

NFSv4 Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: November 25, 2012

J. Lentini  
C. Everhart  
NetApp  
D. Ellard  
Raytheon BBN Technologies  
R. Tewari  
M. Naik  
IBM Almaden  
May 24, 2012

**Administration Protocol for Federated Filesystems**  
**draft-ietf-nfsv4-federated-fs-admin-10**

Abstract

This document describes the administration protocol for a federated file system that enables file access and namespace traversal across collections of independently administered file servers. The protocol specifies a set of interfaces by which file servers with different administrators can form a file server federation that provides a namespace composed of the filesystems physically hosted on and exported by the constituent file servers.

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 [[RFC2119](#)].

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 <http://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 November 25, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](http://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.



## Table of Contents

<a href="#">1.</a>	<a href="#">Introduction . . . . .</a>	<a href="#">5</a>
<a href="#">2.</a>	<a href="#">Protocol . . . . .</a>	<a href="#">5</a>
<a href="#">3.</a>	<a href="#">Error Values . . . . .</a>	<a href="#">11</a>
<a href="#">4.</a>	<a href="#">Data Types . . . . .</a>	<a href="#">14</a>
<a href="#">4.1.</a>	<a href="#">FedFsNsdbName Equality . . . . .</a>	<a href="#">16</a>
<a href="#">5.</a>	<a href="#">Procedures . . . . .</a>	<a href="#">16</a>
<a href="#">5.1.</a>	<a href="#">FEDFS_NULL . . . . .</a>	<a href="#">17</a>
<a href="#">5.1.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">17</a>
<a href="#">5.1.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">17</a>
<a href="#">5.1.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">17</a>
<a href="#">5.2.</a>	<a href="#">FEDFS_CREATE_JUNCTION . . . . .</a>	<a href="#">17</a>
<a href="#">5.2.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">17</a>
<a href="#">5.2.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">17</a>
<a href="#">5.2.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">19</a>
<a href="#">5.3.</a>	<a href="#">FEDFS_DELETE_JUNCTION . . . . .</a>	<a href="#">19</a>
<a href="#">5.3.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">19</a>
<a href="#">5.3.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">19</a>
<a href="#">5.3.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">20</a>
<a href="#">5.4.</a>	<a href="#">FEDFS_LOOKUP_JUNCTION . . . . .</a>	<a href="#">21</a>
<a href="#">5.4.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">21</a>
<a href="#">5.4.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">21</a>
<a href="#">5.4.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">23</a>
<a href="#">5.5.</a>	<a href="#">FEDFS_CREATE_REPLICATION . . . . .</a>	<a href="#">24</a>
<a href="#">5.5.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">24</a>
<a href="#">5.5.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">24</a>
<a href="#">5.5.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">25</a>
<a href="#">5.6.</a>	<a href="#">FEDFS_DELETE_REPLICATION . . . . .</a>	<a href="#">26</a>
<a href="#">5.6.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">26</a>
<a href="#">5.6.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">26</a>
<a href="#">5.6.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">27</a>
<a href="#">5.7.</a>	<a href="#">FEDFS_LOOKUP_REPLICATION . . . . .</a>	<a href="#">27</a>
<a href="#">5.7.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">27</a>
<a href="#">5.7.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">27</a>
<a href="#">5.7.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">28</a>
<a href="#">5.8.</a>	<a href="#">FEDFS_SET_NSDB_PARAMS . . . . .</a>	<a href="#">28</a>
<a href="#">5.8.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">28</a>
<a href="#">5.8.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">29</a>
<a href="#">5.8.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">29</a>
<a href="#">5.9.</a>	<a href="#">FEDFS_GET_NSDB_PARAMS . . . . .</a>	<a href="#">30</a>
<a href="#">5.9.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">30</a>
<a href="#">5.9.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">30</a>
<a href="#">5.9.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">30</a>
<a href="#">5.10.</a>	<a href="#">FEDFS_GET_LIMITED_NSDB_PARAMS . . . . .</a>	<a href="#">30</a>
<a href="#">5.10.1.</a>	<a href="#">Synopsis . . . . .</a>	<a href="#">31</a>
<a href="#">5.10.2.</a>	<a href="#">Description . . . . .</a>	<a href="#">31</a>
<a href="#">5.10.3.</a>	<a href="#">Errors . . . . .</a>	<a href="#">31</a>



<a href="#">6.</a>	Security Considerations . . . . .	<a href="#">32</a>
<a href="#">7.</a>	IANA Considerations . . . . .	<a href="#">32</a>
<a href="#">8.</a>	Glossary . . . . .	<a href="#">32</a>
<a href="#">9.</a>	References . . . . .	<a href="#">34</a>
<a href="#">9.1.</a>	Normative References . . . . .	<a href="#">34</a>
<a href="#">9.2.</a>	Informative References . . . . .	<a href="#">35</a>
<a href="#">Appendix A.</a>	Acknowledgments . . . . .	<a href="#">36</a>
Authors'	Addresses . . . . .	<a href="#">36</a>

## 1. Introduction

A federated filesystem enables file access and namespace traversal in a uniform, secure and consistent manner across multiple independent filesystems within an enterprise (and possibly across multiple enterprises) with reasonably good performance.

Traditionally, building a namespace that spans multiple filesystems has been difficult for two reasons. First, the filesystems that export pieces of the namespace are often not in the same administrative domain. Second, there is no standard mechanism for the filesystems to cooperatively present the namespace. Filesystems might provide proprietary management tools and in some cases an administrator might be able to use the proprietary tools to build a shared namespace out of the exported filesystems. Relying on vendor-proprietary tools does not work in larger enterprises or when collaborating across enterprises because it is likely that the system will contain filesystems running different software, each with their own protocols, with no common protocol to manage the namespace or exchange namespace information.

The requirements for federated namespaces are described in [[RFC5716](#)].

The filesystem federation protocol described in [[FEDFS-NSDB](#)] allows filesystems from different vendors and/or with different administrators to cooperatively build a namespace.

This document describes the protocol used by administrators to configure the filesystems and construct the namespace.

## 2. Protocol

The RPC protocol used by the administration operations is ONC RPC [[RFC5531](#)]. The data structures used for the parameters and return values of these procedures are expressed in this document in XDR [[RFC4506](#)].

The XDR definitions below are formatted to allow the reader to easily extract them from the document. The reader can use the following shell script to extract the definitions:

```
<CODE BEGINS>
```

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





<CODE ENDS>

If the above script is stored in a file called "extract.sh", and this document is in a file called "spec.txt", then the reader can do:

<CODE BEGINS>

```
sh extract.sh < spec.txt > admin1.xdr
```

<CODE ENDS>

The effect of the script is to remove leading white space from each line, plus a sentinel sequence of "///".

The protocol definition in XDR notation is shown below. We begin by defining basic constants and structures used by the protocol. We then present the procedures defined by the protocol.

