RADOS - Bug #15653

crush: low weight devices get too many objects for num_rep > 1

04/28/2016 07:21 PM - Sage Weil

Status:	Resolved	**Start date:**	04/28/2016
Priority:	High	**Due date:**	
Assignee:		**% Done:**	80%
Category:	Performance/Resource Usage	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Source:	Q/A	**Affected Versions:**	
Tags:		**ceph-qa-suite:**	
Backport:		**Component(RADOS):**	CRUSH
Regression:	No	**Pull request ID:**	
Severity:	3 - minor	**Crash signature:**	
Reviewed:			

Description

discussion

- crush multipick anomaly

description, with example

CRUSH will correctly choose items with relative weights with the right probabilities for each independent choice. However, when choosing multiple replicas, each choice is **not** indepent, since it
has to be unique. The result is that low-weighted devices get too many items.

Simple example:

```
maetl:src (master) 03:20 PM $ cat cm.txt
# begin crush map

# devices
device 0 device0
device 1 device1
device 2 device2
device 3 device3
device 4 device4

# types
type 0 osd
type 1 domain
type 2 pool

# buckets
domain root {
    id -1        # do not change unnecessarily
    # weight 5.000
    alg straw2
    hash 0    # rjenkins1
    item device0 weight 10.00
    item device1 weight 10.0
    item device2 weight 10.0
    item device3 weight 10.0
    item device4 weight 1.000
}

# rules
```

```
rule data {
    ruleset 0
    type replicated
    min_size 1
    max_size 10
    step take root
    step choose firstn 0 type osd
    step emit
}

# end crush map
maetl:src (master) 03:20 PM $ ./crushtool -c cm.txt  -o cm
maetl:src (master) 03:20 PM $ ./crushtool -i cm --test --show-utilization --num-rep 1 --min-x 1 --
max-x 1000000 --num-rep 1
rule 0 (data), x = 1..1000000, numrep = 1..1
rule 0 (data) num_rep 1 result size == 1:    1000000/1000000
  device 0:         stored : 243456     expected : 200000
  device 1:         stored : 243624     expected : 200000
  device 2:         stored : 244486     expected : 200000
  device 3:         stored : 243881     expected : 200000
  device 4:         stored : 24553     expected : 200000
maetl:src (master) 03:20 PM $ ./crushtool -i cm --test --show-utilization --num-rep 1 --min-x 1 --
max-x 1000000 --num-rep 3
rule 0 (data), x = 1..1000000, numrep = 3..3
rule 0 (data) num_rep 3 result size == 3:    1000000/1000000
  device 0:         stored : 723984     expected : 600000
  device 1:         stored : 722923     expected : 600000
  device 2:         stored : 723153     expected : 600000
  device 3:         stored : 723394     expected : 600000
  device 4:         stored : 106546     expected : 600000
```

Note that in the 1x case, we get 1/10th the items on device 4, as expected. For 3x, it grows to 1/7th. For lower weights the amplification is more pronounced.

detailed explanation

The chances of getting a particular device during the first draw is the weight of the device divided by the sum of the weight of all devices. For example let say there are 5 devices in a bucket, with the following weights a = 10, b = 10, c = 10, d = 10, e = 1. The chances of getting e is 1/41 and the chances of getting a is 10/41.

Things get more complicated for the second draw because we have to account for a first draw that does not include a given device: it is the sum of the weight of all devices except the one we're interested in, divided by the weight of all devices. So, if we want to know the chances of e showing up in the second draw, the first draw must not include it and this has a 40/41 chance of happening. Also, during the second draw, the chance of getting e is increased because there is one less device to chose from (the one that was picked during the first draw): 1/31 (i.e. 41 - the weight of the device that was chosen). Because the second draw depends on the first draw, the probability must be multiplied: 40/41 * 1/31.

Since the chance of getting the device e in a first draw or getting the device e in a second draw are independent, the chances of getting the device e in both situations is the sum of their probability: 1/41 (first draw) + (40/41 * 1/31) (second draw).

