## NAME

rpc.statd - NSM service daemon

## SYNOPSIS

**rpc.statd [-dh?FLNvV] [-H** *prog***] [-n** *my-name***] [-o** *outgoing-port***]**
**[-p** *listener-port***] [-P** *path***]**
**[--nlm-port** *port***] [--nlm-udp-port** *port***]**

## DESCRIPTION

File locks are not part of persistent file system state. Lock state is thus lost when a host reboots.

Network file systems must also detect when lock state is lost because a remote host has rebooted. After an NFS client reboots, an NFS server must release all file locks held by applications that were running on that client. After a server reboots, a client must remind the server of file locks held by applications running on that client.

For NFS version 2 [RFC1094] and NFS version 3 [RFC1813], the *Network Status Monitor* protocol (or NSM for short) is used to notify NFS peers of reboots. On Linux, two separate user-space components constitute the NSM service:

**rpc.statd**
A daemon that listens for reboot notifications from other hosts, and manages the list of hosts to be notified when the local system reboots

**sm-notify**
A helper program that notifies NFS peers after the local system reboots

The local NFS lock manager alerts its local **rpc.statd** of each remote peer that should be monitored. When the local system reboots, the **sm-notify** command notifies the NSM service on monitored peers of the reboot. When a remote reboots, that peer notifies the local**r pc.statd**, which in turn passes the reboot notification back to the local NFS lock manager.

## NSM OPERATION IN DETAIL

The first file locking interaction between an NFS client and server causes the NFS lock managers on both peers to contact their local NSM service to store information about the opposite peer. On Linux, the local lock manager contacts **rpc.statd**.

**rpc.statd** records information about each monitored NFS peer on persistent storage. This information describes how to contact a remote peer in case the local system reboots, how to recognize which monitored peer is reporting a reboot, and how to notify the local lock manager when a monitored peer indicates it has rebooted.

An NFS client sends a hostname, known as the client's *caller_name*, in each file lock request. An NFS server can use this hostname to send asynchronous GRANT calls to a client, or to notify the client it has rebooted.

The Linux NFS server can provide the client's *caller_name* or the client's network address to **rpc.statd**. For the purposes of the NSM protocol, this name or address is known as the monitored peer's *mon_name*. In addition, the local lock manager tells **rpc.statd** what it thinks its own hostname is. For the purposes of the NSM protocol, this hostname is known as *my_name*.

There is no equivalent interaction between an NFS server and a client to inform the client of the server's *caller_name*. Therefore NFS clients do not actually know what *mon_name* an NFS server might use in an SM_NOTIFY request. The Linux NFS client uses the server hostname from the mount command to identify rebooting NFS servers.

### Reboot notification

When the local system reboots, the **sm-notify** command reads the list of monitored peers from persistent storage and sends an SM_NOTIFY request to the NSM service on each listed remote peer. It uses the *mon_name* string as the destination. To identify which host has rebooted, the **sm-notify** command sends the *my_name* string recorded when that remote was monitored. The remote **rpc.statd** matches incoming SM_NOTIFY requests using this string, or the caller's network address, to one or more peers on its own

monitor list.

If **rpc.statd** does not find a peer on its monitor list that matches an incoming SM_NOTIFY request, the notification is not forwarded to the local lock manager. In addition, each peer has its own *NSM state number*, a 32-bit integer that is bumped after each reboot by the **sm-notify** command. **rpc.statd** uses this number to distinguish between actual reboots and replayed notifications.

Part of NFS lock recovery is rediscovering which peers need to be monitored again. The**sm-notify** command clears the monitor list on persistent storage after each reboot.

## OPTIONS

**-d**, **--no-syslog**
Causes **rpc.statd** to write log messages on *stderr* instead of to the system log, if the **-F** option was also specified.

**-F**, **--foreground**
Keeps **rpc.statd** attached to its controlling terminal so that NSM operation can be monitored directly or run under a debugger. If this option is not specified,**r pc.statd** backgrounds itself soon after it starts.

**-h**, **-?**, **--help**
Causes **rpc.statd** to display usage information on *stderr* and then exit.

**-H, --ha-callout** *prog*
Specifies a high availability callout program. If this option is not specified, no callouts are performed. See the**High-a vailability callouts** section below for details.

**-L**, **--no-notify**
Prevents **rpc.statd** from running the **sm-notify** command when it starts up, preserving the existing NSM state number and monitor list.

Note: the **sm-notify** command contains a check to ensure it runs only once after each system reboot. This prevents spurious reboot notification if **rpc.statd** restarts without the **-L** option.

**-n, --name** *ipaddr* | *hostname*
Specifies the bind address used for RPC listener sockets. The*ipaddr* form can be e xpressed as either an IPv4 or an IPv6 presentation address. If this option is not specified, **rpc.statd** uses a wildcard address as the transport bind address.

This string is also passed to the **sm-notify** command to be used as the source address from which to send reboot notification requests. See sm-notify(8) for details.

**-N** Causes **rpc.statd** to run the **sm-notify** command, and then exit. Since the**sm-notify** command can also be run directly, this option is deprecated.

