NAME

NFSv4 - NFS Version 4 Protocol

DESCRIPTION

The NFS client and server provides support for the NFSv4 specification; see *Network File System (NFS) Version 4 Protocol RFC 7530, Network File System (NFS) Version 4 Minor Version 1 Protocol RFC 5661, Network File System (NFS) Version 4 Minor Version 2 Protocol RFC 7862, File System Extended Attributes in NFSv4 RFC 8276 and Parallel NFS (pNFS) Flexible File Layout RFC 8435.* The protocol is somewhat similar to NFS Version 3, but differs in significant ways. It uses a single compound RPC that concatenates operations to-gether. Each of these operations are similar to the RPCs of NFS Version 3. The operations in the compound are performed in order, until one of them fails (returns an error) and then the RPC terminates at that point.

It has integrated locking support, which implies that the server is no longer stateless. As such, the **NFSv4** server remains in recovery mode for a grace period (always greater than the lease duration the server uses) after a reboot. During this grace period, clients may recover state but not perform other open/lock state changing operations. To provide for correct recovery semantics, a small file described by stablerestart(5) is used by the server during the recovery phase. If this file is missing or empty, there is a backup copy maintained by nfsd(8) that will be used. If either file is missing, they will be created by the nfsd(8). If both the file and the backup copy are empty, it will result in the server starting without providing a grace period for recovery. Note that recovery only occurs when the server machine is rebooted, not when the nfsd(8) are just restarted.

It provides several optional features not present in NFS Version 3:

- NFS Version 4 ACLs
- Referrals, which redirect subtrees to other servers
  (not yet implemented)
- Delegations, which allow a client to operate on a file locally
- pNFS, where I/O operations are separated from Metadata operations

And for NFSv4.2 only

- User namespace extended attributes
- lseek(SEEK_DATA/SEEK_HOLE)
- File copying done locally on the server for copy_file_range(2)
- posix_fallocate(2)
- posix_fadvise(POSIX_FADV_WILLNEED/POSIX_FADV_DONTNEED)

The **NFSv4** protocol does not use a separate mount protocol and assumes that the server provides a single file system tree structure, rooted at the point in the local file system tree specified by one or more

V4: <rootdir> [-sec=secflavors] [host(s) or net]

line(s) in the exports(5) file. (See exports(5) for details.) The nfsd(8) allows a limited subset of operations to be performed on non-exported subtrees of the local file system, so that traversal of the tree to the exported subtrees is possible. As such, the ‘‘<rootdir>’’ can be in a non-exported file system. The exception is ZFS, which checks exports and, as such, all ZFS file systems below the ‘‘<rootdir>’’ must be exported. However, the entire tree that is rooted at that point must be in local file systems that are of types that can be NFS exported. Since the **NFSv4** file system is rooted at ‘‘<rootdir>’’, setting this to anything other than ‘‘/’’ will result in clients being required to use different mount paths for **NFSv4** than for NFS Version 2 or 3. Unlike NFS Version 2 and 3, Version 4 allows a client mount to span across multiple server file systems, although not all clients are capable of doing this.

NFSv4 uses strings for users and groups instead of numbers. On the wire, these strings can either have the numbers in the string or take the form:

<user>@<dns.domain>

where ‘‘<dns.domain>’’ is not the same as the DNS domain used for host name lookups, but is usually set to the same string. Most systems set this ‘‘<dns.domain>’’ to the domain name part of the machine’s hostname(1) by default. However, this can normally be overridden by a command line option or configuration file for the daemon used to do the name<->number mapping. Under FreeBSD, the mapping daemon is called nfsuserd(8) and has a command line option that overrides the domain component of the machine’s hostname. For use of this form of string on **NFSv4**, either client or server, this daemon must be running.

The form where the numbers are in the strings can only be used for AUTH_SYS. To configure your systems this way, the nfsuserd(8) daemon does not need to be running on the server, but the following sysctls need to be set to 1 on the server.

vfs.nfs.enable_uidtostring
vfs.nfsd.enable_stringtouid

On the client, the sysctl

vfs.nfs.enable_uidtostring

must be set to 1 and the nfsuserd(8) daemon does not need to be running.

If these strings are not configured correctly, ‘‘ls -l’’ will typically report a lot of ‘‘nobody’’ and ‘‘nogroup’’ ownerships.

Although uid/gid numbers are no longer used in the **NFSv4** protocol except optionally in the above strings, they will still be in the RPC authentication fields when using AUTH_SYS (sec=sys), which is the default. As such, in this case both the user/group name and number spaces must be consistent between the client and server.

