Network File System (NFS) Upper Layer Binding To RPC-Over-RDMA Version 2
                    draft-ietf-nfsv4-nfs-ulb-v2-00

Abstract

   This document specifies Upper Layer Bindings of Network File System
   (NFS) protocol versions to RPC-over-RDMA version 2.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   The RPC-over-RDMA version 2 transport may employ direct data
   placement to convey data payloads associated with RPC transactions
   [I-D.cel-nfsv4-rpcrdma-version-two]. To enable successful
   interoperation, RPC client and server implementations using RPC-over-
   RDMA version 2 must agree which XDR data items and RPC procedures are
   eligible to use direct data placement (DDP).

An Upper Layer Binding specifies this agreement for one or more
versions of one RPC program. Other operational details, such as RPC
binding assignments, pairing Write chunks with result data items, and
reply size estimation, are also specified by this Binding.

This document contains material required of Upper Layer Bindings, as
specified in [I-D.cel-nfsv4-rpcrdma-version-two], for the following
NFS protocol versions:

o NFS version 2 [RFC1094]

o NFS version 3 [RFC1813]

o NFS version 4.0 [RFC7530]

o NFS version 4.1 [RFC5661]

o NFS version 4.2 [RFC7862]

Upper Layer Bindings are also provided for auxiliary protocols used
with NFS versions 2 and 3 (see Section 5).

This document assumes the reader is already familiar with concepts
and terminology defined in [I-D.cel-nfsv4-rpcrdma-version-two] and
the documents it references.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
"OPTIONAL" in this document are to be interpreted as described in BCP
14 [RFC2119] [RFC8174] when, and only when, they appear in all
capitals, as shown here.

3. Reply Size Estimation

During the construction of each RPC Call message, a Requester is
responsible for allocating appropriate resources for receiving the
corresponding Reply message. If the Requester expects the RPC Reply
message will be larger than its inline threshold, it MAY provide
Write and/or Reply chunks wherein the Responder can place results and
the reply's Payload stream.

4. Upper Layer Binding for NFS Versions 2 and 3

The Upper Layer Binding specification in this section applies to NFS
version 2 [RFC1094] and NFS version 3 [RFC1813]. For brevity, in
this document a "Legacy NFS client" refers to an NFS client using

version 2 or version 3 of the NFS RPC program (100003) to communicate
with an NFS server. Likewise, a "Legacy NFS server" is an NFS server
communicating with clients using NFS version 2 or NFS version 3.

The following XDR data items in NFS versions 2 and 3 are DDP-
eligible:

o The opaque file data argument in the NFS WRITE procedure

o The pathname argument in the NFS SYMLINK procedure

o The opaque file data result in the NFS READ procedure

o The pathname result in the NFS READLINK procedure

All other argument or result data items in NFS versions 2 and 3 are
not DDP-eligible.

A transport error does not give an indication of whether the server
has processed the arguments of the RPC Call, or whether the server
has accessed or modified client memory associated with that RPC.

4.1. Reply Size Estimation

A Legacy NFS client determines the maximum reply size for each
operation using the criteria outlined in Section 3.

4.2. RPC Binding Considerations

Legacy NFS servers traditionally listen for clients on UDP and TCP
port 2049. Additionally, they register these ports with a local
portmapper service [RFC1833].

A Legacy NFS server supporting RPC-over-RDMA version 2 on such a
network and registering itself with the RPC portmapper MAY choose an
arbitrary port, or MAY use the alternative well-known port number for
its RPC-over-RDMA service (see Section 9). The chosen port MAY be
registered with the RPC portmapper using the netids assigned in
[I-D.cel-nfsv4-rpcrdma-version-two].

