RecoveryWQ::_dequeue()+0x73) [0x56bcc3]
 12: (ThreadPool::worker()+0x10a) [0x65799a]
 13: (ThreadPool::WorkThread::entry()+0xd) [0x548c8d]
 14: (()+0x6d8c) [0x7fa904294d8c]
 15: (clone()+0x6d) [0x7fa90314704d]
```

My email:
This is an interesting one -- the invariant that assert is checking
isn't too complicated (that the object lives on the RecoveryWQ's
queue) and seems to hold everywhere the RecoveryWQ is called. And the
functions modifying the queue are always called under the workqueue
lock, and do maintenance if the xlist::item is on a different list.
Which makes me think that the problem must be from conflating the
RecoveryWQ lock and the PG lock in the few places that modify the
PG::recovery_item directly, rather than via RecoveryWQ functions.
Anybody more familiar than me with this have ideas?
Fyodor, based on the time stamps and output you've given us, I assume
you don't have more detailed logs?

Sage agrees it's probably a race from conflating those locks, and we should just switch the things that directly touch the recovery_item to go through the RecoveryWQ functions. We'll need to check the invariants before dashing off a quick patch, and make sure the locking won't hang up on us.

History

#1 - 05/27/2011 12:17 PM - Greg Farnum

- Status changed from New to 4

Okay, checked this out. It turns out that the only function violating the locking was OSD::do_recovery. Simply adding the locks looks safe there and we aren't violating other invariants or doing a significant amount of extra work by allowing it to enqueue while in progress, so I pushed it to the recoverywq_fix branch. (I just took the locks manually rather than using the RecoveryWQ functions as that's the idiom we follow elsewhere in the OSD -- maybe we should change that?)

#2 - 05/27/2011 04:01 PM - Fyodor Ustinov

Looks as though this patch has helped.
At least this osd has completd rebalancing.
Great! Thanks!

#3 - 05/30/2011 01:10 PM - Sage Weil

- Status changed from 4 to Resolved

- Target version set to v0.29

5b7c8ae8bdc26e7593323c76527cb37912b9d833