Hi, recently I work on NFS. I found a bug with Librgw GC process, here is the way to reproduce:

1. set `rgw_num_rados_handles = 10` (set this param as small as possible, then we can easily see the problem). set `rgw_gc` param to small value, too, as we can check gc list soon.
2. create a bucket and expose it with NFS-Ganesha
3. mount this export to local
4. copy 6 local file (1GB) to local path, then the file should be uploaded to bucket
5. once all 1GB file have been successfully written, remove them
6. check cluster usage with `rados df`, you will find out, some data still exists in pool(default.rgw.buckets.data), though the file are all deleted and gc list = [].

Here is some track info:

### rados df before uploading

```
[r@node1 ~]# rados df
POOL_NAME  USED  OBJECTS  CLONES  COPIES  MISSING_ON_PRIMARY  UNFOUND  DEGRADED  RD_OPS  RD_WR_OPS  WR
.rgw.root  7216  21  0  63  0  0  0  664914  432M  107  62464
default.rgw.buckets.data  21028M  6304  0  18912  0  0  0  31198  160M  66520  100803M
default.rgw.buckets.index  0  3584  0  10752  0  0  0  5742379  5608M  48338  0
default.rgw.buckets.non-ec  0  6  0  18  0  0  0  133  86016  137  0
default.rgw.control  0  512  0  1536  0  0  0  504132  411M  4873  0
default.rgw.log  71273  1737  0  5211  0  0  0  220572996  210G  147050225  1306k
default.rgw.meta  6477  38  0  114  0  0  0  504132  411M  4873  1278k
fs_data  4157M  532  0  1596  0  0  0  18119  72473k  1244384  653G
fs_metadata  29189k  29  0  87  0  0  0  47  60416  906  29804k
```

| total_objects | 12763 |
| total_used    | 83599M |
| total_avail   | 6438G  |
| total_space   | 6519G  |

### rados df after uploading

```
[r@node1 ~]# rados df
POOL_NAME  USED  OBJECTS  CLONES  COPIES  MISSING_ON_PRIMARY  UNFOUND  DEGRADED  RD_OPS  RD_WR_OPS  WR
.rgw.root  7216  21  0  63  0  0  0  665100  432M  107  62464
default.rgw.buckets.data  21028M  6304  0  18912  0  0  0  31198  160M  66520  100803M
default.rgw.buckets.index  0  3584  0  10752  0  0  0  5742379  5608M  48338  0
default.rgw.buckets.non-ec  0  6  0  18  0  0  0  133  86016  137  0
default.rgw.control  0  512  0  1536  0  0  0  504132  411M  4873  0
default.rgw.log  71273  1737  0  5211  0  0  0  220572996  210G  147050225  1306k
default.rgw.meta  6477  38  0  114  0  0  0  504132  411M  4873  1278k
fs_data  4157M  532  0  1596  0  0  0  18119  72473k  1244384  653G
fs_metadata  29189k  29  0  87  0  0  0  47  60416  906  29804k
```

| total_objects | 12763 |
| total_used    | 83599M |
| totalavail   | 6438G  |
| totalspace   | 6519G  |
As you can see, only 5GB data are deleted, 1GB still remains.

I also print the obj unique tag:

obj1
RGW GC chain size: 255, with tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502217.0
RGW GC adding chain

obj2
RGW GC chain size: 255, with tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502208.0
RGW GC adding chain

obj3
RGW GC chain size: 255, with tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502219.0
RGW GC adding chain

obj4
RGW GC chain size: 255, with tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502220.0
RGW GC adding chain

obj5
RGW GC chain size: 255, with tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502214.0
RGW GC adding chain

obj6
RGW GC chain size: 255, with tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502214.0
RGW GC adding chain

obj5 and obj6 have same tail tag: b2b41852-866c-4bab-9160-3e1a1b5d7f81.6502214.0

this tag can be divided into 3 part: zone_param_id.rgw_rados_handle_id.RGW_Request_id

obj5 and obj6 seems share the same rgw_rados_handle, but the rgw request id are always 0. I think this is the main reason that confuse RGW GC thread

Related issues:

Copied to rgw - Backport #40106: mimic: Librgw doesn't GC deleted object corr... Rejected
Copied to rgw - Backport #40107: nautilus: Librgw doesn't GC deleted object c... Resolved
Copied to rgw - Backport #40108: luminous: Librgw doesn't GC deleted object c...

Rejected

History

#1 - 12/23/2018 01:34 AM - Brad Hubbard
- Project changed from Ceph to rgw

#2 - 12/23/2018 09:11 AM - Tao CHEN
Here is the patch:
https://github.com/ceph/ceph/pull/25664

#3 - 05/08/2019 06:55 PM - Matt Benjamin
ok, I think I get this; that said--the use of >1 rados handle is not at all recommended; that said, the fix looks acceptable

#4 - 05/15/2019 02:26 PM - Casey Bodley
- Status changed from New to Fix Under Review
- Pull request ID set to 28108

#5 - 05/31/2019 02:30 PM - Matt Benjamin
- Backport set to nautilus, mimic, luminous

#6 - 05/31/2019 05:25 PM - Matt Benjamin
- Status changed from Fix Under Review to Pending Backport

#7 - 06/01/2019 10:24 AM - Nathan Cutler
- Copied to Backport #40106: mimic: Librgw doesn't GC deleted object correctly added

#8 - 06/01/2019 10:24 AM - Nathan Cutler
- Copied to Backport #40107: nautilus: Librgw doesn't GC deleted object correctly added

#9 - 06/01/2019 10:24 AM - Nathan Cutler
- Copied to Backport #40108: luminous: Librgw doesn't GC deleted object correctly added

#10 - 01/27/2021 08:09 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".