

GlusterFS NFS Pre-Beta Release Notes

Jun 15th, 2010

Table of Contents

Introduction.....	2
Disclaimer.....	2
Pre-Installation Information.....	2
Product License.....	2
System Requirements.....	2
Installation.....	3
Installing from RPMs.....	3
Downloading Source Tarball.....	3
Downloading from Source Checkout.....	3
Building from Source.....	3
Co-locating NFS Beta with Existing GlusterFS Version.....	3
Compatibility.....	4
Upgrade.....	4
User Guide.....	4
Quick Start.....	4
Exporting 4 bricks as a distributed volume using glusterfs-volgen.....	4
Exporting 2 bricks as a replicated volume using glusterfs-volgen.....	6
Exporting 4 bricks as a distributed-replicated volume using glusterfs-volgen.....	7
Example Posix backend.....	8
Exporting a single GlusterFS volume through NFS.....	9
Exporting multiple GlusterFS volumes through NFS.....	10
Exporting a single GlusterFS volume to a single NFS client.....	12
Exporting a single GlusterFS volume to a subnet of clients.....	13
Exporting a directory in GlusterFS volume as an NFS export.....	14
NFS Translator Options.....	15
Filing Defects.....	18
Product Documentation.....	18
Frequently Asked Questions.....	18
Copyright/Trademarks.....	21

Introduction

Development of native NFS support in GlusterFS was motivated by the functional and performance limitations imposed by the currently available NFS servers, namely, the kernel NFS server and unfsd. Both servers impose significant performance overhead because of the need to use FUSE between the NFS server and GlusterFS process.

Native NFS for GlusterFS is a translator, in GlusterFS terminology, thus completely avoiding the overhead of FUSE and of switching between GlusterFS and the NFS servers. Like all other translators, NFS can be added to an existing volume file in order to export any of the volumes in the volfile to NFS clients. On one hand, existing GlusterFS volumes can be exported through NFS by addition of only 4 to 5 lines in the volfile and on the other, new volume files can be generated using the NFS support in *glusterfs-volgen* tool.

Beta version of NFS translator supports NFSv3 and MOUNTv3. Support for NLMv4, the NFSv3 locking protocol, will be introduced in a future release.

Beta release has been tested to be interoperable with NFS clients of Linux, Solaris and Vmware.

We look forward to the community trying out other implementations of NFS clients and virtualization platforms that utilize NFS. Please email any feedback to us at:

nfs-alpha@gluster.com.

Disclaimer

This GlusterFS release that includes NFS translator in Beta state is provided AS IS. We recommend that you do not use this on production setups and that Gluster, Inc, is in no way responsible for any loss of data or services.

Pre-Installation Information

Product License

GlusterFS is released under GNU General Public License v3 or later. Documentation is released under GNU Free Documentation License 1.2 or later. The license for GlusterFS NFS Beta can be viewed at <http://www.gluster.com/company/legal.php>

System Requirements

System requirements are same as those documented in the Release Notes for GlusterFS 3.0.3 release available at:

http://ftp.gluster.com/pub/gluster/glusterfs/3.0/3.0.3/GlusterFS_3.0.3_Release_Notes.pdf

Installation

Installing from RPMs

FIXME: RPMs for the NFS Alpha release are available for Fedora 11 and Centos 5 at:

<http://ftp.gluster.com/pub/gluster/glusterfs/qa-releases/nfs-beta/>

Instructions to setup GlusterFS from RPMs are available at:

http://www.gluster.com/community/documentation/index.php/Storage_Server_Installation_and_Configuration

Downloading Source Tarball

Although the following source tarball can be used for a fresh build and install, a later section provides instructions for setting up NFS Beta release on machines which have an existing GlusterFS installation. GlusterFS NFS Beta source tarball is available at:

<http://ftp.gluster.com/pub/gluster/glusterfs/qa-releases/nfs-beta/glusterfs-nfs-beta-rc7.tar.gz>

Downloading from Source Checkout

Latest Beta code can be obtained from the source code repository. Although the following source checkout can be used for a fresh build and install, the next section provides instructions for setting up NFS Beta release on a machines which have an existing GlusterFS installation. The git source code manager is needed to get the source from the repository.

```
$ git clone git://git.gluster.com/users/shehjart/gluster-nfs-beta.git
gluster-nfs-beta
$ cd gluster-nfs-beta
```