5. Upper Layer Bindings for NFS Version 2 and 3 Auxiliary Protocols

NFS versions 2 and 3 are typically deployed with several other
protocols, sometimes referred to as "NFS auxiliary protocols." These
are distinct RPC programs that define procedures which are not part
of the NFS RPC program (100003). The Upper Layer Bindings in this
section apply to:

o Versions 2 and 3 of the MOUNT RPC program (100005) [RFC1813]

o Versions 1, 3, and 4 of the NLM RPC program (100021) [RFC1813]

o Version 1 of the NSM RPC program (100024), described in Chapter 11
   of [XNFS]

o Version 1 of the NFSACL RPC program (100227), which does not have
   a public definition. NFSACL is treated in this document as a de
   facto standard, as there are several interoperating
   implementations.

5.1. MOUNT, NLM, and NSM Protocols

Historically, NFS/RDMA implementations have chosen to convey the
MOUNT, NLM, and NSM protocols via TCP. To enable interoperation of
these protocols when NFS/RDMA is in use, a legacy NFS server MUST
provide support for these protocols via TCP.

5.2. NFSACL Protocol

Legacy clients and servers that support the NFSACL RPC program
typically convey NFSACL procedures on the same connection as the NFS
RPC program (100003). This obviates the need for separate rpcbind
queries to discover server support for this RPC program.

ACLs are typically small, but even large ACLs must be encoded and
decoded to some degree. Thus no data item in this Upper Layer
Protocol is DDP-eligible.

For procedures whose replies do not include an ACL object, the size
of a reply is determined directly from the NFSACL RPC program's XDR
definition. Legacy client implementations should choose a maximum
size for ACLs based on their own internal limits.

6. Upper Layer Binding For NFS Version 4

The Upper Layer Binding specification in this section applies to
versions of the NFS RPC program defined in NFS version 4.0 [RFC7530]
NFS version 4.1 [RFC5661] and NFS version 4.2 [RFC7862]

6.1. DDP-Eligibility

Only the following XDR data items in the COMPOUND procedure of all
NFS version 4 minor versions are DDP-eligible:

o The opaque data field in the WRITE4args structure

o The linkdata field of the NF4LNK arm in the createtype4 union

o The opaque data field in the READ4resok structure

o The linkdata field in the READLINK4resok structure

6.2. Reply Size Estimation

Within NFS version 4, there are certain variable-length result data
items whose maximum size cannot be estimated by clients reliably
because there is no protocol-specified size limit on these result
arrays. These include:

o The attrlist4 field

o Fields containing ACLs such as fattr4_acl, fattr4_dacl, and
   fattr4_sacl

o Fields in the fs_locations4 and fs_locations_info4 data structures

o Fields opaque to the NFS version 4 protocol which pertain to pNFS
   layout metadata, such as loc_body, loh_body, da_addr_body,
   lou_body, lrf_body, fattr_layout_types, and fs_layout_types

6.2.1. Reply Size Estimation for Minor Version 0

The NFS version 4.0 protocol itself does not impose any bound on the
size of NFS calls or responses.

Some of the data items enumerated in Section 6.2 (in particular, the
items related to ACLs and fs_locations) make it difficult to predict
the maximum size of NFS version 4.0 replies that interrogate
variable-length fattr4 attributes. Client implementations might rely
on their own internal architectural limits to constrain the reply
size, but such limits are not always guaranteed to be reliable.

When an especially large fattr4 result is expected, an NFS version
4.0 client can provide a Reply chunk to enable a large result to be
returned via explicit RDMA. An NFS version 4.0 client can use short
Reply chunk retry when an NFS COMPOUND containing a GETATTR operation
encounters a transport error.

6.2.2. Reply Size Estimation for Minor Version 1 and Newer

In NFS version 4.1 and newer minor versions, the csa_fore_chan_attrs
argument of the CREATE_SESSION operation contains a
ca_maxresponsesize field. The value in this field can be taken as

the absolute maximum size of replies generated by an NFS version 4.1
server.

This value can be used in cases where it is not possible to estimate
a reply size upper bound precisely. In practice, objects such as
ACLs, named attributes, layout bodies, and security labels are much
smaller than this maximum.

6.3. RPC Binding Considerations

NFS version 4 servers are required to listen on TCP port 2049, and
they are not required to register with an rpcbind service [RFC7530]

Therefore, an NFS version 4 server supporting RPC-over-RDMA version 2
MUST use the alternative well-known port number for its RPC-over-RDMA
service (see Section 9 Clients SHOULD connect to this well-known port
without consulting the RPC portmapper (as for NFS version 4 on TCP
transports).

