Bug #11424
messenger.c: short reads corrupt following ops
04/17/2015 06:57 PM - Douglas Fuller

Status: New
Priority: Low
Assignee: %
Category: 
Target version: 
Source: other
Backport: 
Regression: 
Severity: 3 - minor

Start date: 04/17/2015
Due date: 
% Done: 0%
Estimated time: 0.00 hour
Spent time: 0.00 hour
Reviewed: 
Affected Versions: ceph-qa-suite:
Release: 

Description
ceph_msg_data_pages_cursor_init consumes min(message length, buffer length), which is incorrect when the read was short and the total message length is less than the buffer length.

History

#1 - 04/27/2015 05:58 AM - Alex Elder
Can you explain more about when ceph_msg_data_pages_cursor_init() causes a problem? It's evidently an error path, and it must be in the read code...

A short read is an erroneous read, right? And I thought that the semantics of a message carrying multiple ops is that if any one op fails, they all do.

Sorry if I missed some of this discussion previously. I'm just trying to zoom in on the situation that results in the error you're trying to fix.

#2 - 04/27/2015 05:23 PM - Douglas Fuller

The underlying problem is that the buffer sizes used when unpacking the replies are those provided by the caller when preparing to send the request and the caller cannot know how long some of the reply buffers will be in advance.

This code is not in an error path, it is in the normal flow unpacking the data elements in replies.

By "short read," I mean a reply that is shorter than the size of the buffer provided by the caller. This is a normal case and to be expected on some occasions.

Perhaps this is easiest constructed by example:

Consider a ceph_msg that calls rbd.get_object_prefix followed by rbd.get_size. The caller does not know the size of the object prefix in advance, so the caller must allocate and pass a buffer of size RBD_OBJ_PREFIX_LEN_MAX = 64 bytes. The buffer and its size (64) are stored in the linked list of data items accompanying the ceph_msg via osd_req_op_cls_response_data_pages. This is also done for rbd.get_size, which has a fixed length reply of (sizeof __le64 + sizeof u8) = 9 bytes.

Suppose that the object prefix is "doug_is_great" and that its length when encoded is 32 bytes. This is a what I refer to as a "short read," because it is less than the "expected" 64 bytes. Altogether, the length of the data section in this reply is 32+9 = 41 bytes.
When processing the reply message, we reach ceph_msg_data_pages_cursor_init via read_partial_message, before we even read decode front of the message. We then calculate the number of bytes to read from the data section for the first data item using:

cursor->resid = min(length, data->length)

This line appears to be intended to avoid reading past the end of the data section. However, it will calculate that the first reply should read min(41, 64) = 41 bytes. This is incorrect because only 32 bytes constitute the reply to this data item.

We do not learn the actual length of the reply until we dispatch the message to osd_client's handle_reply when the op reply itself is decoded.

Does this make sense? I'm happy to provide more information if needed.