Building from Source

To build the source downloaded either from source checkout or from the tarball, please use the commands below in the source directory.

```
$ ./autogen.sh
$ ./configure
$ make install;
```

Co-locating NFS Beta with Existing GlusterFS Version

To co-locate the NFS Beta release with an existing version to facilitate testing, the Gluster NFS Beta source needs to be downloaded using either the tarball approach or directly from the repository using git. Next, in the source directory:

```
$ ./autogen.sh
$ ./configure --prefix=<NEW-GLUSTERFS-DIRECTORY>
$ make install;
```

Now, the GlusterFS NFS Beta release binary resides in<NEW-GLUSTERFS-DIRECTORY>.

As described in detail in a later section, existing volumes files will need modifications to enable volumes defined in them to be exported via NFS.

Compatibility

NFS Beta release requires that both the GlusterFS NFS server as well as the GlusterFS bricks run this same Beta version. No other GlusterFS versions are supported. NFS clients must support NFS version 3.

Upgrade

If users of GlusterFS 2.0.x and 3.0.x want to use GlusterFS NFS Beta, they need to install all GlusterFS servers and clients with GlusterFS NFS Beta software.

Gluster recommends users backup their prior configuration and volume files prior to installing GlusterFS NFS Alpha.

User Guide

GlusterFS NFS server translator is a new translator that provides NFS version 3 file serving functionality for GlusterFS volumes. NFS server can be configured using the *glusterfs-volgen* tool. For advanced users, the following sections also show how to write a volume file by hand. Doing so involves adding the NFS translator section to the vol files.

Quick Start

The following *glusterfs-volgen* commands generate the volume files for various configurations of GlusterFS. Each of these configurations also contains the pieces required to export the GlusterFS volumes through NFS.

Exporting 4 bricks as a distributed volume using glusterfs-volgen

To export a 4-brick distributed GlusterFS volume through NFS, the command below generates the required volume files for bricks and the NFS server.

```
$ glusterfs-volgen -n nfsexport --nfs srv1:/export srv2:/export srv3:/export
srv4:/export
```

The command above assumes that the servers are addressed by the hostnames `srv1`, `srv2`, `srv3` and `srv4` and that each of the servers have a directory `/export` that will be exported through a distributed configuration.

The end product of the command will be 5 files:

```
srv1-nfsexport-export.vol
srv3-nfsexport-export.vol
srv2-nfsexport-export.vol
srv4-nfsexport-export.vol
nfsexport-tcp.vol
```

Next, we'll use these files as input when starting GlusterFS on each of these servers. To start a GlusterFS server on `srv1`:

```
$ glusterfsd -f srv1-nfsexport-export.vol
```

Similarly for the remaining servers, the above `glusterfsd` command needs to be executed with the corresponding volume file.

Once the GlusterFS servers have been started, the Gluster NFS server needs to be started up. Please use the command below on the machine that is the designated NFS server. This machine can be the same as one of the bricks.

```
$ glusterfsd -f nfsexport-tcp.vol -l /tmp/nfssrv.log
```

Please note that any or all of the bricks can serve as a NFS server, as long as the machine contains a Gluster NFS Beta installation . In contrast, a NFS client will be able to mount only one of these servers. In case the NFS server is run on multiple machines, the namespace exported through NFS will be the same across all the NFS servers.

To reduce the number of servers in your setup, exclude the required number of servers from the command line when running the `glusterfs-volgen` command.

Once the GlusterFS servers and Gluster NFS server has been started , the following command on the NFS client mounts the distributed volume. The command assumes that the NFS server is started on `srv4`.

```
$ mount srv4:/distribute /mnt
```

Exporting 2 bricks as a replicated volume using glusterfs-volgen

Two bricks can be exported as a replicated GlusterFS volume through NFS too. The `glusterfs-volgen` command below generates the volume files required for such a configuration.

```
$ glusterfs-volgen -n nfsexport --raid 1 --nfs srv1:/export srv2:/export
```

The resultant output files will be:

```
srv1-nfsexport-export.vol  
srv2-nfsexport-export.vol  
nfsexport-tcp.vol
```

Once the volume files have been generated, we'll start the GlusterFS server on the bricks. On `srv1` the command will be:

```
$ glusterfsd -f srv1-nfsexport-export.vol
```

On `srv2`, a similar command will be run with `srv2-nfsexport-export.vol` in place of `srv1-nfsexport-export.vol`.

