Workgroup: Network File System Version 4
Internet-Draft:
draft-mzhang-nfsv4-sequence-id-calibration-01
Published: 8 March 2023
Intended Status: Standards Track
Expires: 9 September 2023

Authors: M. Zhang, Ed. J. Yang, Ed.
Huawei Technologies Co. Huawei Technologies Co.
C. Tangudu, Ed. K. Parambattu, Ed.
Huawei Technologies Co. Huawei Technologies Co.

Sequence ID calibration for mis-ordered requests

Abstract

This document updates RFC8881, Network File System (NFS) version 4 minor version 1, by adding two operations to prevent the client from destroying session when getting the reply of a mis-ordered request with NFS4ERR_SEQ_MISORDERED.

In NFSv4 minor version 1, sequence ID is used to ensure that the size of the needed reply cache is tightly bounded. If the server gets a mis-ordered request, the client will often break the session and establish a new session with the server. This approach results in a significant burden on the client and the server. During the process of session rebuilding, IO performance will be affected. This is especially troublesome when network latency is substantial, as, for example when client and server are in different locations. This document will propose extensions to NFSv4 that would allow client reconnection to be dispensed with.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In NFSv4 minor version 1, according to RFC 8881, Error code NFS4ERR_SEQ_MISORDERED is returned by three operations.

- The first operation is CREATE_SESSION. csa_sequence is one argument of this operation, which is used for serializing CREATE_SESSION via a per-client ID sequence number by the client. In CREATE_SESSION request, csa_sequence should be equal to ether the sequence ID in the client ID's slot(a retry), or the slot's sequence ID + 1(correct normal request). Otherwise, NFS4ERR_SEQ_MISORDERED will be returned from the server.

- The second operation is SEQUENCE. sa_sequenceid is one argument of this operation. In SEQUENCE request, If the difference between sa_sequenceid and the server's cached sequence ID at the slot ID is two (2) or more, or if sa_sequenceid is less than the cached sequence ID (accounting for wraparound of the unsigned sequence ID value), then the server MUST return NFS4ERR_SEQ_MISORDERED.

- The third operation is CB_ SEQUENCE, which is similar to SEQUENCE. csa_sequenceid is one argument of this operation. In CB_ SEQUENCE request, If the difference between csa_sequenceid and the client's cached sequence ID at the slot ID is two (2) or more, or if csa_sequenceid is less than the cached sequence ID (accounting for wraparound of the unsigned sequence ID value), then the client MUST return NFS4ERR_SEQ_MISORDERED.

Mis-order requests may happen as a result of network partition, software bug, etc. For such request, the operations subsequent to SEQUENCE, if any, are not processed, and so slots state (sequence ID, cached reply) are not changed. That means, requests before this mis-ordered one were processed correctly and the session state was correct. In the current implementation, for most of the clients, this error code will trigger the requester breaking the session and creating a new session. This process unacceptably interferes with ongoing IO operations, especially for the IOs on the normal slots.

For example of a persistent session, there are several slots on a session. Requests on the slots are being processed correctly and replies are being received normally. Suppose on one slot of them, a mis-ordered request is received by the server and a response with NFS4ERR_SEQ_MISORDERED error returns to the client. Then, the client is going to destroy the session and establish a new session. Before the new session is ready, new requests will not be performed until the pending operations finished. The effects on IOs of normal slots will become dramatic especially in network latency is substantial, as, for example when client and server are in different locations. The client has to break the session because it does not know what sequence is expected for that session and slot. This current cached sequence information would be available to the client eliminating any need to break the session.

Two operations, SEQENCE_QUERY and CB_SEQENCE_QUERY, are added to query sequence ID cached when getting NFS4ERR_SEQ_MISORDERED error.

2. Operations for Seqence ID calibration

2.1. Operation 59: SEQENCE_QUERY-Query sequence ID of designated session and slot for calibration

2.1.1. ARGUMENTS

```
struct SEQUENCE_QUERY4args {

sessionid4 sqa_sessionid;

slotid4 sqa_slotid;

};
```

2.1.2. RESULTS

```
struct SEQUENCE_QUERY4resok {

sessionid4 sqr_sessionid;

slotid4 sqr_slotid;

sequenceid4 sqr_sequenceid;

};

union SEQUENCE_QUERY4res switch (nfsstat4 sqr_status) {

case NFS4_OK: SEQUENCE_QUERY4resok sqr_resok4;

default: void;

};
```

2.1.3. DESCRIPTION

The SEQUENCE_QUERY operation is used by the client to query sequence ID cached for designated session and slot.

SEQUENCE_QUERY MUST appear as the sole operation type of any COMPOUND in which it appears. Multiple SEQUENCE_QUERY operations can be used in one COMPOUND to query multiple sequence IDs cached for multiple slots. The error NFS4ERR_NOT_ONLY_OP will be returned when that constraint is not met. Operations other than SEQUENCE, SEQENCE_QUERY, BIND_CONN_TO_SESSION, EXCHANGE_ID, CREATE_SESSION, and DESTROY_SESSION, MUST NOT appear as the first operation in a COMPOUND.

If SEQUENCE_QUERY is received on a connection not associated with the session via CREATE_SESSION or BIND_CONN_TO_SESSION, and connection association enforcement is enabled (see Section 18.35), then the server returns NFS4ERR_CONN_NOT_BOUND_TO_SESSION.

The sqa_sessionid argument identifies the session to which this request applies. The sqr_sessionid result MUST equal sqa_sessionid.

The sqa_slotid argument is the index in the reply cache for the request. The sqr_slotid result MUST equal sqa_slotid.