<CODE BEGINS>

```
/// /*
///  * Copyright (c) 2010 IETF Trust and the persons identified
///  * as authors of the code. All rights reserved.
///  *
///  * The authors of the code are the authors of
///  * [draft-ietf-nfsv4-federated-fs-admin-xx.txt]: J. Lentini,
///  * C. Everhart, D. Ellard, R. Tewari, and M. Naik.
///  *
///  * Redistribution and use in source and binary forms, with
///  * or without modification, are permitted provided that the
///  * following conditions are met:
///  *
///  * - Redistributions of source code must retain the above
///  *   copyright notice, this list of conditions and the
///  *   following disclaimer.
///  *
///  * - Redistributions in binary form must reproduce the above
///  *   copyright notice, this list of conditions and the
///  *   following disclaimer in the documentation and/or other
///  *   materials provided with the distribution.
///  *
///  * - Neither the name of Internet Society, IETF or IETF
///  *   Trust, nor the names of specific contributors, may be
///  *   used to endorse or promote products derived from this
```



```
/// * software without specific prior written permission.
/// *
/// * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS
/// * AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED
/// * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
/// * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
/// * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
/// * EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
/// * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
/// * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
/// * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
/// * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
/// * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
/// * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
/// * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
/// * IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
/// * ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
/// */
///
/// enum FedFsStatus {
///     FEDFS_OK = 0,
///     FEDFS_ERR_ACCESS = 1,
///     FEDFS_ERR_BADCHAR = 2,
///     FEDFS_ERR_BADNAME = 3,
///     FEDFS_ERR_NAMETOOLONG = 4,
///     FEDFS_ERR_LOOP = 5,
///     FEDFS_ERR_BADXDR = 6,
///     FEDFS_ERR_EXIST = 7,
///     FEDFS_ERR_INVALID = 8,
///     FEDFS_ERR_IO = 9,
///     FEDFS_ERR_NOSPC = 10,
///     FEDFS_ERR_NOTJUNCT = 11,
///     FEDFS_ERR_NOTLOCAL = 12,
///     FEDFS_ERR_PERM = 13,
///     FEDFS_ERR_ROFS = 14,
///     FEDFS_ERR_SVRFAULT = 15,
///     FEDFS_ERR_NOTSUPP = 16,
///     FEDFS_ERR_NSDB_ROUTE = 17,
///     FEDFS_ERR_NSDB_DOWN = 18,
///     FEDFS_ERR_NSDB_CONN = 19,
///     FEDFS_ERR_NSDB_AUTH = 20,
///     FEDFS_ERR_NSDB_LDAP = 21,
///     FEDFS_ERR_NSDB_LDAP_VAL = 22,
///     FEDFS_ERR_NSDB_NONCE = 23,
///     FEDFS_ERR_NSDB_NOFSN = 24,
///     FEDFS_ERR_NSDB_NOFSL = 25,
///     FEDFS_ERR_NSDB_RESPONSE = 26,
///     FEDFS_ERR_NSDB_FAULT = 27,
```



```

/// FEDFS_ERR_NSDB_PARAMS = 28,
/// FEDFS_ERR_NSDB_LDAP_REFERRAL = 29,
/// FEDFS_ERR_NSDB_LDAP_REFERRAL_VAL = 30,
/// FEDFS_ERR_NSDB_LDAP_REFERRAL_NOTFOLLOWED = 31,
/// FEDFS_ERR_NSDB_PARAMS_LDAP_REFERRAL = 32,
/// FEDFS_ERR_PATH_TYPE_UNSUPP = 33,
/// FEDFS_ERR_DELAY = 34,
/// FEDFS_ERR_NO_CACHE = 35,
/// FEDFS_ERR_UNKNOWN_CACHE = 36,
/// FEDFS_ERR_NO_CACHE_UPDATE = 37
/// };
///
/// typedef opaque utf8string<>;
/// typedef utf8string utf8str_cs;
/// typedef utf8string utf8str_cis;
///
/// typedef opaque FedFsUuid[16];
///
/// struct FedFsNsdbName {
///     unsigned int port;
///     utf8str_cis hostname;
/// };
///
/// typedef utf8str_cs FedFsPathComponent;
/// typedef FedFsPathComponent FedFsPathName<>;
///
/// struct FedFsFsn {
///     FedFsUuid fsnUuid;
///     FedFsNsdbName nsdbName;
/// };
///
/// enum FedFsFslType {
///     FEDFS_NFS_FSL = 0
///     /* other types TBD */
/// };
///
/// struct FedFsNfsFsl {
///     FedFsUuid fslUuid;
///     unsigned int port;
///     utf8str_cis hostname;
///     FedFsPathName path;
/// };
///
/// union FedFsFsl switch(FedFsFslType type) {
///     case FEDFS_NFS_FSL:
///         FedFsNfsFsl nfsFsl;
///     };
///

```



```

/// enum FedFsPathType {
///     FEDFS_PATH_SYS = 0,
///     FEDFS_PATH_NFS = 1
///     /* other types TBD */
/// };
///
/// union FedFsPath switch(FedFsPathType type) {
///     case FEDFS_PATH_SYS: /* administrative path */
///         FedFsPathName      adminPath;
///     case FEDFS_PATH_NFS: /* NFS namespace path */
///         FedFsPathName      nfsPath;
/// };
///
/// struct FedFsCreateArgs {
///     FedFsPath      path;
///     FedFsFsn       fsn;
/// };
///
/// enum FedFsResolveType {
///     FEDFS_RESOLVE_NONE = 0,
///     FEDFS_RESOLVE_CACHE = 1,
///     FEDFS_RESOLVE_NSDB = 2
/// };
///
/// struct FedFsLookupArgs {
///     FedFsPath      path;
///     FedFsResolveType resolve;
/// };
///
/// struct FedFsLookupResOk {
///     FedFsFsn       fsn;
///     FedFsFsl       fsl<>;
/// };
///
/// struct FedFsLookupResReferralVal {
///     FedFsNsdbName  targetNsdb;
///     unsigned int    ldapResultCode;
/// };
///
/// union FedFsLookupRes switch (FedFsStatus status) {
///     case FEDFS_OK:
///     case FEDFS_ERR_NO_CACHE_UPDATE:
///         FedFsLookupResOk      resok;
///     case FEDFS_ERR_NSDB_LDAP_VAL:
///         unsigned int          ldapResultCode;
///     case FEDFS_ERR_NSDB_LDAP_REFERRAL:
///     case FEDFS_ERR_NSDB_PARAMS_LDAP_REFERRAL:
///         FedFsNsdbName          targetNsdb;

```





```
/// case FEDFS_ERR_NSDB_LDAP_REFERRAL_VAL:
///     FedFsLookupResReferralVal resReferralVal;
/// default:
///     void;
/// };
///
/// enum FedFsConnectionSec {
///     FEDFS_SEC_NONE = 0,
///     FEDFS_SEC_TLS = 1 /* StartTLS mechanism; RFC4513, Section 3 */
///     /* other mechanisms TBD */
/// };
///
/// union FedFsNsdbParams switch (FedFsConnectionSec secType) {
///     case FEDFS_SEC_TLS:
///         opaque secData<>;
///     default:
///         void;
/// };
///
/// struct FedFsSetNsdbParamsArgs {
///     FedFsNsdbName nsdbName;
///     FedFsNsdbParams params;
/// };
///
/// union FedFsGetNsdbParamsRes switch (FedFsStatus status) {
///     case FEDFS_OK:
///         FedFsNsdbParams params;
///     default:
///         void;
/// };
///
/// union FedFsGetLimitedNsdbParamsRes switch (FedFsStatus status) {
///     case FEDFS_OK:
///         FedFsConnectionSec secType;
///     default:
///         void;
/// };
///
/// program FEDFS_PROG {
///     version FEDFS_V1 {
///         void FEDFS_NULL(void) = 0;
///         FedFsStatus FEDFS_CREATE_JUNCTION(
///             FedFsCreateArgs) = 1;
///         FedFsStatus FEDFS_DELETE_JUNCTION(
///             FedFsPath) = 2;
///         FedFsLookupRes FEDFS_LOOKUP_JUNCTION(
///             FedFsLookupArgs) = 3;
///         FedFsStatus FEDFS_CREATE_REPLICATION(
```



```
///          FedFsCreateArgs) = 7;
///      FedFsStatus FEDFS_DELETE_REPLICATION(
///          FedFsPath) = 8;
///      FedFsLookupRes FEDFS_LOOKUP_REPLICATION(
///          FedFsLookupArgs) = 9;
///      FedFsStatus FEDFS_SET_NSDB_PARAMS(
///          FedFsSetNsdbParamsArgs) = 4;
///      FedFsGetNsdbParamsRes FEDFS_GET_NSDB_PARAMS(
///          FedFsNsdbName) = 5;
///      FedFsGetLimitedNsdbParamsRes FEDFS_GET_LIMITED_NSDB_PARAMS(
///          FedFsNsdbName) = 6;
///  } = 1;
///  } = 100418;

<CODE ENDS>
```

### 3. Error Values

The results of successful operations will consist of a status of FEDFS\_OK. The results of unsuccessful operations will begin with a status, other than FEDFS\_OK, that indicates the reason why the operation failed.