Next, we need to start the Gluster NFS server. Please note that Gluster NFS server can be started on either of these two servers or both or even on a third machine, as long as the machine contains a Gluster NFS Beta installation. Regardless of the machine, the following command will start the NFS server program:

```
$ glusterfsd -f nfsexport-tcp.vol -l /tmp/nfssrv.log
```

Assuming the Gluster NFS server was started on `srv2`, a NFS client can run the command below to mount an NFS export:

```
$ mount srv2:/mirror-0 /mnt
```

Exporting 4 bricks as a distributed-replicated volume using glusterfs-volgen

4 bricks can be exported as a distributed-replicated GlusterFS volume through NFS too while all data is replicated over two of the four bricks. The *glusterfs-volgen* command below generates the volume files required for such a configuration.

```
$ glusterfs-volgen -n nfsexport --raid 1 --nfs srv1:/export srv2:/export  
srv3:/export srv4:/export
```

The resultant output files will be:

```
srv1-nfsexport-export.vol  
srv3-nfsexport-export.vol  
srv2-nfsexport-export.vol  
srv4-nfsexport-export.vol  
nfsexport-tcp.vol
```

Once the volume files have been generated, we'll start the GlusterFS server on the bricks. On *srv1* the command will be:

```
$ glusterfsd -f srv1-nfsexport-export.vol
```

On *srv2*, a similar command will be run with *srv2-nfsexport-export.vol* in place of *srv1-nfsexport-export.vol* and so on.

Next, we need to start the Gluster NFS server. Please note that Gluster NFS server can be started on any of these four servers or on all four or even on a fifth machine, as long as the machine contains a Gluster NFS Beta installation. Regardless of the machine, the following command will start the NFS server program:

```
$ glusterfsd -f nfsexport-tcp.vol -l /tmp/nfssrv.log
```

Assuming the Gluster NFS server was started on *srv2*, a NFS client can run the command below to mount an NFS export:

```
$ mount srv2:/distribute /mnt
```

In the sections that follow, we present sample volume files so that advanced users may understand the new elements needed in GlusterFS volume files in order to export GlusterFS volumes as NFS exports.

Example Posix backend

For the sample configurations that follow, the volume file below will be used for a GlusterFS brick.

```
volume posix
    type storage/posix
    option directory /export/homes
end-volume
volume posix-ac
    type features/access-control
    subvolumes posix
end-volume
volume posix-locked
    type features/locks
    subvolumes posix-ac
end-volume
volume brick
    type performance/io-threads
    subvolumes posix-locked
end-volume
volume posix-server
    type protocol/server
    subvolumes brick
    option transport-type tcp
    option transport.socket.listen-port 6102
    option auth.addr.brick.allow *
end-volume
```

For the purpose of this test, let us call this file `posix.vol`. The exported directory, `/export/homes` may be replaced with the directory that needs to be exported from a brick in your setup.

In all cases, a GlusterFS server can be started using the above volume file by the command below:

```
$ glusterfsd -f posix.vol -l /tmp/nfs-posix.log
```

Exporting a single GlusterFS volume through NFS

Following is a volume file importing posix backends from two GlusterFS servers and combining them in a replicate configuration. The replicated volume is then exported using the NFS translator. In this configuration, a GlusterFS server using the `posix.vol` above will have to be run on two servers in order to correspond to the two replicas.

```
volume client0
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.101
    option remote-subvolume brick
end-volume
volume client1
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.102
    option remote-subvolume brick
end-volume
volume repl-0-1
    type cluster/replicate
    subvolumes client0 client1
end-volume
#Export above replicate through NFS
volume nfs-server
    type nfs/server
    subvolumes repl-0-1
    option rpc-auth.addr.allow *
end-volume
```

