Network File System Version 4 T. Haynes
Internet-Draft Hammerspace
Intended status: Standards Track 5 January 2026
Expires: 9 July 2026

Adding an Atomic SWAP Operation to NFSv4.2
draft-haynes-nfsv4-swap-01

Abstract

The Network File System version 4.2 (NFSv4.2) does not provide support for the atomic update of data. This document introduces a new SWAP operation which provides for such atomic updates. This document extends NFSv4.2 (see RFC7862).

Note to Readers

Discussion of this draft takes place on the NFSv4 working group mailing list (nfsv4@ietf.org), which is archived at https://mailarchive.ietf.org/arch/search/?email_list=nfsv4. Source code and issues list for this draft can be found at https://github.com/ietf-wg-nfsv4/swap.

Working Group information can be found at https://github.com/ietf-wg-nfsv4.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 9 July 2026.

Copyright Notice

Table of Contents

1. Introduction

With the Network File System version 4.2 (NFSv4.2), atomic updates to a file are not guaranteed. A single WRITE operation might span multiple data blocks and another client doing a READ might encounter a partial WRITE. In addition, multiple WRITE operations, even within the same compound, may not be atomically applied to a file. I.e., multiple WRITE operations within the same compound could be atomically applied, but that is implementation specific. There is nothing in the protocol that ensures such coordination.

This document introduces the OPTIONAL to implement SWAP operation to NFSv4.2 to atomically apply multiple WRITE operations to a file. A client can easily determine whether or not a server supports the SWAP operation by examining the return code of the operation. If the server does not support the SWAP operation, it will return an error of NFS4ERR_NOTSUPP.

One way to use the SWAP operation is to CLONE a file and make modifications to the cloned copy. Once the cloned copy is ready, the SWAP operation can be used to atomically exchange the modified contents. Upon success, the cloned copy can be deleted.

1.1. Definitions

The definitions of the following terms are referenced as follows:

* CLONE (Section 15.13 of [RFC7862])

* clone_blksize (Section 12.2.1 of [RFC7862])

* NFS4ERR_NOTSUPP (Section 15.1.1.5 of [RFC8881])

* READ (Section 18.22 of [RFC8881])

* WRITE (Section 18.32 of [RFC8881])

1.2. 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.

2. Operation 81: SWAP - Swap a range of a file into another file

2.1. ARGUMENTS

```
/// struct SWAP4args {
///         /* SAVED_FH: source file */
///         /* CURRENT_FH: destination file */
///         stateid4        sa_src_stateid;
///         stateid4        sa_dst_stateid;
///         offset4         sa_src_offset;
///         offset4         sa_dst_offset;
///         length4         sa_count;
/// };
```

Figure 1: XDR for SWAP4args

2.2. RESULTS

```
/// struct SWAP4res {
///         nfsstat4        sr_status;
/// };
```

Figure 2: XDR for SWAP4res

2.3. DESCRIPTION

The SWAP operation is used to swap file content from a source file specified by the SAVED_FH value into a destination file specified by CURRENT_FH without actually copying the data, e.g., by using a block exchange mechanism.

Both SAVED_FH and CURRENT_FH must be regular files. If either SAVED_FH or CURRENT_FH is not a regular file, the operation MUST fail and return NFS4ERR_WRONG_TYPE.

The sa_dst_stateid MUST refer to a stateid that is valid for a WRITE operation and follows the rules for stateids in Sections 8.2.5 of [RFC7862] and 18.32.3 of [RFC8881]. The sa_src_stateid MUST refer to a stateid that is valid for a READ operation and follows the rules for stateids in Sections 8.2.5 of [RFC7862] and 18.22.3 of [RFC8881]. If either stateid is invalid, then the operation MUST fail.

The sa_src_offset is the starting offset within the source file from which the data to be swapped will be obtained, and the sa_dst_offset is the starting offset of the target region into which the swapped data will be placed. An offset of 0 (zero) indicates the start of the respective file. The number of bytes to be swapped is obtained from sa_count, except that a sa_count of 0 (zero) indicates that the number of bytes to be swapped is the count of bytes between sa_src_offset and the EOF of the source file. Both sa_src_offset and sa_dst_offset must be aligned to the clone block size (Section 12.2.1 of [RFC7862]). The number of bytes to be swapped must be a multiple of the clone block size, except in the case in which sa_src_offset plus the number of bytes to be swapped is equal to the source file size.

If the source offset or the source offset plus count is greater than the size of the source file, the operation MUST fail with NFS4ERR_INVAL. The destination offset or destination offset plus count may be greater than the size of the destination file.

If SAVED_FH and CURRENT_FH refer to the same file and the source and target ranges overlap, the operation MUST fail with NFS4ERR_INVAL.