**-o, --outgoing-port** *port*
Specifies the source port number the **sm-notify** command should use when sending reboot notifications. See sm-notify(8) for details.

**-p, --port** *port*
Specifies the port number used for RPC listener sockets. If this option is not specified,**r pc.statd** will try to consult */etc/services*, if gets port succeed, set the same port for all listener socket, otherwise chooses a random ephemeral port for each listener socket.

This option can be used to fix the port value of its listeners when SM_NOTIFY requests must traverse a firewall between clients and servers.

**-T, --nlm-port** *port*
Specifies the port number that *lockd* should listen on for **NLM** requests. This sets both the TCP and UDP ports unless the UDP port is set separately.

**-U, --nlm-udp-port** *port*
Specifies the UDP port number that *lockd* should listen on for **NLM** requests.

**-P, --state-directory-path** *pathname*
Specifies the pathname of the parent directory where NSM state information resides. If this option is not specified, **rpc.statd** uses */var/lib/nfs* by default.

After starting, **rpc.statd** attempts to set its effective UID and GID to the owner and group of this directory.

**-v**, **-V**, **--version**
Causes **rpc.statd** to display version information on *stderr* and then exit.

## SECURITY

The **rpc.statd** daemon must be started as root to acquire privileges needed to create sockets with privileged source ports, and to access the state information database. Because **rpc.statd** maintains a long-running network service, however, it drops root privileges as soon as it starts up to reduce the risk of a privilege escalation attack.

During normal operation, the effective user ID it chooses is the owner of the state directory. This allows it to continue to access files in that directory after it has dropped its root privileges. To control which user ID **rpc.statd** chooses, simply use chown(1) to set the owner of the state directory.

You can also protect your **rpc.statd** listeners using the **tcp_wrapper** library or iptables(8). To use the **tcp_wrapper** library, add the hostnames of peers that should be allowed access to */etc/hosts.allow*. Use the daemon name **statd** even if the **rpc.statd** binary has a different filename.

For further information see the **tcpd(8)** and hosts_access(5) man pages.

## ADDITIONAL NOTES

Lock recovery after a reboot is critical to maintaining data integrity and preventing unnecessary application hangs. To help **rpc.statd** match SM_NOTIFY requests to NLM requests, a number of best practices should be observed, including:

The UTS nodename of your systems should match the DNS names that NFS peers use to contact them

The UTS nodenames of your systems should always be fully qualified domain names

The forward and reverse DNS mapping of the UTS nodenames should be consistent

The hostname the client uses to mount the server should match the server's *mon_name* in SM_NOTIFY requests it sends

Unmounting an NFS file system does not necessarily stop either the NFS client or server from monitoring each other. Both may continue monitoring each other for a time in case subsequent NFS traffic between the two results in fresh mounts and additional file locking.

On Linux, if the **lockd** kernel module is unloaded during normal operation, all remote NFS peers are unmonitored. This can happen on an NFS client, for example, if an automounter removes all NFS mount points due to inactivity.

### High-availability callouts

**rpc.statd** can exec a special callout program during processing of successful SM_MON, SM_UNMON, and SM_UNMON_ALL requests, or when it receives SM_NOTIFY. Such a program may be used in High Availability NFS (HA-NFS) environments to track lock state that may need to be migrated after a system reboot.

The name of the callout program is specified with the **-H** option. The program is run with 3 arguments: The first is either **add-client del-client** or **sm-notify** depending on the reason for the callout. The second is the *mon_name* of the monitored peer. The third is the*caller_name* of the requesting lock manager for **add-client** or **del-client** , otherwise it is *IP_address* of the caller sending SM_NOTIFY. The forth is the *state_value* in the SM_NOTIFY request.

### IPv6 and TI-RPC support

TI-RPC is a pre-requisite for supporting NFS on IPv6. If TI-RPC support is built into **rpc.statd**, it attempts to start listeners on network transports marked 'visible' in */etc/netconfig*. As long as at least one network

transport listener starts successfully, **rpc.statd** will operate.

## FILES

| | |
|---|---|
| */var/lib/nfs/sm* | directory containing monitor list |
| */var/lib/nfs/sm.bak* | directory containing notify list |
| */var/lib/nfs/state* | NSM state number for this host |
| */run/run.statd.pid* | pid file |
| */etc/netconfig* | network transport capability database |

## SEE ALSO

sm-notify(8), nfs(5), rpc.nfsd(8), rpcbind(8), **tcpd(8),** hosts_access(5), iptables(8), netconfig(5)

RFC 1094 - "NFS: Network File System Protocol Specification"
RFC 1813 - "NFS Version 3 Protocol Specification"
OpenGroup Protocols for Interworking: XNFS, Version 3W - Chapter 11

## AUTHORS

Jeff Uphoff <juphoff@users.sourceforge.net>
Olaf Kirch <okir@monad.swb.de>
H.J. Lu <hjl@gnu.org>
Lon Hohberger <hohberger@missioncriticallinux.com>
Paul Clements <paul.clements@steeleye.com>
Chuck Lever <chuck.lever@oracle.com>