Suppose the volume configuration above is copied into a file called `nfssrv.vol`, then the NFS server can be started by:

```
$ glusterfsd -f nfssrv.vol -l /tmp/nfssrv.log
```

On a Linux NFS client, the replicate export will be mounted by the command:

```
$ mount nfsserver:/repl-0-1 /mnt
```

Exporting multiple GlusterFS volumes through NFS

Suppose the NFS server needs to export two GlusterFS volumes as described by the volume file below:

```
volume client0
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.101
    option remote-subvolume brick
end-volume
volume client1
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.102
    option remote-subvolume brick
end-volume

volume repl-0-1
    type cluster/replicate
    subvolumes client0 client1
end-volume
volume client2
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.103
```

```

        option remote-subvolume brick
    end-volume
    volume client3
        type protocol/client
        option transport-type tcp
        option remote-port 6102
        option remote-host 192.168.1.104
        option remote-subvolume brick
    end-volume
    volume dist-2-3
        type cluster/distribute
        subvolumes client2 client 3
    end-volume
    #Export above replicate and distribute as two
    # separate volumes through NFS
    volume nfs-server
        type nfs/server
        subvolumes repl-0-1 dist-2-3
        option rpc-auth.addr.allow *
    end-volume

```

Suppose the volume configuration above is copied into a file called `nfssrv.vol`, then the NFS server can be started by:

```
$ glusterfsd -f nfssrv.vol -l /tmp/nfssrv.log
```

On a Linux NFS client, the two exports can be mounted by the command:

```

$ mount nfsserver:/repl-0-1 /mnt
$ mount nfsserver:/dist-2-3 /mnt2

```

Exporting a single GlusterFS volume to a single NFS client

The NFS translator allows restricting the exports to certain client machines using the `rpc-auth.addr.allow` option of the NFS translator.

```
volume client0
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.101
    option remote-subvolume brick
end-volume
volume client1
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.102
    option remote-subvolume brick
end-volume
volume repl-0-1
    type cluster/replicate
    subvolumes client0 client1
end-volume
#Export above replicate through NFS
volume nfs-server
    type nfs/server
    subvolumes repl-0-1
    option rpc-auth.addr.allow 192.168.1.107
end-volume
```

Suppose the volume configuration above is copied into a file called `nfssrv.vol`, then the NFS server can be started by:

```
$ glusterfsd -f nfssrv.vol -l /tmp/nfssrv.log
```

On a Linux NFS client machine 192.168.1.107, the replicate export will be mounted by the command:

```
$ mount nfsserver:/repl-0-1 /mnt
```

For all other machines, the mount request will be denied.

Exporting a single GlusterFS volume to a subnet of clients

The NFS translator allows restricting the exports to certain client machines.

```
volume client0
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.101
    option remote-subvolume brick
end-volume
volume client1
    type protocol/client
    option transport-type tcp
    option remote-port 6102
    option remote-host 192.168.1.102
    option remote-subvolume brick
end-volume
volume repl-0-1
    type cluster/replicate
    subvolumes client0 client1
end-volume
#Export above replicate through NFS
volume nfs-server
    type nfs/server
    subvolumes repl-0-1
    option rpc-auth.addr.allow 192.168.1.*
end-volume
```

Suppose the volume configuration above is copied into a file called `nfssrv.vol`, then the NFS

server can be started by:

```
$ glusterfsd -f nfssrv.vol -l /tmp/nfssrv.log
```

On a Linux NFS client machine on the given subnet, the replicate export will be mounted by the command:

```
$ mount nfsserver:/repl-0-1 /mnt
```

For machines with IP addresses other than 192.168.1.*, the mount request will be denied.

Exporting a directory in GlusterFS volume as an NFS export

Gluster NFS server allows exporting volumes as well as individual directories within GlusterFS volumes. Directory exports are useful in situations where users only need to mount specific directories and not whole volumes.