Many of the error status names and meanings (and the prose for their descriptions) are taken from the specification for NFSv4 [[3530bis](#)]. Note, however, that the numeric values for the status codes are different. For example, the name and meaning of FEDFS\_ERR\_ACCESS was inspired by NFSv4's NFS4ERR\_ACCESS, but their numeric values are different.

The status of an unsuccessful operation will generally only indicate the first error encountered during the attempt to execute the operation.

FEDFS\_OK: No errors were encountered. The operation was a success.

FEDFS\_ERR\_ACCESS: Permission denied. The caller does not have the correct permission to perform the requested operation.

FEDFS\_ERR\_BADCHAR: A UTF-8 string contains a character which is not supported by the server in the context in which it being used.

FEDFS\_ERR\_BADNAME: A name string in a request consisted of valid UTF-8 characters supported by the server, but the name is not supported by the server as a valid name for the current operation.



FEDFS\_ERR\_NAMETOOLONG: Returned when the pathname in an operation exceeds the server's implementation limit.

FEDFS\_ERR\_LOOP: Returned when too many symbolic links were encountered in resolving pathname.

FEDFS\_ERR\_BADXDR: The server encountered an XDR decoding error while processing an operation.

FEDFS\_ERR\_EXIST: The junction specified already exists.

FEDFS\_ERR\_INVALID: Invalid argument for an operation.

FEDFS\_ERR\_IO: A hard error occurred while processing the requested operation.

FEDFS\_ERR\_NOSPC: The requested operation would have caused the server's filesystem to exceed some limit (for example, if there is a fixed number of junctions per fileset or per server).

FEDFS\_ERR\_NOTJUNCT: The caller specified a path that does not end in a junction as the operand for an operation that requires the last component of the path to be a junction.

FEDFS\_ERR\_NOTLOCAL: The caller specified a path that contains a junction in any position other than the last component.

FEDFS\_ERR\_PERM: The operation was not allowed because the caller is either not a privileged user or not the owner of an object that would be modified by the operation.

FEDFS\_ERR\_ROFS: A modifying operation was attempted on a read-only filesystem.

FEDFS\_ERR\_SVRFAULT: An unanticipated non-protocol error occurred on the server.

FEDFS\_ERR\_NSDB\_ROUTE: The fileserver was unable to find a route to the NSDB.

FEDFS\_ERR\_NSDB\_DOWN: The fileserver determined that the NSDB was down.

FEDFS\_ERR\_NSDB\_CONN: The fileserver was unable to establish a connection with the NSDB.



FEDFS\_ERR\_NSDB\_AUTH: The fileserver was unable to authenticate and establish a secure connection with the NSDB.

FEDFS\_ERR\_NSDB\_LDAP: An LDAP error occurred on the connection between the fileserver and NSDB.

FEDFS\_ERR\_NSDB\_LDAP\_VAL: Indicates the same error as FEDFS\_ERR\_NSDB\_LDAP, and allows the LDAP protocol error value to be returned back to the client.

FEDFS\_ERR\_NSDB\_NONCE: The fileserver was unable to locate the NCE in the appropriate NSDB.

FEDFS\_ERR\_NSDB\_NOFSN: The fileserver was unable to locate the given FSN in the appropriate NSDB.

FEDFS\_ERR\_NSDB\_NOFSL: The fileserver was unable to locate any FSLs for the given FSN in the appropriate NSDB.

FEDFS\_ERR\_NSDB\_RESPONSE: The fileserver received a malformed response from the NSDB. This includes situations when an NSDB entry (e.g. FSN or FSL) is missing a required attribute.

FEDFS\_ERR\_NSDB\_FAULT: An unanticipated error related to the NSDB occurred.

FEDFS\_ERR\_NSDB\_PARAMS: The fileserver does not have any connection parameters on record for the specified NSDB.

FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL: The fileserver received an LDAP referral that it was unable to follow.

FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_VAL: Indicates the same error as FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL, and allows the LDAP protocol error value to be returned back to the client.

FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_NOTFOLLOWED: The fileserver received an LDAP referral that it chose not to follow, either because the fileserver does not support following LDAP referrals or LDAP referral following is disabled.

FEDFS\_ERR\_NSDB\_PARAMS\_LDAP\_REFERRAL: The fileserver received an LDAP referral that it chose not to follow because the fileserver had no NSDB parameters for the NSDB targeted by the LDAP referral.





FEDFS\_ERR\_PATH\_TYPE\_UNSUPP: The fileserver does not support the specified FedFsPathType value.

FEDFS\_ERR\_NOTSUPP: The fileserver does not support the specified procedure.

FEDFS\_ERR\_DELAY: The fileserver initiated the request, but was not able to complete it in a timely fashion. The client should wait and then try the request with a new RPC transaction ID.

FEDFS\_ERR\_NO\_CACHE: The fileserver does not implement an FSN-to-FSL cache.

FEDFS\_ERR\_UNKNOWN\_CACHE: The software receiving the ONC RPC request is unaware if the fileserver implements an FSN-to-FSL cache or unable to communicate with the FSN-to-FSL cache if it exists.

FEDFS\_ERR\_NO\_CACHE\_UPDATE: The fileserver was unable to update its FSN-to-FSL cache.

#### 4. Data Types

The basic data types defined above MUST be formatted as follows:

FedFsUuid: A universally unique identifier (UUID) as described in [RFC4122] as a version 1 UUID. The UUID should be formatted in network byte order.

FedFsNsdbName: A (hostname, port) pair.

The hostname is a variable length UTF-8 string that represents an NSDB's network location in DNS name notation. It SHOULD be prepared using the server4 rules defined in Chapter 12 "Internationalization" of [3530bis]. The DNS name MUST be represented using a fully qualified domain name. A system (i.e. fileserver or administrative host) SHOULD resolve the fully qualified domain name to a network address using the system's standard resolution mechanisms.

The port is the NSDB transport port. The port value MUST be in the range 0 (zero) to 65535. A value of 0 (zero) indicates that the standard LDAP port number, 389, SHOULD be assumed.

FSNs are immutable and invariant. The attributes of an FSN, including the fedfsNsdbName, are expected to remain constant. Therefore, a FedFsNsdbName SHOULD NOT contain a network address, such as an IPv4 or IPv6 address, as this would indefinitely assign



the network address.

**FedFsPathComponent:** A case sensitive UTF-8 string containing a filesystem path component. It SHOULD be prepared using the component4 rules defined in Chapter 12 "Internationalization" of [\[3530bis\]](#).

**FedFsPathName:** A variable length array of FedFsPathComponent values representing a filesystem path. The path's first component is stored at the first position of the array, the second component is stored at the second position of the array, and so on.

The path "/" MUST be encoded as an array with zero components.

