Ceph - Bug #943

3-mon cluster won't start

03/25/2011 08:49 PM - Alexandre Oliva

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

Description

I had run into this before, when I had only two monitors up, and thought the monitors had gone out of sync due to btrfs failures, and that this couldn't occur with 3 active mons, but it did.

The problem is that the cluster wouldn't start, from ceph -w's perspective. Monitors would exchange messages and choose a leader, but then, before any activity occurred, a new election would be proposed, over and over.

With 3 monitors, the scenario is even more interesting. Once I got into this situation, two monitors would choose a leader, then the third would come in and propose an election, but the second wouldn't participate, so the winner would announce the result without the second in the quorum, and the second would react by proposing another election, in which now the third node wouldn't participate, so it would call for another election, and so on, and so forth.

My guess is that this may be related with very large entries in logm taking longer to propagate than nodes are willing to wait before calling for another election and starting the process over. The next-after-committed logm entry was almost 60+MB, after a log of ping-ponging between the 3 monitors. This was after I accidentally got it to succeed, bringing all nodes down (mon, mds, and osd), then bringing up only two of the mons. When I brought it all up, it came to a halt and never started again. Or, rather, I had to manually commit a single-mon monmap to get the cluster going again. Just rsyncing the mon directory over to the other mons and restarting them was not enough, but keeping their broken configuration and adding them back one by one enabled them to sync up. Phew!

History

#1 - 03/25/2011 08:51 PM - Alexandre Oliva

Oops, I just realized the title is a bit too general. won't always restart from scratch would probably be more accurate, for it certainly works during initialization and when various other regular conditions are met, but there's something unusual that makes it fail.

#2 - 03/25/2011 10:50 PM - Alexandre Oliva

Confirmation that the problem has to do with the length of logm messages (or rather the delay in transmitting them).

mon0 logs this (grep 'election.*101[78]\|logm.*86400' | grep mon1):

2011-03-25 21:01:52.650243 7f934c652700 -- 172.31.160.4:6789/0 --> mon1 172.31.160.6:6789/0 -- paxos(logm collect lc 5198 fc 0 pn 86400 opn 0) v1 -- ?+0 0x7f9318032a90
2011-03-25 21:02:01.228867 7f934ce53700 -- 172.31.160.4:6789/0 <== mon1 172.31.160.6:6789/0 43 ==== paxos(logm last lc 5198 fc 0 pn 86400 opn 86400) v1 ==== 97750621+0+0 (1265543880 0 0) 0x7f932800cb10 con 0x7f9344037490
2011-03-25 21:02:02.540397 7f934ce53700 -- 172.31.160.4:6789/0 --> mon1 172.31.160.6:6789/0 -- paxos(logm begin lc 5198 fc 0 pn 86400 opn 0) v1 -- ?+0 0x7f93400046d0
2011-03-25 21:02:08.928465 7f934ce53700 -- 172.31.160.4:6789/0 <== mon1 172.31.160.6:6789/0 52 ==== election(propose 1017) v1 ==== 481+0+0 (3926543899 0 0) 0x7f932800cb10 con 0x7f9344037490
2011-03-25 21:02:09.029397 7f934ce53700 -- 172.31.160.4:6789/0 --> mon1 172.31.160.6:6789/0 -- election(propose 1017) v1 -- ?+0 0x7f934000fe80
2011-03-25 21:02:10.955424 7f934ce53700 -- 172.31.160.4:6789/0 <== mon1 172.31.160.6:6789/0 53 ==== election(ack 1017) v1 ==== 481+0+0 (110164590 0 0) 0x7f932800c810 con 0x7f9344037490
2011-03-25 21:02:16.118359 7f934c652700 -- 172.31.160.4:6789/0 --> mon1 172.31.160.6:6789/0 -- election(victory 1018) v1 -- ?+0 0x7f9318032a90

while mon1 logs the following corresponding messages:

2011-03-25 21:01:52.669233 7fa4dffff700 -- 172.31.160.6:6789/0 <== mon0 172.31.160.4:6789/0 59 ==== paxos(logm collect lc 5198 fc 0 pn 86400 opn 0) v1 ==== 84+0+0 (3643991308 0 0) 0x7fa4c0002160 con 0xfeb8a0
2011-03-25 21:01:52.789802 7fa4dffff700 -- 172.31.160.6:6789/0 --> mon0 172.31.160.4:6789/0 -- paxos(logm last lc 5198 fc 0 pn 86400 opn 86400) v1 -- ?+0 0x7fa4d40036b0
2011-03-25 21:02:08.926797 7fa4df7fe700 -- 172.31.160.6:6789/0 --> mon0 172.31.160.4:6789/0 -- election(propose 1017) v1 -- ?+0 0x7fa4b8000e60
2011-03-25 21:02:10.952731 7fa4dffff700 -- 172.31.160.6:6789/0 <== mon0 172.31.160.4:6789/0 68 ==== paxos(logm begin lc 5198 fc 0 pn 86400 opn 0) v1 ==== 97750621+0+0 (479195503 0 0) 0x7fa4c0000e00 con 0xfeb8a0
2011-03-25 21:02:10.954709 7fa4dffff700 -- 172.31.160.6:6789/0 <== mon0 172.31.160.4:6789/0 85 ==== election(propose 1017) v1 ==== 481+0+0 (3926543899 0 0) 0x7fa4c0004ea0 con 0xfeb8a0
2011-03-25 21:02:10.954785 7fa4dffff700 -- 172.31.160.6:6789/0 --> mon0 172.31.160.4:6789/0 -- election(ack 1017) v1 -- ?+0 0x7fa4d40017e0
2011-03-25 21:02:16.119091 7fa4dffff700 -- 172.31.160.6:6789/0 <== mon0 172.31.160.4:6789/0 86 ==== election(victory 1018) v1 ==== 489+0+0 (3599670355 0 0) 0x7fa4c0000e00 con 0xfeb8a0