To enable directory export, an option needs to be enabled in the NFS volume section as shown below.

```
#Export a replicate through NFS
volume nfs-server
    type nfs/server
    subvolumes repl-0-1
    option nfs3.repl-0-1.export-dir /johndoe
    option rpc-auth.addr.allow 192.168.1.*
end-volume
```

The above volume section exports a volume called `repl-0-1` and assumes that a directory `/johndoe` exists on that volume. Assuming that we used the server volume file shown in section “Example posix backend”, the server directory that will be exported will be the `/export/homes/johndoe`.

There are use cases where only the given directories need to be visible to the users in a given volume. This requires that the remaining volume not be exported through NFS. The following NFS volume option disables export of complete volumes.

```
option nfs3.repl-0-1.export-volume off
```

Volume export is enabled by default for all volumes.

At the NFS client, the mount command for mounting the exported directory will be:

```
$ mount <nfs-server>:/repl-0-1/johndoe /mnt
```

NFS Translator Options

Option	Type	Description
rpc-auth.auth-unix	Boolean eg. <onloff>	Use this option to enable or disable the AUTH_UNIX authentication scheme for all exported volumes. By default, this scheme is enabled for all exports and in fact, some NFS clients will not allow mounting an export without AUTH_UNIX enabled. Enabled by default.
rpc-auth.auth-null	Boolean eg. <onloff>	Use this option to enable or disable the AUTH_NULL authentication scheme for all exported volumes. By default, this scheme is enabled for all exports and turn it off if you require that all NFS clients be forced to use a better authentication scheme since AUTH_NULL is basically no authentication. Enabled by default.
rpc-auth.auth-null.*	Boolean eg. <onloff>	Use this option to enable or disable the AUTH_NULL authentication scheme for a specific export. By default, this scheme is enabled for all exports. This per-volume option can be used to over-ride the global value set using the previous two options.
rpc-auth.auth-unix.*	Boolean eg. <onloff>	Use this option to enable or disable the AUTH_UNIX authentication scheme for a specific export. By default, this scheme is enabled for all

		exports. This per-volume option can be used to over-ride the global value set using the previous two options.
rpc-auth.addr.reject	* or comma-separated list of Ipv4 addresses or hostnames	Reject only the given addresses/hostnames
rpc-auth.addr.allow	* or comma-separated list of Ipv4 addresses or hostnames	Allow only the given addresses/hostnames
rpc-auth.addr.<export>.allow	* or comma-separated list of Ipv4 addresses or hostnames	Allow access to <export> only from the given addresses/hostnames.
rpc-auth.addr.<export>.reject	* or comma-separated list of Ipv4 addresses or hostnames	Reject access to <export> only from the given addresses/hostnames.
rpc-auth.ports.insecure	Boolean, eg, <on/off>	Allow clients to mount exports from unprivileged ports.
rpc-auth.ports.<export>.insecure	Boolean, eg, <on/off>	Allow clients to only mount <export> from unprivileged ports.
nfs3.<export>.volume-access	String: “read-only” or “read-write”	Read-write access to <export> can be restricted using this option.
rpc-auth.addr.namelookup	Boolean, eg. <on/off>	Users have the option of turning off name lookup for incoming client connections using this option. In some setups, the name server can take too long to reply to DNS queries resulting in timeouts of mount requests. Use this option to turn off name lookups during address authentication. Note, turning this off will prevent you from using hostnames in rpc-auth.addr.* filters. By default name lookup is on.
New Options Since Alpha		
nfs3.<export>.trusted-write	Boolean, eg. <on off>	In some environments, combined with a replicated GlusterFS setup, this option can improve write performance.

		This flag allows user to trust Gluster replication logic to sync data to the disks and recover when required. Sync or COMMIT requests, if received will be handled in a default manner by fsyncing. IN NFS protocol jargon, on an UNSTABLE write from client, return STABLE flag to force client to not send a COMMIT request. STABLE writes are still handled in a synchronous manner. Off by default.
nfs3.<export>.trusted-sync	Boolean, eg. <on off>	All writes and COMMIT requests are treated as async. This implies that no write requests are guaranteed to be on server disks when the write reply is received at the NFS client. Trusted sync includes trusted-write behaviour. It allows some environments to trust that the GlusterFS replication will allow recovering data when a replica goes down. Off by default.
nfs3.<export>.export-dir	Comma-separate list of directories in the given volume.	By default, all subvolumes of nfs are exported as individual exports. There are cases where a subdirectory or subdirectories in the volume need to be exported separately. This option can also be used in conjunction with nfs3.export-volumes option to restrict exports only to the subdirectories specified through this option. Must be an absolute path.
nfs3.export-volumes	Boolean, eg. on, off	Enable or disable exporting whole volumes, instead if used in conjunction with nfs3.export-dir, can allow setting up only subdirectories as exports. On by default.