A FedFsPathName MUST NOT contain any zero-length components.

**FedFsPath:** A pathname container. The format and semantics of the pathname are defined by the FedFsPathType value.

**FedFsPathType:** The type specific description of a pathname.

A FEDFS\_PATH\_SYS is an implementation dependent administrative pathname. For example, it could be a local file system path.

A FEDFS\_PATH\_NFS is a pathname in the NFSv4 server's single-server namespace.

**FedFsNsdbParams:** A set of parameters for connecting to an NSDB. Conceptually the fileserver contains a data structure that maps an NSDB name (DNS name and port value) to these LDAP connection parameters.

The secType field indicates the security mechanism that MUST be used to protect all connections to the NSDB with the connection parameters.

A value of FEDFS\_SEC\_NONE indicates that no security mechanism is necessary. In this case, the secData array will have 0 (zero) length.

A value of FEDFS\_SEC\_TLS indicates that the StartTLS security mechanism [\[RFC4513\]](#) MUST be used to protect all connections to the NSDB. In this case, the secData array will contain an X.509v3 certificate in binary DER format [\[RFC5280\]](#). The certificate SHOULD be used by the fileserver to authenticate the identity of the NSDB. In particular, this certificate SHOULD be used to validate the NSDB's TLS certificate list chain (see 7.4.2 of [\[RFC5246\]](#)). The certificate could be that of a certificate



authority or a self-signed certificate.

#### **4.1. FedFsNsdbName Equality**

Two FedFsNsdbNames are considered equal if both their DNS name and port values are the same. As described above, the standard LDAP port number, 389, SHOULD be assumed if a port number of 0 (zero) is specified. Therefore, the FedFsNsdbName "(nsdb.example.com, 0)" is considered equal to "(nsdb.example.com, 389)" but not equal to "(nsdb.example.com, 1066)" since the port number is different or "(nsdb.foo.example.com, 389)" since the DNS name is different.

### **5. Procedures**

The procedures defined in [Section 2](#) are described in detail in the following sections.

Fileservers that participate as "internal" nodes in the federated namespace MUST implement the following procedures:

```
FEDFS_NULL
FEDFS_CREATE_JUNCTION
FEDFS_DELETE_JUNCTION
FEDFS_LOOKUP_JUNCTION
FEDFS_SET_NSDB_PARAMS
FEDFS_GET_NSDB_PARAMS
FEDFS_GET_LIMITED_NSDB_PARAMS
```

and SHOULD implement the following procedures:

```
FEDFS_CREATE_REPLICATION
FEDFS_DELETE_REPLICATION
FEDFS_LOOKUP_REPLICATION
```

Fileservers that participate as "leaf" nodes in the namespace (i.e., filesystems that host filesets that are the target of junctions, but that do not contain any junctions) are not required to implement any of these operations.

Operations that modify the state of a replicated fileset MUST result in the update of all of the replicas in a consistent manner. Ideally all of the replicas SHOULD be updated before any operation returns. If one or more of the replicas are unavailable, the operation MAY succeed, but the changes MUST be applied before the unavailable replicas are brought back online. We assume that replicas are updated via some protocol that permits state changes to be reflected consistently across the set of replicas in such a manner that the



replicas will converge to a consistent state within a bounded number of successful message exchanges between the servers hosting the replicas.

## **5.1. FEDFS\_NULL**

### **5.1.1. Synopsis**

The standard NULL procedure.

### **5.1.2. Description**

The null RPC, which is included, by convention, in every ONC RPC protocol. This procedure does not take any arguments and does not produce a result.

### **5.1.3. Errors**

None.

## **5.2. FEDFS\_CREATE\_JUNCTION**

### **5.2.1. Synopsis**

Create a new junction from some location on the server (defined as a pathname) to an FSN.

### **5.2.2. Description**

This operation creates a junction from a server-relative path to a (potentially) remote fileset named by the given FSN.

The junction directory on the server is identified by a pathname in the form of an array of one or more UTF-8 path component strings. It is not required that this path be accessible in any other manner (e.g., to a client). This path does not appear in the federated namespace, except by coincidence; there is no requirement that the global namespace parallel the server namespace, nor is it required that this path be relative to the server pseudo-root. It does not need to be a path that is accessible via NFS (although the junction will be of limited utility if the directory specified by the path is not also accessible via NFS).

If the fileset is read-only, then this operation MUST indicate this with a status of FEDFS\_ERR\_ROFS.

If the path contains an invalid UTF-8 character, then status FEDFS\_ERR\_BADCHAR MUST be returned.





The path is REQUIRED to exist and be completely local to the server. It MUST NOT contain a junction. If the last component of the path is a junction (i.e., this operation is attempting to create a junction where one already exists), then this operation MUST return the error FEDFS\_ERR\_EXIST (even if the requested junction is identical to the current junction). If any other component of the path is a junction, then this operation MUST fail with status FEDFS\_ERR\_NOTLOCAL. The path might contain a symbolic link (if supported by the local server), but the traversal of the path MUST remain within the server-local namespace.

If any component of the path does not exist, then the operation fails with status FEDFS\_ERR\_INVALID.

The server MAY enforce the local permissions on the path, including the final component. If the path cannot be traversed because of insufficient permissions, or the final component is an unexecutable or unwritable directory, then the operation MAY fail with status FEDFS\_ERR\_ACCESS.

The operation SHOULD fail with status FEDFS\_ERR\_NSDB\_PARAMS if the fileserver does not have any connection parameters on record for the specified NSDB.

The association between the path and the FSN MUST be durable before the operation returns successfully. If the operation return codes indicates success, then the junction was successfully created and is immediately accessible.

If successful, subsequent references via NFSv4 [[3530bis](#)] or NFSv4.1 [[RFC5661](#)] clients to the directory that has been replaced by the junction will result in a referral to a current location of the target fileset [[FEDFS-NSDB](#)].

The effective permissions of the directory that is converted, by this operation, into a junction are the permissions of the root directory of the target fileset. The original permissions of the directory (and any other attributes it might have) are subsumed by the junction.

This operation does not create a fileset at the location targeted by the junction. If the target fileset does not exist, the junction will still be created. An NFS client will discover the missing fileset when it traverses the junction.



### **5.2.3. Errors**

FEDFS\_ERR\_ACCESS  
FEDFS\_ERR\_BADCHAR  
FEDFS\_ERR\_BADNAME  
FEDFS\_ERR\_NAMETOOLONG  
FEDFS\_ERR\_LOOP  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_EXIST  
FEDFS\_ERR\_INVALID  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_NOSPC  
FEDFS\_ERR\_NOTLOCAL  
FEDFS\_ERR\_PERM  
FEDFS\_ERR\_ROFS  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_PATH\_TYPE\_UNSUPP  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY

## **5.3. FEDFS\_DELETE\_JUNCTION**

### **5.3.1. Synopsis**

Delete an existing junction from some location on the server (defined as a pathname).

### **5.3.2. Description**

This operation removes a junction specified by a server-relative path.

As with FEDFS\_CREATE\_JUNCTION, the junction on the server is identified by a pathname in the form of an array of one or more UTF-8 path component strings. It is not required that this path be accessible in any other manner (e.g., to a client). This path does not appear in the federated namespace, except by coincidence; there is no requirement that the global namespace reflect the server namespace, nor is it required that this path be relative to the server pseudo-root. It does not need to be a path that is accessible via NFS.

If the fileset is read-only, then this operation SHOULD indicate this with a status of FEDFS\_ERR\_ROFS.

If the path contains an invalid UTF-8 character, then status FEDFS\_ERR\_BADCHAR MUST be returned.