If the target area of the SWAP operation ends beyond the end of the destination file, the offset at the end of the target area will determine the new size of the destination file. The contents of any block not part of the target area will be the same as if the file size were extended by a WRITE.

If the area to be swapped is not a multiple of the clone block size and the size of the destination file is past the end of the target area, the area between the end of the target area and the next multiple of the clone block size will be zeroed.

The SWAP operation is atomic in that other operations may not see any intermediate states between the state of the two files before the operation and after the operation. READs of the destination file will never see some blocks of the target area cloned without all of them being swapped. WRITEs of the source area will either have no effect on the data of the target file or be fully reflected in the target area of the destination file.

The completion status of the operation is indicated by sr_status.

3. Operations and Their Valid Errors

The operations and their valid errors are presented in Table 1. All error codes not defined in this document are defined in Section 15 of [RFC8881] and Section 11 of [RFC7862].

Operation	Errors
SWAP	NFS4ERR_ACCESS, NFS4ERR_ADMIN_REVOKED, NFS4ERR_BADXDR, NFS4ERR_BAD_STATEID, NFS4ERR_DEADSESSION, NFS4ERR_DELAY, NFS4ERR_DELEG_REVOKED, NFS4ERR_DQUOT, NFS4ERR_EXPIRED, NFS4ERR_FBIG, NFS4ERR_FHEXPIRED, NFS4ERR_GRACE, NFS4ERR_INVAL, NFS4ERR_IO, NFS4ERR_ISDIR, NFS4ERR_LOCKED, NFS4ERR_MOVED, NFS4ERR_NOFILEHANDLE, NFS4ERR_NOSPC, NFS4ERR_NOTSUPP, NFS4ERR_OLD_STATEID, NFS4ERR_OPENMODE, NFS4ERR_OP_NOT_IN_SESSION, NFS4ERR_PNFS_IO_HOLE, NFS4ERR_PNFS_NO_LAYOUT, NFS4ERR_REP_TOO_BIG, NFS4ERR_REP_TOO_BIG_TO_CACHE, NFS4ERR_REQ_TOO_BIG, NFS4ERR_RETRY_UNCACHED_REP, NFS4ERR_ROFS, NFS4ERR_SERVERFAULT, NFS4ERR_STALE, NFS4ERR_SYMLINK, NFS4ERR_TOO_MANY_OPS, NFS4ERR_WRONG_TYPE

Table 1: Operations and Their Valid Errors

4. Extraction of XDR

This document contains the external data representation (XDR) [RFC4506] description of the SWAP operation. The XDR description is presented in a manner that facilitates easy extraction into a ready-to-compile format. To extract the machine-readable XDR description, use the following shell script:

```
#!/bin/sh
grep '^ *///' $* | sed 's?^ */// ??' | sed 's?^ *///$??'
```

For example, if the script is named 'extract.sh' and this document is named 'spec.txt', execute the following command:

```
sh extract.sh < spec.txt > swap_prot.x
```

This script removes leading blank spaces and the sentinel sequence '///' from each line. XDR descriptions with the sentinel sequence are embedded throughout the document.

Note that the XDR code contained in this document depends on types from the NFSv4.2 nfs4_prot.x file (generated from [RFC7863]). This includes both nfs types that end with a 4, such as offset4, length4, etc., as well as more generic types such as uint32_t and uint64_t.

While the XDR can be appended to that from [RFC7863], the code snippets should be placed in their appropriate sections within the existing XDR.

5. Security Considerations

This document imposes no new security considerations to NFSv4.2.

6. IANA Considerations

This document has no IANA actions.

7. References

7.1. 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/rfc/rfc2119>.

[RFC4506] Eisler, M., Ed., "XDR: External Data Representation Standard", STD 67, RFC 4506, DOI 10.17487/RFC4506, May 2006, <https://www.rfc-editor.org/rfc/rfc4506>.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <https://www.rfc-editor.org/rfc/rfc4949>.

[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/rfc/rfc7862>.

[RFC7863] Haynes, T., "Network File System (NFS) Version 4 Minor Version 2 External Data Representation Standard (XDR) Description", RFC 7863, DOI 10.17487/RFC7863, November 2016, <https://www.rfc-editor.org/rfc/rfc7863>.

[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/rfc/rfc8174>.

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

[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/rfc/rfc8881>.

7.2. Informative References

[XFS-EXCHANGE-RANGE] Linux man-pages project, "ioctl_xfs_exchange_range(2) - Exchange data between files (Accessed January 2026)", Linux 6.10, July 2024.

Acknowledgments

Christoph Helwig inadvertently pointed out the XFS swap implementation ([XFS-EXCHANGE-RANGE]) which prompted this document.

David Noveck and Pali Rohar helped review the document.

Chris Inacio, Brian Pawlowski, and Gorry Fairhurst helped guide this process.

Author’s Address

Thomas Haynes
Hammerspace
Email: loghyr@gmail.com