Filing Defects

Defects can be filed at the Gluster Bug tracker at:

<http://bugs.gluster.com>

To help us identify NFS Beta related bugs, please use the following values when filing the bugs:

Product: GlusterFS

Component: nfs

Release: nfs-alpha

Product Documentation

Complete GlusterFS product documentation is available at:

http://www.gluster.com/community/documentation/index.php/GlusterFS_User_Guide

Frequently Asked Questions

1. mount command on NFS client fails with “RPC Error: Program not registered”?

Start portmap or rpcbind service on the NFS server.

This error is encountered when the server has not started correctly.

On most Linux distributions this can be done by using the following command:

```
$ /etc/init.d/portmap start
```

On some distributions where portmap has been replaced by rpcbind, the following command is required:

```
$ /etc/init.d/rpcbind start
```

2. NFS server glusterfsd start-up fails with “Address already in use” error in the log file.

Change port numbers in the protocol/server section of either of the volume files.

If the NFS server volume file also contains a protocol/server section, there can be a situation where the port numbers are being re-used between two glusterfsd instances on the same server. This can prevent NFS server from starting up correctly with the following error messages in the log file:

```
[2010-05-26 23:31:05] E [socket.c:207:__socket_server_bind] posix-server:
binding to failed: Address already in use
[2010-05-26 23:31:05] E [socket.c:210:__socket_server_bind] posix-server:
Port is already in use
[2010-05-26 23:31:05] E [server-protocol.c:6294:init] posix-server: failed
to bind/listen on socket
[2010-05-26 23:31:05] E [xlator.c:834:xlator_init_rec] posix-server:
Initialization of volume 'posix-server' failed, review your volfile again
[2010-05-26 23:31:05] E [glusterfsd.c:631:_xlator_graph_init] glusterfs:
initializing translator failed
```

```
[2010-05-26 23:31:05] E [glusterfsd.c:1463:main] glusterfs: translator
initialization failed. Exiting
```

Port numbers in a protocol/server section can be changed by editing or adding the following option to the protocol/server section:

```
option listen-port <NEW-PORT-NUMBER>
```

Another Gluster NFS server is running on the same machine.

In other situations, a similar error can arise in case there is already a Gluster NFS server running on the same machine. This situation can be confirmed from the log file, if the following error lines exist:

```
[2010-05-26 23:40:49] E [rpc-socket.c:126:rpcsvc_socket_listen] rpc-
socket: binding socket failed: Address already in use
[2010-05-26 23:40:49] E [rpc-socket.c:129:rpcsvc_socket_listen] rpc-
socket: Port is already in use
[2010-05-26 23:40:49] E [rpcsvc.c:2636:rpcsvc_stage_program_register] rpc-
service: could not create listening connection
[2010-05-26 23:40:49] E [rpcsvc.c:2675:rpcsvc_program_register] rpc-
service: stage registration of program failed
[2010-05-26 23:40:49] E [rpcsvc.c:2695:rpcsvc_program_register] rpc-
service: Program registration failed: MOUNT3, Num: 100005, Ver: 3, Port:
38465
[2010-05-26 23:40:49] E [nfs.c:125:nfs_init_versions] nfs: Program init
failed
[2010-05-26 23:40:49] C [nfs.c:531:notify] nfs: Failed to initialize
protocols
```

To resolve this error one of the Gluster NFS servers will have to be shutdown. At this time, Gluster NFS server does not support running multiple NFS servers on the same machine.

3. mount command fails with the following error message:
mount.nfs: rpc.statd is not running but is required for remote locking.
mount.nfs: Either use '-o nolock' to keep locks local, or start statd.

