NFSv4.1 Server Implementation

Server support for minorversion 1 can be controlled using the /proc/fs/nfsd/versions control file. The string output returned by reading this file will contain either "+4.1" or "-4.1" correspondingly.

Currently, server support for minorversion 1 is disabled by default. It can be enabled at run time by writing the string "+4.1" to the /proc/fs/nfsd/versions control file. Note that to write this control file, the nfsd service must be taken down. Use your user-mode nfs-utils to set this up; see rpc.nfsd(8)

(Warning: older servers will interpret "+4.1" and "-4.1" as "+4" and "-4", respectively. Therefore, code meant to work on both new and old kernels must turn 4.1 on or off *before* turning support for version 4 on or off; rpc.nfsd does this correctly.)

The NFSv4 minorversion 1 (NFSv4.1) implementation in nfsd is based on RFC 5661.

From the many new features in NFSv4.1 the current implementation focuses on the mandatory-to-implement NFSv4.1 Sessions, providing "exactly once" semantics and better control and throttling of the resources allocated for each client.

Other NFSv4.1 features, Parallel NFS operations in particular, are still under development out of tree.
See http://wiki.linux-nfs.org/wiki/index.php/PNFS_prototype_design for more information.

The current implementation is intended for developers only: while it does support ordinary file operations on clients we have tested against (including the linux client), it is incomplete in ways which may limit features unexpectedly, cause known bugs in rare cases, or cause interoperability problems with future clients. Known issues:

- gss support is questionable: currently mounts with kerberos from a linux client are possible, but we aren't really conformant with the spec (for example, we don't use kerberos on the backchannel correctly).
- no trunking support: no clients currently take advantage of trunking, but this is a mandatory feature, and its use is recommended to clients in a number of places. (E.g. to ensure timely renewal in case an existing connection's retry timeouts have gotten too long; see section 8.3 of the RFC.) Therefore, lack of this feature may cause future clients to fail.
- Incomplete backchannel support: incomplete backchannel gss support and no support for BACKCHANNEL_CTL mean that callbacks (hence delegations and layouts) may not be available and clients confused by the incomplete implementation may fail.
- Server reboot recovery is unsupported; if the server reboots, clients may fail.
- We do not support SSV, which provides security for shared

client-server state (thus preventing unauthorized tampering with locks and opens, for example). It is mandatory for servers to support this, though no clients use it yet.

- Mandatory operations which we do not support, such as DESTROY_CLIENTID, FREE_STATEID, SECINFO_NO_NAME, and TEST_STATEID, are not currently used by clients, but will be (and the spec recommends their uses in common cases), and clients should not be expected to know how to recover from the case where they are not supported. This will eventually cause interoperability failures.

In addition, some limitations are inherited from the current NFSv4 implementation:

- Incomplete delegation enforcement: if a file is renamed or unlinked, a client holding a delegation may continue to indefinitely allow opens of the file under the old name.

The table below, taken from the NFSv4.1 document, lists the operations that are mandatory to implement (REQ), optional (OPT), and NFSv4.0 operations that are required not to implement (MNI) in minor version 1. The first column indicates the operations that are not supported yet by the linux server implementation.

The OPTIONAL features identified and their abbreviations are as follows:

pNFS	Parallel NFS
FDELG	File Delegations
DDELG	Directory Delegations

The following abbreviations indicate the linux server implementation status.

I	Implemented NFSv4.1 operations.
NS	Not Supported.
NS*	unimplemented optional feature.
P	pNFS features implemented out of tree.
PNS	pNFS features that are not supported yet (out of tree).

Operations

	Operation	REQ, REC, OPT, or MNI	Feature (REQ, REC, or OPT)	Definition
	ACCESS	REQ		Section 18.1
NS	BACKCHANNEL_CTL	REQ		Section 18.33
NS	BIND_CONN_TO_SESSION	REQ		Section 18.34
	CLOSE	REQ		Section 18.2
	COMMIT	REQ		Section 18.3
	CREATE	REQ		Section 18.4
I	CREATE_SESSION	REQ		Section 18.36
NS*	DELEGPURGE	OPT	FDELG (REQ)	Section 18.5
	DELEGRETURN	OPT	FDELG, DDELG, pNFS (REQ)	Section 18.6
NS	DESTROY_CLIENTID	REQ		Section 18.50
I	DESTROY_SESSION	REQ		Section 18.37