The path used to delete a junction might not be the same path that was used to create the junction. If the namespace on the server has changed, then the junction might now appear at a different path than where it was created. If there is more than one valid path to the junction, any of them can be used.

The path is REQUIRED to exist and be completely local to the server. It MUST NOT contain a junction, except as the final component, which MUST be a junction. If any other component of the path is a junction, then this operation MUST fail with status FEDFS\_ERR\_NOTLOCAL. If the last component of the path is not a junction then this operation MUST return status FEDFS\_ERR\_NOTJUNCT. The path might contain a symbolic link (if supported by the local server), but the traversal of the path MUST remain within the server-local namespace.

The server MAY enforce the local permissions on the path, including the final component. If the path cannot be traversed because of insufficient permissions, or the parent directory of the junction is an unexecutable or unwritable directory, then the operation MAY fail with status FEDFS\_ERR\_ACCESS.

The removal of the association between the path and the FSN MUST be durable before the operation returns successfully. If the operation return codes indicates success, then the the junction was successfully destroyed.

The effective permissions and other attributes of the directory that is restored by this operation SHOULD be identical to their value prior to the creation of the junction.

After removal of the junction, the fileserver MAY check if any of its existing junctions reference the NSDB specified in the removed junction's FSN. If the NSDB is not referenced, the fileserver MAY delete the connection parameters of the unreferenced NSDB.

#### **5.3.3. Errors**

- FEDFS\_ERR\_ACCESS
- FEDFS\_ERR\_BADCHAR
- FEDFS\_ERR\_BADNAME
- FEDFS\_ERR\_NAMETOOLONG
- FEDFS\_ERR\_LOOP
- FEDFS\_ERR\_BADXDR
- FEDFS\_ERR\_INVALID
- FEDFS\_ERR\_IO



FEDFS\_ERR\_NOTJUNCT  
FEDFS\_ERR\_NOTLOCAL  
FEDFS\_ERR\_PERM  
FEDFS\_ERR\_ROFS  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_PATH\_TYPE\_UNSUPP  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY

## **5.4. FEDFS\_LOOKUP\_JUNCTION**

### **5.4.1. Synopsis**

Query the server to discover the current value of the junction (if any) at a given path in the server namespace.

### **5.4.2. Description**

This operation queries a server to determine whether a given path ends in a junction, and if so, the FSN to which the junction refers and the filerserver's ability to resolve the junction.

Ordinary NFSv4 operations do not provide any general mechanism to determine whether an object is a junction -- there is no encoding specified by the NFSv4 protocol that can represent this information.

As with FEDFS\_CREATE\_JUNCTION, the pathname MUST be in the form of an array of one or more UTF-8 path component strings. It is not required that this path be accessible in any other manner (e.g., to a client). This path does not appear in the federated namespace, except by coincidence; there is no requirement that the global namespace reflect the server namespace, nor is it required that this path be relative to the server pseudo-root. It does not need to be a path that is accessible via NFS.

If the path contains an invalid UTF-8 character, then status FEDFS\_ERR\_BADCHAR MUST be returned.

The path used to lookup a junction might not be the same path that was used to create the junction. If the namespace on the server has changed, then a junction might now appear at a different path than where it was created. If there is more than one valid path to the junction, any of them might be used.

The path is REQUIRED to exist and be completely local to the server. It MUST NOT contain a junction, except as the final component. If any other component of the path is a junction, then this operation MUST fail with status FEDFS\_ERR\_NOTLOCAL. If the last component of





the path is not a junction then this operation MUST return the status `FEDFS_ERR_NOTJUNCT`. The path might contain a symbolic link (if supported by the local server), but the traversal of the path MUST remain within the server-local namespace.

The server MAY enforce the local permissions on the path, including the final component. If the path cannot be traversed because of insufficient permissions, or the parent directory of the junction is an unexecutable or unwritable directory, then the operation MAY fail with status `FEDFS_ERR_ACCESS`.

If the junction exists, the resolve parameter allows for testing the fileserver's ability to resolve the junction. If the junction does not exist, the fileserver will ignore the resolve parameter.

If the junction exists and the resolve parameter is set to `FEDFS_RESOLVE_NONE`, the fileserver MUST NOT attempt to resolve the FSN. This will allow the administrator to obtain the junction's FSN even if the resolution would fail. Therefore on success, the result of a `FEDFS_RESOLVE_NONE` call will return a 0 (zero) length fsl list in the `FedFsLookupResOk` structure.

If the junction exists and the resolve parameter is set to `FEDFS_RESOLVE_CACHE`, the fileserver MUST attempt to resolve the FSN using its FSL cache, if one exists. The fileserver MUST NOT resolve the FSN by contacting the appropriate NSDB. If the fileserver's cache does not have a mapping for the FSN in question, the result of the operation MUST be `FEDFS_OK` with 0 (zero) elements in the `FedFsLookupResOk` structure's fsl array. The operation MAY fail with status `FEDFS_ERR_NO_CACHE` if the fileserver does not contain an FSN-to-FSL cache or with status `FEDFS_ERR_UNKNOWN_CACHE` if the state of the cache is unknown.

If the junction exists and the resolve parameter is set to `FEDFS_RESOLVE_NSDB`, the fileserver MUST attempt to resolve the FSN by contacting the appropriate NSDB. The FSN MUST NOT be resolved using cached information. The resolution MAY fail with `FEDFS_ERR_NSDB_ROUTE`, `FEDFS_ERR_NSDB_DOWN`, `FEDFS_ERR_NSDB_CONN`, `FEDFS_ERR_NSDB_AUTH`, `FEDFS_ERR_NSDB_LDAP`, `FEDFS_ERR_NSDB_LDAP_VAL`, `FEDFS_ERR_NSDB_NOFSN`, `FEDFS_ERR_NSDB_NOFSL`, `FEDFS_ERR_NSDB_NONCE`, `FEDFS_ERR_NSDB_RESPONSE`, `FEDFS_ERR_NSDB_FAULT`, `FEDFS_ERR_NSDB_LDAP_REFERRAL`, `FEDFS_ERR_NSDB_LDAP_REFERRAL_VAL`, `FEDFS_ERR_NSDB_LDAP_REFERRAL_NOTFOLLOWED`, or `FEDFS_ERR_NSDB_PARAMS_LDAP_REFERRAL`, depending on the nature of the failure.

In the case of a LDAP failure, the fileserver MAY return either `FEDFS_ERR_NSDB_LDAP` or `FEDFS_ERR_NSDB_LDAP_VAL`. `FEDFS_ERR_NSDB_LDAP`



indicates that an LDAP protocol error occurred during the resolution. FEDFS\_ERR\_NSDB\_LDAP\_VAL also indicates that an LDAP protocol error occurred during the resolution and allows the LDAP protocol error value to be returned in the FedFsLookupRes's ldapResultCode field (see the resultCode values in [Section 4.1.9 of \[RFC4511\]](#)).

If the NSDB responds with an LDAP referral, either the Referral type defined in [Section 4.1.10 of \[RFC4511\]](#) or the SearchResultReference type defined in [Section 4.5.3 of \[RFC4511\]](#), the fileserver SHOULD process the LDAP referral using the same policies as the fileserver's file-access protocol server. The fileserver can indicate a failure while processing the LDAP referral using FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL, FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_VAL, FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_NOTFOLLOWED, or FEDFS\_ERR\_NSDB\_PARAMS\_LDAP\_REFERRAL. The FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_VAL is analogous to the FEDFS\_ERR\_NSDB\_LDAP\_VAL error and allows the LDAP protocol error value to be returned in the FedFsLookupResReferralVal's ldapResultCode field. The FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL and FEDFS\_ERR\_NSDB\_PARAMS\_LDAP\_REFERRAL errors allow the NSDB targeted by the LDAP referral to be returned in the FedFsLookupRes's targetNsdb field. Similarly, the FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_VAL error includes this information in the FedFsLookupResReferralVal's targetNsdb.