However, if you run **NFSv4** with RPCSEC_GSS (sec=krb5, krb5i, krb5p), only names and KerberosV tickets will go on the wire.

SERVER SETUP

To set up the NFS server that supports **NFSv4**, you will need to set the variables in rc.conf(5) as follows:

```
nfs_server_enable="YES"
nfsv4_server_enable="YES"
```

plus

```
nfsuserd_enable="YES"
```

if the server is using the ‘‘<user>@<domain>’’ form of user/group strings or is using the ‘‘-manage-gids’’ option for nfsuserd(8).

You will also need to add at least one ‘‘V4:’’ line to the exports(5) file for **NFSv4** to work.

If the file systems you are exporting are only being accessed via **NFSv4** there are a couple of sysctl(8) variables that you can change, which might improve performance.

vfs.nfsd.issue_delegations

when set non-zero, allows the server to issue Open Delegations to clients. These delegations permit the client to manipulate the file locally on the client. Unfortunately, at this time, client use of delegations is limited, so performance gains may not be observed. This can only be enabled when the file systems being exported to **NFSv4** clients are not being accessed locally on the server and, if being accessed via NFS Version 2 or 3 clients, these clients cannot be using the NLM.

vfs.nfsd.enable_locallocks

can be set to 0 to disable acquisition of local byte range locks. Disabling local locking can only be done if neither local accesses to the exported file systems nor the NLM is operating on them.

Note that Samba server access would be considered ‘‘local access’’ for the above discussion.

To build a kernel with the NFS server that supports **NFSv4** linked into it, the

```
options    NFSD
```

must be specified in the kernel's config(5) file.

CLIENT MOUNTS

To do an **NFSv4** mount, specify the ''nfsv4'' option on the mount_nfs(8) command line. This will force use of the client that supports **NFSv4** plus set ''tcp'' and **NFSv4**.

The nfsuserd(8) must be running if name<->uid/gid mapping is being used, as above. Also, since an **NFSv4** mount uses the host uuid to identify the client uniquely to the server, you cannot safely do an **NFSv4** mount when

```
hostid_enable="NO"
```

is set in rc.conf(5).

If the **NFSv4** server that is being mounted on supports delegations, you can start the nfscbd(8) daemon to handle client side callbacks. This will occur if

```
nfsuserd_enable="YES"        <-- If name<->uid/gid mapping is being used.
nfscbd_enable="YES"
```

are set in rc.conf(5).

Without a functioning callback path, a server will never issue Delegations to a client.

For NFSv4.0, by default, the callback address will be set to the IP address acquired via **rtalloc**() in the kernel and port# 7745. To override the default port#, a command line option for nfscbd(8) can be used.

To get callbacks to work when behind a NAT gateway, a port for the callback service will need to be set up on the NAT gateway and then the address of the NAT gateway (host IP plus port#) will need to be set by assigning the sysctl(8) variable vfs.nfs.callback_addr to a string of the form:

N.N.N.N.N.N

where the first 4 Ns are the host IP address and the last two are the port# in network byte order (all decimal #s in the range 0-255).

For NFSv4.1 and NFSv4.2, the callback path (called a backchannel) uses the same TCP connection as the mount, so none of the above applies and should work through gateways without any issues.

To build a kernel with the client that supports **NFSv4** linked into it, the option

```
options    NFSCL
```

must be specified in the kernel's config(5) file.

Options can be specified for the nfsuserd(8) and nfscbd(8) daemons at boot time via the ''nfsuserd_flags'' and ''nfscbd_flags'' rc.conf(5) variables.

NFSv4 mount(s) against exported volume(s) on the same host are not recommended, since this can result in a hung NFS server. It occurs when an nfsd thread tries to do an NFSv4 **VOP_RECLAIM**() / Close RPC as part of acquiring a new vnode. If all other nfsd threads are blocked waiting for lock(s) held by this nfsd thread, then there isn't an nfsd thread to service the Close RPC.

FILES

/var/db/nfs-stablerestart NFS V4 stable restart file
/var/db/nfs-stablerestart.bak backup copy of the file

SEE ALSO

stablerestart(5), mountd(8), nfscbd(8), nfsd(8), nfsdumpstate(8), nfsrevoke(8), nfsuserd(8)

BUGS

At this time, there is no recall of delegations for local file system operations. As such, delegations should only be enabled for file systems that are being used solely as NFS export volumes and are not being accessed via local system calls nor services such as Samba.