I suppose it would make sense to delay the initiation of a subsequent election somehow, say, if the leader sends an estimate of the mon data sync size in the victory message (would require knowledge about the state of the interlocutor), or in an initial short message exchange. Translating size to bandwidth might require more than one transmission. An alternative would be for the leader to remember that it already sent and got an ack on a lot of data to a client one or more election cycles ago, and avoid retransmitting it if at all possible. Other thoughts?

#3 - 03/25/2011 11:02 PM - Alexandre Oliva

Oh, and please, no mocking of my slow 100Mbps home network ;-)

I'm using ceph for data replication, not for speed.

But, yeah, I should get a Gigabit switch one of these days and see if the wiring between the rooms can work at that speed. But one thing at a time ;-)

#4 - 03/28/2011 10:28 AM - Sage Weil

- Target version set to v0.27

- translation missing: en.field_position set to 545

#5 - 03/28/2011 11:13 AM - Sage Weil

- translation missing: en.field_story_points set to 2

- translation missing: en.field_position deleted (550)

- translation missing: en.field_position set to 335

#6 - 03/28/2011 12:05 PM - Sage Weil

- translation missing: en.field_position deleted (335)

- translation missing: en.field_position set to 574

#7 - 04/07/2011 04:57 PM - Greg Farnum

- Assignee set to Greg Farnum

Sorry Alexandre, I guess we lost this in the shuffle.

A programmatic solution based on log size is certainly possible, but probably overkill. Can you post up some or all of that 60MB logm file? I don't know of anything offhand that should produce a file that large, and in my testing logm messages are generally measured in the hundreds of bytes!

#8 - 04/08/2011 09:29 AM - Alexandre Oliva

'fraid I no longer have that log file :-(

From what I remember, it had lots of messages from a bunch of osds each reporting multiple times the failure of lots of other osds, and doing so over and over and over until I restarted enough monitors to (I hoped) get the system going. In this case, the elected monitor consolidated all the thousands (?) of received messages into a single log update, and that one took longer to propagate than the post-election recovery time-out.

I gather this might occur in any scenario in which a majority of the monitors and osds is brought down for an extended period (rather than becoming a separate partition), and the rest of the cluster keeps running, trying to get enough quorum for an election and accumulating osd failure messages. I can't tell how unlikely this sort of scenario is to tell whether this is an important case to fix.

As for myself, I haven't run into the problem again, we have a relatively easy work-around documented in this bug report, and my clustered servers are now on a Gigabit network (yay! :-)

#9 - 04/08/2011 09:34 AM - Greg Farnum

- Status changed from New to Won't Fix

Ah, that makes sense. I think we'll drop this one unless somebody with a more likely network configuration runs into trouble again. :)

#10 - 05/17/2011 02:27 PM - Alexandre Oliva

It happened again, even on the Gbps network. After two mons failed, the third kept accummulating messages in logm for a few hours. logm/19127 grew up to 100m, and last_committed was 19126. When the others came back up, it succeeded in transferring the 100m to them, but they wouldn't commit it, starting new elections instead. I ended up moving the logm file aside, enabling them to complete recovery. I saved the logm file on all monitors, and I have the logs for mon.1 (the one that survived) and the other two. Greg, let me know if you'd like to have a look at them when you get back from vacations.

#11 - 05/17/2011 03:04 PM - Sage Weil

- Status changed from Won't Fix to In Progress

- Assignee deleted (Greg Farnum)

- Target version changed from v0.27 to v0.29

Can you attach a tarball of one of the mon directories with the big files? It's possible this is a side effect of a logging bug. I also want to make sure I understand the issue before we plan any fixes.

Thanks!

#12 - 05/17/2011 03:44 PM - Alexandre Oliva

- File mon1-logm.tar.xz added

This tarball contains the huge logm file I had to drop to recover the cluster in the logm.dropped directory, as well as the logm directory of the live monitor (the one that survived the failure, but that's not the leader).

#13 - 05/18/2011 10:20 AM - Sage Weil

- Category set to Monitor

- Status changed from In Progress to Resolved

- Assignee set to Sage Weil

Ok, this should be fixed by 4237da886e61c88935d7fb856b49a2d9676cbf9d. Subsequent patches have some further cleanup. Thanks for your help with the logs!

Files

mon1-logm.tar.xz	457 KB	05/17/2011	Alexandre Oliva