I	EXCHANGE_ID	REQ		Section 18.35
NS	FREE_STATEID	REQ		Section 18.38
	GETATTR	REQ		Section 18.7
P	GETDEVICEINFO	OPT	pNFS (REQ)	Section 18.40
P	GETDEVICELIST	OPT	pNFS (OPT)	Section 18.41
	GETFH	REQ		Section 18.8
NS*	GET_DIR_DELEGATION	OPT	DDELG (REQ)	Section 18.39
P	LAYOUTCOMMIT	OPT	pNFS (REQ)	Section 18.42
P	LAYOUTGET	OPT	pNFS (REQ)	Section 18.43
P	LAYOUTRETURN	OPT	pNFS (REQ)	Section 18.44
	LINK	OPT		Section 18.9
	LOCK	REQ		Section 18.10
	LOCKT	REQ		Section 18.11
	LOCKU	REQ		Section 18.12
	LOOKUP	REQ		Section 18.13
	LOOKUPP	REQ		Section 18.14
	NVERIFY	REQ		Section 18.15
	OPEN	REQ		Section 18.16
NS*	OPENATTR	OPT		Section 18.17
	OPEN_CONFIRM	MNI		N/A
	OPEN_DOWNGRADE	REQ		Section 18.18
	PUTFH	REQ		Section 18.19
	PUTPUBFH	REQ		Section 18.20
	PUTROOTFH	REQ		Section 18.21
	READ	REQ		Section 18.22
	READDIR	REQ		Section 18.23
	READLINK	OPT		Section 18.24
	RECLAIM_COMPLETE	REQ		Section 18.51
	RELEASE_LOCKOWNER	MNI		N/A
	REMOVE	REQ		Section 18.25
	RENAME	REQ		Section 18.26
	RENEW	MNI		N/A
	RESTOREFH	REQ		Section 18.27
	SAVEFH	REQ		Section 18.28
	SECINFO	REQ		Section 18.29
NS	SECINFO_NO_NAME	REC	pNFS files layout (REQ)	Section 18.45, Section 13.12
I	SEQUENCE	REQ		Section 18.46
	SETATTR	REQ		Section 18.30
	SETCLIENTID	MNI		N/A
	SETCLIENTID_CONFIRM	MNI		N/A
NS	SET_SSV	REQ		Section 18.47
NS	TEST_STATEID	REQ		Section 18.48
	VERIFY	REQ		Section 18.31
NS*	WANT_DELEGATION	OPT	FDELG (OPT)	Section 18.49
	WRITE	REQ		Section 18.32

Callback Operations

	Operation	REQ, REC, OPT, or MNI	Feature (REQ, REC, or OPT)	Definition
	CB_GETATTR	OPT	FDELG (REQ)	Section 20.1
P	CB_LAYOUTRECALL	OPT	pNFS (REQ)	Section 20.3

NS*	CB_NOTIFY	OPT	DDELG (REQ)	Section 20.4
P	CB_NOTIFY_DEVICEID	OPT	pNFS (OPT)	Section 20.12
NS*	CB_NOTIFY_LOCK	OPT		Section 20.11
NS*	CB_PUSH_DELEG	OPT	FDELG (OPT)	Section 20.5
	CB_RECALL	OPT	FDELG, DDELG, pNFS (REQ)	Section 20.2
NS*	CB_RECALL_ANY	OPT	FDELG, DDELG, pNFS (REQ)	Section 20.6
NS	CB_RECALL_SLOT	REQ		Section 20.8
NS*	CB_RECALLABLE_OBJ_AVAIL	OPT	DDELG, pNFS (REQ)	Section 20.7
I	CB_SEQUENCE	OPT	FDELG, DDELG, pNFS (REQ)	Section 20.9
NS*	CB_WANTS_CANCELLED	OPT	FDELG, DDELG, pNFS (REQ)	Section 20.10

Implementation notes:

DELEGPURGE:
* mandatory only for servers that support CLAIM_DELEGATE_PREV and/or CLAIM_DELEG_PREV_FH (which allows clients to keep delegations that persist across client reboots). Thus we need not implement this for now.

EXCHANGE_ID:
* only SP4_NONE state protection supported
* implementation ids are ignored

CREATE_SESSION:
* backchannel attributes are ignored
* backchannel security parameters are ignored

SEQUENCE:
* no support for dynamic slot table renegotiation (optional)

nfsv4.1 COMPOUND rules:
The following cases aren't supported yet:
* Enforcing of NFS4ERR_NOT_ONLY_OP for: BIND_CONN_TO_SESSION, CREATE_SESSION, DESTROY_CLIENTID, DESTROY_SESSION, EXCHANGE_ID.
* DESTROY_SESSION MUST be the final operation in the COMPOUND request.

Nonstandard compound limitations:
* No support for a sessions fore channel RPC compound that requires both a ca_maxrequestsize request and a ca_maxresponsesize reply, so we may fail to live up to the promise we made in CREATE_SESSION fore channel negotiation.
* No more than one IO operation (read, write, readdir) allowed per compound.