If the fileserver has a cache of FSL records, the process of resolving an FSN using an NSDB SHOULD result in the cache being updated. A failure to update the cache can be indicated with the FEDFS\_ERR\_NO\_CACHE\_UPDATE status value. When updating the cache, new FSLs for the given FSN SHOULD be added to the cache and deleted FSLs SHOULD be removed from the cache. This behavior is desirable because it allows an administrator to proactively request that the fileserver refresh its FSL cache. For example, the administrator might like to refresh the fileserver's cache when changes are made to an FSN's FSLs.

If the junction is resolved, the fileserver will include a list of UUIDs for the FSN's FSLs in the FedFsLookupResOk structure's fsl array.

#### **5.4.3. Errors**

- FEDFS\_ERR\_ACCESS
- FEDFS\_ERR\_BADCHAR
- FEDFS\_ERR\_BADNAME
- FEDFS\_ERR\_NAMETOOLONG



FEDFS\_ERR\_LOOP  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_INVALID  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_NOTJUNCT  
FEDFS\_ERR\_NOTLOCAL  
FEDFS\_ERR\_PERM  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_NSDB\_ROUTE  
FEDFS\_ERR\_NSDB\_DOWN  
FEDFS\_ERR\_NSDB\_CONN  
FEDFS\_ERR\_NSDB\_AUTH  
FEDFS\_ERR\_NSDB\_LDAP  
FEDFS\_ERR\_NSDB\_LDAP\_VAL  
FEDFS\_ERR\_NSDB\_NONCE  
FEDFS\_ERR\_NSDB\_NOFSN  
FEDFS\_ERR\_NSDB\_NOFSL  
FEDFS\_ERR\_NSDB\_RESPONSE  
FEDFS\_ERR\_NSDB\_FAULT  
FEDFS\_ERR\_NSDB\_PARAMS  
FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL  
FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_VAL  
FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_NOTFOLLOWED  
FEDFS\_ERR\_NSDB\_PARAMS\_LDAP\_REFERRAL  
FEDFS\_ERR\_PATH\_TYPE\_UNSUPP  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY  
FEDFS\_ERR\_NO\_CACHE  
FEDFS\_ERR\_UNKNOWN\_CACHE  
FEDFS\_ERR\_NO\_CACHE\_UPDATE

## **5.5. FEDFS\_CREATE\_REPLICATION**

### **5.5.1. Synopsis**

Set an FSN representing the replication information for the fileset containing the pathname.

### **5.5.2. Description**

This operation indicates the replication information to be returned for a particular fileset. An NFSv4.x client might request `fs_locations` or `fs_locations_info` at any time to detect other copies of this fileset, and this operation supports this by supplying the FSN the fileserver should use to respond. This FSN should be associated with the entire fileset in which the path resides, and should be used to satisfy `fs_locations` or `fs_locations_info` attribute requests whenever no junction is being accessed; if a junction is



being accessed, the FSN specified by FEDFS\_CREATE\_JUNCTION will take precedence. Setting the replication FSN on a fileset that already has a replication FSN set is allowed.

This operation differs from FEDFS\_CREATE\_JUNCTION in that it controls a fileset-wide attribute not associated with a junction.

The server SHOULD permit this operation even on read-only filesets, but MAY return FEDFS\_ERR\_ROFS if this is not possible.

If the path contains an invalid UTF-8 character, then status FEDFS\_ERR\_BADCHAR MUST be returned.

The path is REQUIRED to exist and be completely local to the server. It MUST NOT contain a junction. If any component of the path is a junction, then this operation MUST fail with status FEDFS\_ERR\_NOTLOCAL. The path might contain a symbolic link (if supported by the local server), but the traversal of the path MUST remain within the server-local namespace.

The server MAY enforce the local permissions on the path, including the final component. If the path cannot be traversed because of insufficient permissions, or the final component is an unexecutable or unwritable directory, then the operation MAY fail with status FEDFS\_ERR\_ACCESS.

The operation SHOULD fail with status FEDFS\_ERR\_NSDB\_PARAMS if the fileserver does not have any connection parameters on record for the specified NSDB.

The same FSN value SHOULD be associated with all replicas of a filesystem. Depending on the underlying representation, the FSN associated with a filesystem might or might not be replicated automatically with the filesystem replication mechanism. Therefore if FEDFS\_CREATE\_REPLICATION is used on one replica of a filesystem, it SHOULD be used on all replicas.

### **5.5.3. Errors**

- FEDFS\_ERR\_ACCESS
- FEDFS\_ERR\_BADCHAR
- FEDFS\_ERR\_BADNAME
- FEDFS\_ERR\_NAMETOOLONG
- FEDFS\_ERR\_LOOP
- FEDFS\_ERR\_BADXDR
- FEDFS\_ERR\_EXIST





FEDFS\_ERR\_INVALID  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_NOSPC  
FEDFS\_ERR\_NOTLOCAL  
FEDFS\_ERR\_PERM  
FEDFS\_ERR\_ROFS  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_PATH\_TYPE\_UNSUPP  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY

## **5.6. FEDFS\_DELETE\_REPLICATION**

### **5.6.1. Synopsis**

Remove the replication information for the fileset containing the pathname.

### **5.6.2. Description**

This operation removes any replication information from the fileset in which the path resides, such that NFSv4.x client requests for `fs_locations` or `fs_locations_info` in the absence of a junction will not be satisfied.

This operation differs from `FEDFS_DELETE_JUNCTION` in that it controls a fileset-wide attribute not associated with a junction.

The server SHOULD permit this operation even on read-only filesets, but MAY return `FEDFS_ERR_ROFS` if this is not possible.

If the path contains an invalid UTF-8 character, then status `FEDFS_ERR_BADCHAR` MUST be returned.

The path is REQUIRED to exist and be completely local to the server. It MUST NOT contain a junction. If any component of the path is a junction, then this operation MUST fail with status `FEDFS_ERR_NOTLOCAL`.

The server MAY enforce the local permissions on the path, including the final component. If the path cannot be traversed because of insufficient permissions, or the parent directory of the junction unexecutable or unwritable directory, then the operation MAY fail with status `FEDFS_ERR_ACCESS`.



### **5.6.3. Errors**

FEDFS\_ERR\_ACCESS  
FEDFS\_ERR\_BADCHAR  
FEDFS\_ERR\_BADNAME  
FEDFS\_ERR\_NAMETOOLONG  
FEDFS\_ERR\_LOOP  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_INVAL  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_NOTJUNCT  
FEDFS\_ERR\_NOTLOCAL  
FEDFS\_ERR\_PERM  
FEDFS\_ERR\_ROFS  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_PATH\_TYPE\_UNSUPP  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY

## **5.7. FEDFS\_LOOKUP\_REPLICATION**

### **5.7.1. Synopsis**

Query the server to discover the current replication information (if any) at the given path.

### **5.7.2. Description**

This operation queries a server to determine whether a fileset containing the given path has replication information associated with it, and if so, the FSN for that replication information.

This operation differs from FEDFS\_LOOKUP\_JUNCTION in that it inquires about a fileset-wide attribute not associated with a junction.

If the path contains an invalid UTF-8 character, then status FEDFS\_ERR\_BADCHAR MUST be returned.

