## NAME

**nfsd -** remote NFS server

## SYNOPSIS

**nfsd** [-**ardute**] [-**n** *num_servers*] [-**h** *bindip*] [-**p** *pnfs_setup*] [-**m** *mirror_level*] [-**V** *virtual_hostname*] [--**maxthreads** *max_threads*] [--**minthreads** *min_threads*]

## DESCRIPTION

The **nfsd** utility runs on a server machine to service NFS requests from client machines. At least one **nfsd** must be running for a machine to operate as a server.

Unless otherwise specified, eight servers per CPU for UDP transport are started.

When **nfsd** is run in an appropriately configured vnet jail, the server is restricted to TCP transport and no pNFS service. Therefore, the **-t** option must be specified and none of the **-u**, **-p** and **-m** options can be specified when run in a vnet jail. See jail(8) for more information.

The following options are available:

**-r** Register the NFS service with rpcbind(8) without creating any servers. This option can be used along with the **-u** or **-t** options to re-register NFS if the rpcbind server is restarted.

**-d** Unregister the NFS service with rpcbind(8) without creating any servers.

**-V** *virtual_hostname*

Specifies a hostname to be used as a principal name, instead of the default hostname.

**-n** *threads*

Specifies how many servers to create. This option is equivalent to specifying **--maxthreads** and **--minthreads** with their respective arguments to *threads*.

**--maxthreads** *threads*

Specifies the maximum servers that will be kept around to service requests.

**--minthreads** *threads*

Specifies the minimum servers that will be kept around to service requests.

**-h** *bindip*

Specifies which IP address or hostname to bind to on the local host. This option is recommended when a host has multiple interfaces. Multiple **-h** options may be specified.

**-a** Specifies that nfsd should bind to the wildcard IP address. This is the default if no **-h** options are given. It may also be specified in addition to any **-h** options given. Note that NFS/UDP does not operate properly when bound to the wildcard IP address whether you use -a or do not use -h.

**-p** *pnfs_setup*

Enables pNFS support in the server and specifies the information that the daemon needs to start it. This option can only be used on one server and specifies that this server will be the MetaData Server (MDS) for the pNFS service. This can only be done if there is at least one FreeBSD system configured as a Data Server (DS) for it to use.

The *pnfs_setup* string is a set of fields separated by ’,’ characters: Each of these fields specifies one DS. It consists of a server hostname, followed by a ’:’ and the directory path where the DS’s data storage file system is mounted on this MDS server. This can optionally be followed by a ’#’ and the mds_path, which is the directory path for an exported file system on this MDS. If this is specified, it means that this DS is to be used to store data files for this mds_path file system only. If this optional component does not exist, the DS will be used to store data files for all exported MDS file systems. The DS storage file systems must be mounted on this system before the **nfsd** is started with this option specified.
For example:

```
nfsv4-data0:/data0,nfsv4-data1:/data1
```

would specify two DS servers called nfsv4-data0 and nfsv4-data1 that comprise the data storage component of the pNFS service. These two DSs would be used to store data files for all exported file systems on this MDS. The directories "/data0" and "/data1" are where the data storage servers exported storage directories are mounted on this system (which will act as the MDS).
Whereas, for the example:

```
nfsv4-data0:/data0#/export1,nfsv4-data1:/data1#/export2
```

would specify two DSs as above, however nfsv4-data0 will be used to store data files for "/export1" and nfsv4-data1 will be used to store data files for "/export2".

When using IPv6 addresses for DSs be wary of using link local addresses. The IPv6 address for the DS is sent to the client and there is no scope zone in it. As such, a link local address may not work for a pNFS client to DS TCP connection. When parsed, **nfsd** will only use a link local address if it is the only address returned by getaddrinfo(3) for the DS hostname.