This is a special case because all devices have the same weight except for e. If we are to calculate the probability of a being selected in the second draw, we have to sum the case where e is selected and the case where b, c or d is selected during the first draw, because they do not have the same weight. If e is selected during the first draw, a will be selected during the second draw with a probability of (1/41 * 10/40). If b, c, or d is selected during the first draw, a will be selected during the second draw with a probability of (30/41 * 10/31).

The chances of getting the device a in the first draw and the second draw is therefore: 10/41+(30/41 * 10/31)+(1/41 * 10/40)

To summarize:

- probability of getting e : 1/41 + (40/41 * 1/31) = .05586
- probability of getting a : 10/41+(30/41 * 10/31)+(1/41 * 10/40) = .48603

We are therefore 8.7 (0.48603/0.05586) more likely to get e than to get a.

From the point of view of the users, this is counter intuitive because they expect that the weight reflects the probability, which is only true for a single draw. With just one draw a is (10/41)/(1/41) = 10 times more likely to be selected than e. With two draws, a is only 8.7 times more likely to be selected than e, as shown above.

History

#1 - 04/28/2016 07:22 PM - Sage Weil

- Description updated

#2 - 04/28/2016 07:23 PM - Sage Weil

- Description updated

#3 - 06/24/2016 10:27 PM - Adam Emerson

- Category set to 10

- Assignee set to Adam Emerson

- % Done changed from 0 to 50

I just need to put in a special case for LIST, (I don't know if I should just not bother with TREE) test it some, and then everything should be set.

#4 - 06/30/2016 06:55 PM - Adam Emerson

- Status changed from Verified to Testing

- % Done changed from 50 to 80

#5 - 11/22/2016 12:03 PM - Dan van der Ster

Does this issue explain our uneven distribution? We have four racks, with 7, 8, 8, 4 hosts in each, respectively. The rack with 4 hosts (RA13) is getting noticeably more data than the others (as is the 7-host rack, RA01):

```
ID   WEIGHT     REWEIGHT SIZE   USE    AVAIL  %USE  VAR  TYPE NAME
  -2 3532.61450        -  3530T  1528T  2002T 43.29 1.19     room 0513-R-0050
 -72  911.81860        -   911T   399T   511T 43.85 1.20         rack RA01
  -4 1048.31836        -  1047T   424T   622T 40.54 1.11         rack RA05
  -6 1048.31836        -  1047T   421T   626T 40.21 1.10         rack RA09
  -9  524.15918        -   523T   282T   241T 53.94 1.48         rack RA13
```

I suppose the workaround for this is to decrease the weight of the smaller rack?

#6 - 01/24/2017 11:52 AM - Loic Dachary

See https://github.com/ceph/ceph/pull/10218 for a discussion and a tentative fix.

#7 - 01/24/2017 12:13 PM - Loic Dachary

The test Adam wrote to demonstrate the problem, made into a pull request: https://github.com/ceph/ceph/pull/13083

#8 - 01/24/2017 01:40 PM - Loic Dachary

- Description updated

- Status changed from Testing to In Progress

#9 - 01/24/2017 05:39 PM - Loic Dachary

- Description updated

#10 - 01/26/2017 06:16 AM - Loic Dachary

- Description updated

#11 - 01/27/2017 05:39 PM - Loic Dachary

- Description updated

#12 - 06/19/2017 03:57 PM - Greg Farnum

- Project changed from Ceph to RADOS

- Category deleted (10)

#13 - 06/19/2017 03:57 PM - Greg Farnum

- Category set to Performance/Resource Usage

- Component(RADOS) CRUSH added

#14 - 08/08/2017 03:39 PM - Sage Weil

- Status changed from In Progress to Verified

#15 - 11/29/2017 05:03 PM - Sage Weil

- Assignee deleted (Adam Emerson)

The new weight-set capability in crush gives us the tool to fix this, but balancer module does not try to do fix it yet via crush. the 'upmap' mode will work around it, though.

#16 - 08/20/2019 09:21 PM - Josh Durgin

- Status changed from Verified to Resolved

Closed since upmap fixes this.