6.4. NFS COMPOUND Requests

6.4.1. Multiple DDP-eligible Data Items

An NFS version 4 COMPOUND procedure can contain more than one
operation that carries a DDP-eligible data item. An NFS version 4
client provides XDR Position values in each Read chunk to
disambiguate which chunk is associated with which argument data item.
However NFS version 4 server and client implementations must agree in
advance on how to pair Write chunks with returned result data items.

In the following list, a "READ operation" refers to any NFS version 4
operation which has a DDP-eligible result data item. The mechanism
specified in Section 4.3.2 of [I-D.cel-nfsv4-rpcrdma-version-two] is
applied to this class of operations:

o If an NFS version 4 client wishes all DDP-eligible items in an NFS
   reply to be conveyed inline, it leaves the Write list empty.

o The first chunk in the Write list MUST be used by the first READ
   operation in an NFS version 4 COMPOUND procedure. The next Write
   chunk is used by the next READ operation, and so on.

o If an NFS version 4 client has provided a matching non-empty Write
   chunk, then the corresponding READ operation MUST return its DDP-
   eligible data item using that chunk.

o If an NFS version 4 client has provided an empty matching Write
   chunk, then the corresponding READ operation MUST return all of
   its result data items inline.

o If a READ operation returns a union arm which does not contain a
   DDP-eligible result, and the NFS version 4 client has provided a
   matching non-empty Write chunk, an NFS version 4 server MUST
   return an empty Write chunk in that Write list position.

o If there are more READ operations than Write chunks, then
   remaining NFS Read operations in an NFS version 4 COMPOUND that
   have no matching Write chunk MUST return their results inline.

6.4.2. Chunk List Complexity

The RPC-over-RDMA version 2 protocol does not place any limit on the
number of chunks or segments that may appear in Read or Write lists.
However, for various reasons NFS version 4 server implementations
often have practical limits on the number of chunks or segments they
are prepared to process in a single RPC transaction conveyed via RPC-
over-RDMA version 2.

These implementation limits are especially important when Kerberos
integrity or privacy is in use [RFC7861]. GSS services increase the
size of credential material in RPC headers, potentially requiring the
use of Long messages. This can increase the complexity of chunk
lists independent of the NFS version 4 COMPOUND being conveyed.

In the absence of explicit knowledge of the server's limits, NFS
version 4 clients SHOULD follow the prescriptions listed below when
constructing RPC-over-RDMA version 2 messages. NFS version 4 servers
MUST accept and process such requests.

o The Read list can contain either a Position-Zero Read chunk, one
   Read chunk with a non-zero Position, or both.

o The Write list can contain no more than one Write chunk.

o Any chunk can contain up to sixteen RDMA segments.

NFS version 4 clients wishing to send more complex chunk lists can
provide configuration interfaces to bound the complexity of NFS
version 4 COMPOUNDs, limit the number of elements in scatter-gather
operations, and avoid other sources of chunk overruns at the
receiving peer.

An NFS version 4 server SHOULD return one of the following responses
to a client that has sent an RPC transaction via RPC-over-RDMA

version 2 which cannot be processed due to chunk list complexity
limits on the server:

o A problem is detected by the transport layer while parsing the
   transport header in an RPC Call message. The server responds with
   an RDMA2_ERROR message with the err field set to ERR_CHUNK.

o A problem is detected during XDR decoding of the RPC Call message
   while the RPC layer reassembles the call's XDR stream. The server
   responds with an RPC reply with its "reply_stat" field set to
   MSG_ACCEPTED and its "accept_stat" field set to GARBAGE_ARGS.

After receiving one of these errors, an NFS version 4 client SHOULD
NOT retransmit the failing request, as the result would be the same
error. It SHOULD immediately terminate the RPC transaction
associated with the XID in the reply.