The path is REQUIRED to exist and be completely local to the server. It MUST NOT contain a junction. If any component of the path is a junction, then this operation MUST fail with status FEDFS\_ERR\_NOTLOCAL.

The server MAY enforce the local permissions on the path, including the final component. If the path cannot be traversed because of insufficient permissions, or the parent directory of the junction is an unexecutable or unwritable directory, then the operation MAY fail with status FEDFS\_ERR\_ACCESS.



Interpretation of the 'resolve' parameter and the procedure's results shall be the same as specified in [Section 5.4](#) for the FEDFS\_LOOKUP\_JUNCTION operation.

### **5.7.3. Errors**

FEDFS\_ERR\_ACCESS  
FEDFS\_ERR\_BADCHAR  
FEDFS\_ERR\_BADNAME  
FEDFS\_ERR\_NAMETOOLONG  
FEDFS\_ERR\_LOOP  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_INVALID  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_NOTJUNCT  
FEDFS\_ERR\_NOTLOCAL  
FEDFS\_ERR\_PERM  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_NSDB\_ROUTE  
FEDFS\_ERR\_NSDB\_DOWN  
FEDFS\_ERR\_NSDB\_CONN  
FEDFS\_ERR\_NSDB\_AUTH  
FEDFS\_ERR\_NSDB\_LDAP  
FEDFS\_ERR\_NSDB\_LDAP\_VAL  
FEDFS\_ERR\_NSDB\_NONCE  
FEDFS\_ERR\_NSDB\_NOFSN  
FEDFS\_ERR\_NSDB\_NOFSL  
FEDFS\_ERR\_NSDB\_RESPONSE  
FEDFS\_ERR\_NSDB\_FAULT  
FEDFS\_ERR\_NSDB\_PARAMS  
FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL  
FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_VAL  
FEDFS\_ERR\_NSDB\_LDAP\_REFERRAL\_NOTFOLLOWED  
FEDFS\_ERR\_NSDB\_PARAMS\_LDAP\_REFERRAL  
FEDFS\_ERR\_PATH\_TYPE\_UNSUPP  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY  
FEDFS\_ERR\_NO\_CACHE  
FEDFS\_ERR\_UNKNOWN\_CACHE

## **5.8. FEDFS\_SET\_NSDB\_PARAMS**

### **5.8.1. Synopsis**

Set the connection parameters for the specified NSDB.



### **5.8.2. Description**

This operations allows the administrator to set the connection parameters for a given NSDB.

If a record for the given NSDB does not exist, a new record is created with the specified connection parameters.

If a record for the given NSDB does exist, the existing connection parameters are replaced with the specified connection parameters.

An NSDB is specified using a FedFsNsdbName. The rules in [Section 4.1](#) define when two FedFsNsdbNames are considered equal

The given NSDB need not be referenced by any junctions on the fileserver. This situation will occur when connection parameters for a new NSDB are installed.

The format of the connection parameters is described above.

On success, this operation returns FEDFS\_OK. When the operation returns, the new connection parameters SHOULD be used for all subsequent LDAP connections to the given NSDB. Existing connections MAY be terminated and re-established using the new connection parameters. The connection parameters SHOULD be durable across fileserver reboots.

On failure, an error value indicating the type of error is returned. The operation MAY return FEDFS\_ERR\_ACCESS if the operation's associated user does not have sufficient permissions to create/modify NSDB connection parameters.

### **5.8.3. Errors**

FEDFS\_ERR\_ACCESS  
FEDFS\_ERR\_BADCHAR  
FEDFS\_ERR\_BADNAME  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_INVALID  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_NOSPC  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY





## **5.9. FEDFS\_GET\_NSDB\_PARAMS**

### **5.9.1. Synopsis**

Get the connection parameters for the specified NSDB.

### **5.9.2. Description**

This operations allows the administrator to retrieve connection parameters, if they exist, for the given NSDB.

An NSDB is specified using a FedFsNsdbName. The rules in [Section 4.1](#) define when two FedFsNsdbNames are considered equal

A set of connection parameters is considered a match if their associated NSDB is equal (as defined above) to the operation's NSDB argument. Therefore, there is at most one set of connection parameters that can match the query described by this operation.

The format of the connection parameters is described above.

On success, this operation returns FEDFS\_OK and the connection parameters on record for the given NSDB.

On failure, an error value indicating the type of error is returned. This operation MAY return FEDFS\_ERR\_NSDB\_PARAMS to indicate that there are no connection parameters on record for the given NSDB. The operation MAY return FEDFS\_ERR\_ACCESS if the operation's associated user does not have sufficient permissions to view NSDB connection parameters.

### **5.9.3. Errors**

FEDFS\_ERR\_ACCESS  
FEDFS\_ERR\_BADCHAR  
FEDFS\_ERR\_BADNAME  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_INVALID  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_NSDB\_PARAMS  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY

## **5.10. FEDFS\_GET\_LIMITED\_NSDB\_PARAMS**



### **5.10.1. Synopsis**

Get a limited subset of the connection parameters for the specified NSDB.

### **5.10.2. Description**

This operations allows the administrator to retrieve a limited subset of information on the connection parameters, if they exist, for the given NSDB.

An NSDB is specified using a FedFsNsdbName. The rules in [Section 4.1](#) define when two FedFsNsdbNames are considered equal

A set of connection parameters is considered a match if their associated NSDB is equal (as defined above) to the operation's NSDB argument. Therefore, there is at most one set of connection parameters that can match the query described by this operation.

This operation returns a limited subset of the connection parameters. Only the FedFsConnectionSec mechanism that is used to protect communication between the fileserver and NSDB is returned.

Viewing the limited subset of NSDB connection parameters returned by FEDFS\_GET\_LIMITED\_NSDB\_PARAMS MAY be a less privileged operation than viewing the entire set of NSDB connection parameters returned by FEDFS\_GET\_NSDB\_PARAMS. For example, the full contents of an NSDB's connection parameters could contain sensitive information for some security mechanisms. FEDFS\_GET\_LIMITED\_NSDB\_PARAMS allows the fileserver to communicate a subset of the connection parameters (the security mechanism) to users with sufficient permissions without revealing more sensitive information.

On success, this operation returns FEDFS\_OK and the FedFsConnectionSec value on record for the given NSDB.

On failure, an error value indicating the type of error is returned. This operation MAY return FEDFS\_ERR\_NSDB\_PARAMS to indicate that there are no connection parameters on record for the given NSDB. The operation MAY return FEDFS\_ERR\_ACCESS if the operation's associated user does not have sufficient permissions to view the subset of NSDB connection parameters returned by this procedure.

### **5.10.3. Errors**

FEDFS\_ERR\_ACCESS



FEDFS\_ERR\_BADCHAR  
FEDFS\_ERR\_BADNAME  
FEDFS\_ERR\_BADXDR  
FEDFS\_ERR\_INVAL  
FEDFS\_ERR\_IO  
FEDFS\_ERR\_SVRFAULT  
FEDFS\_ERR\_NSDB\_PARAMS  
FEDFS\_ERR\_NOTSUPP  
FEDFS\_ERR\_DELAY

## **6. Security Considerations**

The ONC RPC protocol supports authentication, integrity and privacy via the RPCSEC\_GSS framework [[RFC2203](#)]. Fileservers which support the FedFS administration protocol described above MUST support RPCSEC\_GSS.

## **7. IANA Considerations**

A range of ONC RPC program numbers were assigned for use by FedFS using the procedure described in [Section 7.3](#) "Program Number Assignment" of [[RFC5531](#)]. The FedFS range is:

IETF NFSv4 Working Group - FedFS 100418 - 100421

This document describes version 1 of the ONC RPC program 100418 with short name "fedfs\_admin".

