Ceph - Bug #49861

BlueFS _flush_range coredump

03/17/2021 12:14 PM - xinwei wei

Status:	Resolved	**% Done:**	0%
Priority:	Normal	**Spent time:**	0.00 hour
Assignee:			
Category:	OSD		
Target version:	v15.2.10		
Source:	Community (dev)	**Affected Versions:**	v15.2.1
Tags:		**ceph-qa-suite:**	
Backport:	nautilus, octopus, pacific	**Pull request ID:**	40581
Regression:	No	**Crash signature (v1):**	
Severity:	3 - minor	**Crash signature (v2):**	
Reviewed:	03/17/2021		

Description

Hi,
BlueFS coredump occurred during rocksdb background compact.

```
FAILED ceph_assert(!h->file->deleted)
 1: (ceph::__ceph_assert_fail(char const*, char const*, int, char const*)+0x296) [0x7f3d5bb66be6]
 2: (()+0x245f220) [0x7f3d5bb67220]
 3: (BlueFS::_flush_range(BlueFS::FileWriter*, unsigned long, unsigned long)+0x5eb) [0x55d441acf19
3]
 4: (BlueFS::_flush(BlueFS::FileWriter*, bool, bool*)+0xd83) [0x55d441ad6055]
 5: (BlueFS::_flush(BlueFS::FileWriter*, bool, std::unique_lock<std::mutex>&)+0x56) [0x55d441ad50c
6]
 6: (BlueFS::flush(BlueFS::FileWriter*, bool)+0x78) [0x55d441b36918]
 7: (BlueRocksWritableFile::Close()+0x51) [0x55d441b37ad9]
 8: (rocksdb::WritableFileWriter::Close()+0x2a7) [0x55d4414be0a1]
 9: (rocksdb::WritableFileWriter::~WritableFileWriter()+0x1f) [0x55d4414a85c9]
 10: (std::default_delete<rocksdb::WritableFileWriter>::operator()(rocksdb::WritableFileWriter*) c
onst+0x22) [0x55d4414aa85c]
 11: (std::unique_ptr<rocksdb::WritableFileWriter, std::default_delete<rocksdb::WritableFileWriter
> >::~unique_ptr()+0x49) [0x55d4414a9e3f]
 12: (rocksdb::log::Writer::~Writer()+0x37) [0x55d4415f84f7]
 13: (rocksdb::JobContext::Clean()+0x19d) [0x55d4414f3d01]
 14: (rocksdb::DBImpl::BackgroundCallFlush(rocksdb::Env::Priority)+0x509) [0x55d44157a195]
 15: (rocksdb::DBImpl::BGWorkFlush(void*)+0xb8) [0x55d441578f60]
```

our cluster: ceph version 15.2.0
It seems that rocksdb does cleanup on file close even when the file is unlinked.
As like liewegas metioned https://github.com/ceph/ceph/pull/10686#issuecomment-239447408, rocksdb probably doing such behavior.
So can we do nothing in BlueFS::_flush_range if h->file->deleted == true, like that

```
int BlueFS::_flush_range(FileWriter *h, uint64_t offset, uint64_t length)
{
    if (h->file->deleted) {
        dout(10) << __func__ << "  deleted, no-op" << dendl;
        return 0;
    }
}
```

Related issues:	
Copied to Ceph - Backport #50210: octopus: BlueFS _flush_range coredump	**Resolved**
Copied to Ceph - Backport #50211: nautilus: BlueFS _flush_range coredump	**Resolved**
Copied to Ceph - Backport #50212: pacific: BlueFS _flush_range coredump	**Resolved**

History

#1 - 04/06/2021 07:01 AM - Kefu Chai

- Status changed from New to Fix Under Review

- Backport set to nautilus, octopus, pacific

- Pull request ID set to 40581

#2 - 04/07/2021 01:18 PM - Kefu Chai

- Status changed from Fix Under Review to Pending Backport

#3 - 04/07/2021 01:20 PM - Backport Bot

- Copied to Backport #50210: octopus: BlueFS _flush_range coredump added

#4 - 04/07/2021 01:20 PM - Backport Bot

- Copied to Backport #50211: nautilus: BlueFS _flush_range coredump added

#5 - 04/07/2021 01:20 PM - Backport Bot

- Copied to Backport #50212: pacific: BlueFS _flush_range coredump added

#6 - 05/03/2021 08:52 PM - Loïc Dachary

- 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".