rgw - Feature #41080

rgw: break up user reset-stats into multiple cls ops

08/05/2019 08:57 PM - J. Eric Ivancich

Status:	Resolved	**% Done:**	0%
Priority:	Low	**Spent time:**	0.00 hour
Assignee:	Matt Benjamin		
Category:			
Target version:	v15.0.0		
Source:		**Reviewed:**	
Tags:		**Affected Versions:**	
Backport:	octopus	**Pull request ID:**	34869

Description

Currently when a user requests the reset of user stats via radosgw-admin, a single write op is sent to the OSD holding the user's info object, and it's omap entries are read in a loop and the final result written to the object's header.

The advantage to this technique is that it is atomic, manipulating a single object with a write operation.

The downside, though, is that on OSDs that may be bogged down for other reasons, this write operation may take a while and limit access to the pg on which this object resides.

There are a couple of ideas that might mitigate this. **However given how infrequent this operation is, these changes are likely not worth implementing, at least at this point in time. Instead, this tracker is here primarily to capture these ideas for the future.**

It's important to understand that one object is read from and written to for this op.

For a user that has a lot of buckets this operation incrementally reads through their buckets, totaling the stats as it goes along, with one final write. Bucket stats are read in groups of 1000, so if a user had 100,000 buckets, this would involve 100 reads.

So the first idea is to do the reads as one op to determine the total and the write as a second op, to update the header. Presumably other reads on the PG could take place during the read op. The primary challenge here is to make sure there were no intervening writes between the read op and write op. A generation number and/or timestamp of the header write could be used to insure that the write op is ok to complete. Otherwise an error could take place, and possibly a set of retries.

The second idea would be even to break the reads into multiple ops, with enough information returned from each to continue the operation with more reads, followed by a single write. The same challenge as listed above is applicable here, although with more opportunities for races with other write ops.

Related issues:

Copied to rgw - Backport #46968: octopus: rgw: break up user reset-stats into...	**Resolved**

History

#1 - 08/05/2019 08:58 PM - J. Eric Ivancich

Comments to this tracker are invited.

#2 - 08/05/2019 10:47 PM - Josh Durgin

There are a finite number of OSD op threads. If the 100 reads in a single op take a while, they will block one of those threads. By default there are 2 threads per shard for SSD, and 8 shards, so if these kind of ops were more common, they could end up blocking I/O for 1/8th of the PGs.

The first idea doesn't help much since reads to the same PG would still be blocked on each other - there's no parallelism there today. The 2nd idea, with multiple ops, would get around this and let other work happen interspersed with these operations.

#3 - 08/07/2019 01:36 PM - J. Eric Ivancich

Thank you, Josh. That's very helpful info.

#4 - 04/30/2020 11:21 PM - Matt Benjamin

- Pull request ID set to 34869

#5 - 05/01/2020 01:07 PM - Matt Benjamin

- Status changed from New to Fix Under Review

- Assignee set to Matt Benjamin

- Backport set to octopus

#6 - 08/10/2020 03:47 PM - J. Eric Ivancich

- Status changed from Fix Under Review to Pending Backport

#7 - 08/13/2020 08:53 PM - Nathan Cutler

- Copied to Backport #46968: octopus: rgw: break up user reset-stats into multiple cls ops added

#8 - 08/18/2020 10:21 PM - Nathan Cutler

- Status changed from Pending Backport to Resolved

While running with --resolve-parent, the script "backport-create-issue" noticed that all backports of this issue are in status "Resolved" or "Rejected".