## **8. Glossary**

Administrator: user with the necessary authority to initiate administrative tasks on one or more servers.

Admin Entity: A server or agent that administers a collection of filesystems and persistently stores the namespace information.

Client: Any client that accesses the fileserver data using a supported filesystem access protocol.

Federation: A set of server collections and singleton servers that use a common set of interfaces and protocols in order to provide to their clients a federated namespace accessible through a filesystem access protocol.



**Fileserver:** A server exporting a filesystem via a network filesystem access protocol.

**Fileset:** The abstraction of a set of files and the directory tree that contains them. A fileset is the fundamental unit of data management in the federation.

Note that all files within a fileset are descendants of one directory, and that filesets do not span filesystems.

**Filesystem:** A self-contained unit of export for a fileserver, and the mechanism used to implement filesets. The fileset does not need to be rooted at the root of the filesystem, nor at the export point for the filesystem.

A single filesystem MAY implement more than one fileset, if the client protocol and the fileserver permit this.

**Filesystem Access Protocol:** A network filesystem access protocol such as NFSv2 [[RFC1094](#)], NFSv3 [[RFC1813](#)], NFSv4 [[3530bis](#)], or CIFS (Common Internet File System) [[MS-SMB](#)] [[MS-SMB2](#)] [[MS-CIFS](#)].

**FSL (Fileset Location):** The location of the implementation of a fileset at a particular moment in time. An FSL MUST be something that can be translated into a protocol-specific description of a resource that a client can access directly, such as an `fs_location` (for NFSv4), or share name (for CIFS). Note that not all FSLs need to be explicitly exported as long as they are contained within an exported path on the fileserver.

**FSN (Fileset Name):** A platform-independent and globally unique name for a fileset. Two FSLs that implement replicas of the same fileset MUST have the same FSN, and if a fileset is migrated from one location to another, the FSN of that fileset MUST remain the same.

**Junction:** A filesystem object used to link a directory name in the current fileset with an object within another fileset. The server-side "link" from a leaf node in one fileset to the root of another fileset.

**Namespace:** A filename/directory tree that a sufficiently authorized client can observe.

**NSDB (Namespace Database) Service:** A service that maps FSNs to FSLs. The NSDB may also be used to store other information, such as annotations for these mappings and their components.





**NSDB Node:** The name or location of a server that implements part of the NSDB service and is responsible for keeping track of the FSLs (and related info) that implement a given partition of the FSNs.

**Referral:** A server response to a client access that directs the client to evaluate the current object as a reference to an object at a different location (specified by an FSL) in another fileset, and possibly hosted on another fileserver. The client re-attempts the access to the object at the new location.

**Replica:** A replica is a redundant implementation of a fileset. Each replica shares the same FSN, but has a different FSL.

Replicas may be used to increase availability or performance. Updates to replicas of the same fileset **MUST** appear to occur in the same order, and therefore each replica is self-consistent at any moment.

We do not assume that updates to each replica occur simultaneously. If a replica is offline or unreachable, the other replicas may be updated.

**Server Collection:** A set of fileserver administered as a unit. A server collection may be administered with vendor-specific software.

The namespace provided by a server collection could be part of the federated namespace.

**Singleton Server:** A server collection containing only one server; a stand-alone fileserver.

## **9. References**

### **9.1. Normative References**

- [3530bis] Haynes, T. and D. Noveck, "NFS Version 4 Protocol", [draft-ietf-nfsv4-rfc3530bis](#) (Work In Progress), 2010.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2203] Eisler, M., Chiu, A., and L. Ling, "RPCSEC\_GSS Protocol Specification", [RFC 2203](#), September 1997.
- [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", [RFC 4122](#),



July 2005.

- [RFC4506] Eisler, M., "XDR: External Data Representation Standard", STD 67, [RFC 4506](#), May 2006.
- [RFC4511] Sermersheim, J., "Lightweight Directory Access Protocol (LDAP): The Protocol", [RFC 4511](#), June 2006.
- [RFC4513] Harrison, R., "Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms", [RFC 4513](#), June 2006.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), August 2008.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 5280](#), May 2008.
- [RFC5531] Thurlow, R., "RPC: Remote Procedure Call Protocol Specification Version 2", [RFC 5531](#), May 2009.

## **9.2. Informative References**

- [FEDFS-NSDB] Lentini, J., Everhart, C., Ellard, D., Tewari, R., and M. Naik, "NSDB Protocol for Federated Filesystems", [draft-ietf-nfsv4-federated-fs-protocol](#) (Work In Progress), 2010.
- [MS-CIFS] Microsoft Corporation, "Common Internet File System (CIFS) Protocol Specification", MS-CIFS 2.0, November 2009.
- [MS-SMB] Microsoft Corporation, "Server Message Block (SMB) Protocol Specification", MS-SMB 17.0, November 2009.
- [MS-SMB2] Microsoft Corporation, "Server Message Block (SMB) Version 2 Protocol Specification", MS-SMB2 19.0, November 2009.
- [RFC1094] Nowicki, B., "NFS: Network File System Protocol specification", [RFC 1094](#), March 1989.
- [RFC1813] Callaghan, B., Pawlowski, B., and P. Staubach, "NFS Version 3 Protocol Specification", [RFC 1813](#), June 1995.
- [RFC5661] Shepler, S., Eisler, M., and D. Noveck, "Network File System (NFS) Version 4 Minor Version 1 Protocol",



[RFC 5661](#), January 2010.

- [RFC5662] Shepler, S., Eisler, M., and D. Noveck, "Network File System (NFS) Version 4 Minor Version 1 External Data Representation Standard (XDR) Description", [RFC 5662](#), January 2010.
- [RFC5716] Lentini, J., Everhart, C., Ellard, D., Tewari, R., and M. Naik, "Requirements for Federated File Systems", [RFC 5716](#), January 2010.

## [Appendix A](#). Acknowledgments

We would like to thank Robert Thurlow of Sun Microsystems for helping to author this document, including drafting the replication procedure text.

We would also like to thank Paul Lemahieu of EMC and Mario Wurzl of EMC for helping to author this document.

We would also like to thank Trond Myklebust for suggesting improvements to the FSL pathname format, Chuck Lever for suggesting improvements to the XDR type definitions and error codes, David Noveck for his suggestions on internationalization and path encoding rules, and Nicolas Williams for his suggestions.

The `extract.sh` shell script and formatting conventions were first described by the authors of the NFSv4.1 XDR specification [[RFC5662](#)].

## Authors' Addresses

James Lentini  
NetApp  
1601 Trapelo Rd, Suite 16  
Waltham, MA 02451  
US

Phone: +1 781-768-5359  
Email: [jlentini@netapp.com](mailto:jlentini@netapp.com)



Craig Everhart  
NetApp  
800 Cranberry Woods Drive  
Cranberry Township, PA 16066  
US

Phone: +1 724-741-5101  
Email: [Craig.Everhart@netapp.com](mailto:Craig.Everhart@netapp.com)

Daniel Ellard  
Raytheon BBN Technologies  
10 Moulton Street  
Cambridge, MA 02138  
US

Phone: +1 617-873-8004  
Email: [dellard@bbn.com](mailto:dellard@bbn.com)

Renu Tewari  
IBM Almaden  
650 Harry Rd  
San Jose, CA 95120  
US

Email: [tewarir@us.ibm.com](mailto:tewarir@us.ibm.com)

Manoj Naik  
IBM Almaden  
650 Harry Rd  
San Jose, CA 95120  
US

Email: [manoj@almaden.ibm.com](mailto:manoj@almaden.ibm.com)