6.4.3. NFS Version 4 COMPOUND Example

The following example shows a Write list with three Write chunks, A,
B, and C. The NFS version 4 server consumes the provided Write
chunks by writing the results of the designated operations in the
compound request (READ and READLINK) back to each chunk.

    Write list:

       A --> B --> C

    NFS version 4 COMPOUND request:

       PUTFH LOOKUP READ PUTFH LOOKUP READLINK PUTFH LOOKUP READ
                    | | |
                    v   v   v
                    A   B   C

If the NFS version 4 client does not want to have the READLINK result
returned via RDMA, it provides an empty Write chunk for buffer B to
indicate that the READLINK result must be returned inline.

6.5. NFS Callback Requests

The NFS version 4 family of protocols support server-initiated
callbacks to notify NFS version 4 clients of events such as recalled
delegations.

6.5.1. NFS Version 4.0 Callback

   NFS version 4.0 implementations typically employ a separate TCP
   connection to handle callback operations, even when the forward
   channel uses an RPC-over-RDMA version 2 transport.

   No operation in the NFS version 4.0 callback RPC program conveys a
   data payload of significant size. Therefore, no XDR data items in
   this RPC program is DDP-eligible.

   A CB_RECALL reply is small and fixed in size. The CB_GETATTR reply
   contains a variable-length fattr4 data item. See Section 6.2.1 for a
   discussion of reply size prediction for this data item.

   An NFS version 4.0 client advertises netids and ad hoc port addresses
   for contacting its NFS version 4.0 callback service using the
   SETCLIENTID operation.

6.5.2. NFS Version 4.1 Callback

   In NFS version 4.1 and newer minor versions, callback operations may
   appear on the same connection as is used for NFS version 4 forward
   channel client requests. NFS version 4 clients and servers MUST use
   the approach described in [RFC8167] when backchannel operations are
   conveyed on RPC-over-RDMA version 2 transports.

   The csa_back_chan_attrs argument of the CREATE_SESSION operation
   contains a ca_maxresponsesize field. The value in this field can be
   taken as the absolute maximum size of backchannel replies generated
   by a replying NFS version 4 client.

   There are no DDP-eligible data items in callback procedures defined
   in NFS version 4.1 or NFS version 4.2. However, some callback
   operations, such as messages that convey device ID information, can
   be large. Message Continuation or a Long message might be used in
   this situation.

   When an NFS version 4.1 client can support Long Calls in its
   backchannel, it reports a backchannel ca_maxrequestsize that is
   larger than the connection's inline thresholds. Otherwise an NFS
   version 4 server MUST use only Short messages to convey backchannel
   operations.

6.6. Session-Related Considerations

   The presence of an NFS session (defined in [RFC5661] has no effect on
   the operation of RPC-over-RDMA version 2. None of the operations
   introduced to support NFS sessions (e.g. the SEQUENCE operation)

contain DDP-eligible data items. There is no need to match the
number of session slots with the number of available RPC-over-RDMA
version 2 credits.

However, there are a few new cases where an RPC transaction can fail.
For example, a Requester might receive, in response to an RPC
request, an RDMA2_ERROR message with an rdma_err value of ERR_CHUNK.
These situations are not different from existing RPC errors which an
NFS session implementation is already prepared to handle for other
transports. And as with other transports during such a failure,
there might be no SEQUENCE result available to the Requester to
distinguish whether failure occurred before or after the requested
operations were executed on the Responder.

When a transport error occurs (e.g. RDMA2_ERROR), the Requester
proceeds as usual to match the incoming XID value to a waiting RPC
Call. The RPC transaction is terminated, and the result status is
reported to the Upper Layer Protocol. The Requester's session
implementation then determines the session ID and slot for the failed
request, and performs slot recovery to make that slot usable again.
If this were not done, that slot could be rendered permanently
unavailable.

When an NFS session is not present (for example, when NFS version 4.0
is in use), a transport error does not provide an indication of
whether the server has processed the arguments of the RPC Call, or
whether the server has accessed or modified client memory associated
with that RPC.