The sqr_sequenceid field is the cached sequence ID on the slot. The client SHOULD use this value to calibrate sa_sequenceid in the next SEQUENCE operation, that is, sqr_sequenceid+1 SHOULD be used as the sequence ID of the next request on this slot.

2.1.4. IMPLEMENTATION

For CREATE_SESSION, SEQUENCE operations, if the sequence ID in the request is mis-ordered(see RFC8881 18.46.3 Section), the replier will fail the request by NFS4ERR_SEQ_MISORDERED and keep the reply cache unchanged on the slot of this session. When getting NFS4ERR_SEQ_MISORDERED error code in the response, the client SHOULD query the cached sequence ID of the slot and session by SEQUENCE_QUERY to calibrate its sequence ID for the subsequent requests. That is, the sequence ID in next request on this slot SHOULD be sqr_sequenceid+1.

SEQUENCE_QUERY will leave the state of the slot (sequence ID, cached reply) unchanged and lease of state related to the client ID not renewed.

If the client is querying an unknown session ID to the server, the server SHOULD return NFS4ERR_BADSESSION in the response.

If the client is attempting to access a slot the replier does not have in its slot table (It is possible the slot may have been retired), NFS4ERR_BADSLOT SHOULD be returned in the response.

2.2. Operation 15:CB_SEQUENCE_QUERY- Query backchannel sequence ID of designated session and slot for calibration

2.2.1. ARGUMENT

```
struct CB_SEQUENCE_QUERY4args {
```

```
sessionid4 csqa_sessionid;

slotid4 csqa_slotid;

};
```

2.2.2. RESULT

```
struct CB_SEQUENCE_QUERY4resok {

sessionid4 csqr_sessionid;

slotid4 csqr_slotid;

sequenceid4 csqr_sequenceid;

};

union CB_SEQUENCE_QUERY4res switch (nfsstat4 csqr_status) {

case NFS4_OK: CB_SEQUENCE_QUERY4resok csqr_resok4;

default: void;

};
```

2.2.3. DESCRIPTION

CB_SEQUENCE_QUERY is used to calibrate sequence ID of the call back request of the server for the backchannel of the session.

For each CB_COMPOUND request, the first operation MUST be CB_SEQUENCE or CB_SEQUENCE_QUERY. If any other operation is in the first position of CB_COMPOUND except CB_SEQUENCE_QUERY and CB_SEQUENCE, NFS4ERR_OP_NOT_IN_SESSION MUST be returned.

If the first operation is CB_SEQUENCE, CB_SEQUENCE MUST appear once. The error NFS4ERR_SEQUENCE_POS MUST be returned when CB_SEQUENCE is found in any position in a CB_COMPOUND beyond the first. If the first operation of a CB_COMPOUND is CB_SEQUENCE_QUERY, CB_SEQUENCE_QUERY MUST be the sole operation type. There can be multiple CB_SEQUENCE_QUERY in this CB_COMPOUND to request multiple cached sequence IDs of designated sessions and slots. If any other operations are found in this CB_COMPOUND, NFS4ERR_NOT_ONLY_OP MUST be returned.

The csqa_sessionid argument identifies the session to which this request applies. The csqr_sessionid result MUST equal csqa_sessionid.

The csqa_slotid argument is the index in the reply cache for the request. The csqr_slotid result MUST equal sqa_slotid.

The csqr_sequenceid field is the cached sequence ID on the slot. The server SHOULD use this value to calibrate csa_sequenceid in the next SEQUENCE operation, that is, csqr_sequenceid+1 SHOULD be used as the sequence ID of the next request on this slot.

2.2.4. IMPLEMENTATION

For CB_SEQUENCE operations, if the sequence ID in the call back request is mis-ordered(see RFC8881 20.9 Section), the client will fail this request by NFS4ERR_SEQ_MISORDERED and keep the reply cache unchanged on the slot of this session. When getting NFS4ERR_SEQ_MISORDERED error code in the response, the server SHOULD query the cached sequence ID of the slot and session by CB_SEQUENCE_QUERY to calibrate its sequence ID for the subsequent requests. That is, the sequence ID in next call back request on this slot should be csqr_sequenceid+1. CB_SEQUENCE_QUERY will leave the state of the slot (sequence ID, cached reply) unchanged and the reply of CB_SEQUENCE_QUERY will not renew the lease of state related to the client ID on the server side.

If the server is querying an unknown session ID to the client, the client SHOULD return NFS4ERR_BADSESSION in the response.

If the server is attempting to access a slot the client does not have in its slot table (It is possible the slot may have been retired), NFS4ERR_BADSLOT SHOULD be returned in the response.

3. IANA Considerations

This memo includes no request to IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

4. Security Considerations

All considerations from RFC 8881 security relative sections [RFC8881].

5. Acknowledgements

The authors would like to acknowledge David Noveck for reviews of the various versions of the draft.

6. References

6.1. Normative References

[RFC2119]
Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8881] Noveck, D., Ed. and C. Lever, "Network File System (NFS) Version 4 Minor Version 1 Protocol", RFC 8881, DOI 10.17487/RFC8881, August 2020, <https://www.rfc-editor.org/info/rfc8881>.

6.2. Informative References

Appendix A. An Appendix

Authors' Addresses

Zhang Mingqian (editor)
Huawei Technologies Co.
China

Email: zhangmingqian.zhang@huawei.com

Yang Jing (editor)
Huawei Technologies Co.
China

Email: yangjing8@huawei.com

Sai Chakravarthy Tangudu (editor)
Huawei Technologies Co.
India

Email: sai.chakravarthy.tangudu@huawei.com

Rijesh Kunhi Parambattu (editor)
Huawei Technologies Co.
India

Email: rijesh.kunhi.parambattu@huawei.com