**-m** *mirror_level*

This option is only meaningful when used with the **-p** option. It specifies the "mirror_level", which defines how many of the DSs will have a copy of a file's data storage file. The default of one implies no mirroring of data storage files on the DSs. The "mirror_level" would normally be set to 2 to enable mirroring, but can be as high as NFSDEV_MAXMIRRORS. There must be at least "mirror_level" DSs for each exported file system on the MDS, as specified in the **-p** option. This implies that, for the above example using "#/export1" and "#/export2", mirroring cannot be done. There would need to be two DS entries for each of "#/export1" and "#/export2" in order to support a "mirror_level" of two.

If mirroring is enabled, the server must use the Flexible File layout. If mirroring is not enabled, the server will use the File layout by default, but this default can be changed to the Flexible File layout if the sysctl(8) vfs.nfsd.default_flexfile is set non-zero.

**-t** Serve TCP NFS clients.

**-u** Serve UDP NFS clients.

**-e** Ignored; included for backward compatibility.

For example, "nfsd -u -t -n 6" serves UDP and TCP transports using six daemons.

A server should run enough daemons to handle the maximum level of concurrency from its clients, typically four to six.

The **nfsd** utility listens for service requests at the port indicated in the NFS server specification; see *Network File System Protocol Specification*, RFC1094, *NFS: Network File System Version 3 Protocol Specification*, RFC1813, *Network File System (NFS) Version 4 Protocol*, RFC7530, *Network File System (NFS) Version 4 Minor Version 1 Protocol*, RFC5661, *Network File System (NFS) Version 4 Minor Version 2 Protocol*, RFC7862, *File System Extended Attributes in NFSv4*, RFC8276 and *Parallel NFS (pNFS) Flexible File Layout*, RFC8435.

If **nfsd** detects that NFS is not loaded in the running kernel, it will attempt to load a loadable kernel module containing NFS support using kldload(2). If this fails, or no NFS KLD is available, **nfsd** will exit with an error.

If **nfsd** is to be run on a host with multiple interfaces or interface aliases, use of the **-h** option is recommended. If you do not use the option NFS may not respond to UDP packets from the same IP address they were sent to. Use of this option is also recommended when securing NFS exports on a firewalling machine such that the NFS sockets can only be accessed by the inside interface. The **ipfw** utility would then be used to block NFS-related packets that come in on the outside interface.

If the server has stopped servicing clients and has generated a console message like "nfsd server cache flooded...", the value for vfs.nfsd.tcphighwater needs to be increased. This should allow the server to again handle requests without a reboot. Also, you may want to consider decreasing the value for vfs.nfsd.tcpcachetimeo to several minutes (in seconds) instead of 12 hours when this occurs.

Unfortunately making vfs.nfsd.tcphighwater too large can result in the mbuf limit being reached, as indicated by a console message like "kern.ipc.nmbufs limit reached". If you cannot find values of the above **sysctl** values that work, you can disable the DRC cache for TCP by setting vfs.nfsd.cachetcp to 0.

The **nfsd** utility has to be terminated with SIGUSR1 and cannot be killed with SIGTERM or SIGQUIT. The **nfsd** utility needs to ignore these signals in order to stay alive as long as possible during a shutdown, otherwise loopback mounts will not be able to unmount. If you have to kill **nfsd** just do a "kill -USR1 <PID of master nfsd>"

## EXIT STATUS

The **nfsd** utility exits 0 on success, and >0 if an error occurs.

## SEE ALSO

nfsstat(1), kldload(2), nfssvc(2), nfsv4(4), pnfs(4), pnfsserver(4), exports(5), stablerestart(5), gssd(8), ipfw(8), jail(8), mountd(8), nfsiod(8), nfsrevoke(8), nfsuserd(8), rpcbind(8)

## HISTORY

The **nfsd** utility first appeared in 4.4BSD.

## BUGS

If **nfsd** is started when gssd(8) is not running, it will service AUTH_SYS requests only. To fix the problem you must kill **nfsd** and then restart it, after the gssd(8) is running.

For a Flexible File Layout pNFS server, if there are Linux clients doing NFSv4.1 or NFSv4.2 mounts, those clients might need the sysctl(8) vfs.nfsd.flexlinuxhack to be set to one on the MDS as a workaround.

Linux 5.n kernels appear to have been patched such that this sysctl(8) does not need to be set.