6.7. Transport Considerations

6.7.1. Congestion Avoidance

Section 3.1 of [RFC7530] states:

   Where an NFS version 4 implementation supports operation over the
   IP network protocol, the supported transport layer between NFS and
   IP MUST be an IETF standardized transport protocol that is
   specified to avoid network congestion; such transports include TCP
   and the Stream Control Transmission Protocol (SCTP).

Section 2.9.1 of [RFC5661] further states:

   Even if NFS version 4.1 is used over a non-IP network protocol, it
   is RECOMMENDED that the transport support congestion control.

   It is permissible for a connectionless transport to be used under
   NFS version 4.1; however, reliable and in-order delivery of data

combined with congestion control by the connectionless transport
is REQUIRED. As a consequence, UDP by itself MUST NOT be used as
an NFS version 4.1 transport.

RPC-over-RDMA version 2 is constructed on a platform of RDMA Reliable
Connected QP type connections [I-D.cel-nfsv4-rpcrdma-version-two]
[RFC5041]. RDMA Reliable Connected QPs are reliable, connection-
oriented transports that guarantee in-order delivery, meeting all
above requirements for NFS version 4 transports.

6.7.2. Retransmission and Keep-alive

NFS version 4 client implementations often rely on a transport-layer
keep-alive mechanism to detect when an NFS version 4 server has
become unresponsive. When an NFS server is no longer responsive,
client-side keep-alive terminates the connection, which in turn
triggers reconnection and RPC retransmission.

Some RDMA transports (such as the Reliable Connected QP type on
InfiniBand) have no keep-alive mechanism. Without a disconnect or
new RPC traffic, such connections can remain alive long after an NFS
server has become unresponsive. Once an NFS client has consumed all
available RPC-over-RDMA version 2 credits on that transport
connection, it will forever await a reply before sending another RPC
request.

NFS version 4 clients SHOULD reserve one RPC-over-RDMA version 2
credit to use for periodic server or connection health assessment.
This credit can be used to drive an RPC request on an otherwise idle
connection, triggering either a quick affirmative server response or
immediate connection termination.

In addition to network partition and request loss scenarios, RPC-
over-RDMA version 2 transport connections can be terminated when a
Transport header is malformed, Reply messages are larger than receive
resources, or when too many RPC-over-RDMA messages are sent at once.
In such cases:

o If there is a transport error indicated (ie, RDMA2_ERROR) before
   the disconnect or instead of a disconnect, the Requester MUST
   respond to that error as prescribed by the specification of the
   RPC transport. Then the NFS version 4 rules for handling
   retransmission apply.

o If there is a transport disconnect and the Responder has provided
   no other response for a request, then only the NFS version 4 rules
   for handling retransmission apply.

7. Extending NFS Upper Layer Bindings

RPC programs such as NFS are required to have an Upper Layer Binding
specification to interoperate on RPC-over-RDMA version 2 transports
[I-D.cel-nfsv4-rpcrdma-version-two]. Via standards action, the Upper
Layer Binding specified in this document can be extended to cover
versions of the NFS version 4 protocol specified after NFS version 4
minor version 2, or to cover separately published extensions to an
existing NFS version 4 minor version, as described in [RFC8178].

8. Security Considerations

RPC-over-RDMA version 2 supports all RPC security models, including
RPCSEC_GSS security and transport-level security [RFC7861]. The
choice of what Direct Data Placement mechanism to convey RPC argument
and results does not affect this, since it changes only the method of
data transfer. Because the current document defines only the binding
of the NFS protocols atop [I-D.cel-nfsv4-rpcrdma-version-two], all
relevant security considerations are therefore to be described at
that layer.

9. IANA Considerations

The use of direct data placement in NFS introduces a need for an
additional port number assignment for networks that share traditional
UDP and TCP port spaces with RDMA services. The iWARP protocol is
such an example [RFC5040] [RFC5041].

For this purpose, a set of transport protocol port number assignments
is specified by this document. IANA has assigned the following ports
for NFS/RDMA in the IANA port registry, according to the guidelines
described in [RFC6335].

   nfsrdma 20049/tcp Network File System (NFS) over RDMA
   nfsrdma 20049/udp Network File System (NFS) over RDMA
   nfsrdma 20049/sctp Network File System (NFS) over RDMA

This document should be listed as a reference for the nfsrdma port
assignments. This document does not alter these assignments.

10. References

10.1. Normative References

   [I-D.cel-nfsv4-rpcrdma-version-two]
              Lever, C. and D. Noveck, "RPC-over-RDMA Version 2
              Protocol", draft-cel-nfsv4-rpcrdma-version-two-09 (work in
              progress), May 2019.

   [RFC1833] Srinivasan, R., "Binding Protocols for ONC RPC Version 2",
              RFC 1833, DOI 10.17487/RFC1833, August 1995,
              <https://www.rfc-editor.org/info/rfc1833>.

   [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>.

   [RFC5661] Shepler, S., Ed., Eisler, M., Ed., and D. Noveck, Ed.,
              "Network File System (NFS) Version 4 Minor Version 1
              Protocol", RFC 5661, DOI 10.17487/RFC5661, January 2010,
              <https://www.rfc-editor.org/info/rfc5661>.

   [RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S.
              Cheshire, "Internet Assigned Numbers Authority (IANA)
              Procedures for the Management of the Service Name and
              Transport Protocol Port Number Registry", BCP 165,
              RFC 6335, DOI 10.17487/RFC6335, August 2011,
              <https://www.rfc-editor.org/info/rfc6335>.

   [RFC7530] Haynes, T., Ed. and D. Noveck, Ed., "Network File System
              (NFS) Version 4 Protocol", RFC 7530, DOI 10.17487/RFC7530,
              March 2015, <https://www.rfc-editor.org/info/rfc7530>.

   [RFC7861] Adamson, A. and N. Williams, "Remote Procedure Call (RPC)
              Security Version 3", RFC 7861, DOI 10.17487/RFC7861,
              November 2016, <https://www.rfc-editor.org/info/rfc7861>.

   [RFC7862] Haynes, T., "Network File System (NFS) Version 4 Minor
              Version 2 Protocol", RFC 7862, DOI 10.17487/RFC7862,
              November 2016, <https://www.rfc-editor.org/info/rfc7862>.

   [RFC8167] Lever, C., "Bidirectional Remote Procedure Call on RPC-
              over-RDMA Transports", RFC 8167, DOI 10.17487/RFC8167,
              June 2017, <https://www.rfc-editor.org/info/rfc8167>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[RFC1094] Nowicki, B., "NFS: Network File System Protocol
           specification", RFC 1094, DOI 10.17487/RFC1094, March
           1989, <https://www.rfc-editor.org/info/rfc1094>.

[RFC1813] Callaghan, B., Pawlowski, B., and P. Staubach, "NFS
           Version 3 Protocol Specification", RFC 1813,
           DOI 10.17487/RFC1813, June 1995,
           <https://www.rfc-editor.org/info/rfc1813>.

[RFC5040] Recio, R., Metzler, B., Culley, P., Hilland, J., and D.
           Garcia, "A Remote Direct Memory Access Protocol
           Specification", RFC 5040, DOI 10.17487/RFC5040, October
           2007, <https://www.rfc-editor.org/info/rfc5040>.

[RFC5041] Shah, H., Pinkerton, J., Recio, R., and P. Culley, "Direct
           Data Placement over Reliable Transports", RFC 5041,
           DOI 10.17487/RFC5041, October 2007,
           <https://www.rfc-editor.org/info/rfc5041>.

[RFC8178] Noveck, D., "Rules for NFSv4 Extensions and Minor
           Versions", RFC 8178, DOI 10.17487/RFC8178, July 2017,
           <https://www.rfc-editor.org/info/rfc8178>.

[XNFS]   The Open Group, "Protocols for Interworking: XNFS, Version
           3W", February 1998.

Author's Address

   Charles Lever
   Oracle Corporation
   United States of America

   Email: chuck.lever@oracle.com