For NFS clients to mount the NFS server the rpc.statd service must be running on the clients. Please start rpc.statd service by running the following command:

```
$ rpc.statd
```

4. mount command fails with a RPC timeout error or takes too long finish.

Start rpcbind service on the NFS server

Apart from various networking related issues, generally the timeout occurs in two most common situations. First, where the rpcbind service is not running on the NFS server. The resolution for this is given in FAQ 1.

Disable name resolution in NFS server configuration

Another situation where timeouts can occur is when a DNS server used by the NFS server is

not reachable or is taking too long to respond to DNS requests from the NFS server. The correct resolution is to check the connections between the NFS server and the DNS server. A workaround provided in NFS server is to disable DNS requests. NFS server sends DNS requests to authenticate NFS clients based on hostnames. To disable name-lookups, use the following option in the NFS volume section:

```
option rpc-auth.addr.name-lookup off
```

By turning off name lookup support, this authentication will fall back on using IP addresses of the NFS clients. In case an authentication rule in the NFS volume section uses a hostname, this authentication will always fail when name-lookup is disabled.

5. NFS server `glusterfsd` starts but initialization fails with “`rpc-service: portmap registration of program failed`” error message in the log.

NFS start-up can succeed but the initialization of the NFS service can still fail preventing clients from accessing the mount points. Such a situation can be confirmed from the following error messages in the log file:

```
[2010-05-26 23:33:47] E [rpcsvc.c:2598:rpcsvc_program_register_portmap]
rpc-service: Could not register with portmap
[2010-05-26 23:33:47] E [rpcsvc.c:2682:rpcsvc_program_register] rpc-
service: portmap registration of program failed
[2010-05-26 23:33:47] E [rpcsvc.c:2695:rpcsvc_program_register] rpc-
service: Program registration failed: MOUNT3, Num: 100005, Ver: 3, Port:
38465
[2010-05-26 23:33:47] E [nfs.c:125:nfs_init_versions] nfs: Program init
failed
[2010-05-26 23:33:47] C [nfs.c:531:notify] nfs: Failed to initialize
protocols
^C[2010-05-26 23:33:49] E
[rpcsvc.c:2614:rpcsvc_program_unregister_portmap] rpc-service: Could not
unregister with portmap
[2010-05-26 23:33:49] E [rpcsvc.c:2731:rpcsvc_program_unregister] rpc-
service: portmap unregistration of program failed
[2010-05-26 23:33:49] E [rpcsvc.c:2744:rpcsvc_program_unregister] rpc-
service: Program unregistration failed: MOUNT3, Num: 100005, Ver: 3, Port:
38465
```

Start portmap or rpcbind service on the NFS server

Please see FAQ #1 for instructions on starting portmap and rpcbind on the NFS server.

Stop another NFS server running on the same machine.

Such an error is also seen when there is another NFS server running on the same machine but it is not the Gluster NFS server. On Linux systems, this could be the kernel NFS server. Resolution involves stopping the other NFS server or not running the Gluster NFS server on the machine. Before stopping the kernel NFS server please ensure that no critical service depends on access to that NFS server's exports.

On Linux, kernel NFS servers can be stopped by using either of the following commands depending on the distribution in use:

```
/etc/init.d/nfs-kernel-server stop
```

or

```
/etc/init.d/nfs stop
```

6. Operations on the NFS mount point fail with “Access denied” or “Operation not permitted” errors.

Add the access-control translator over every storage/posix volume section.

NFS server requires a slightly different volume file configuration in order to support POSIX conformance. This support is implemented in a new translator called `access-control` which needs to be added over every `storage/posix` volume section that will be exported through NFS. For eg, if your volume files contain a `storage/posix` volume section as below:

```
volume posix
  type storage/posix
  option directory /exports/
end-volume
```

It should be changed to have an access-control translator as shown below:

```
volume posix
  type storage/posix
  option directory /exports/
end-volume
```

```
volume posix-ac
  type features/access-control
  subvolumes posix
end-volume
```

Copyright/Trademarks

Gluster, the Gluster logo and GlusterFS are all trademarks of Gluster, Inc. All other trademarks, registered trademarks, and product names may be trademarks of their respective owners. Additional legal information is at: <http://www.gluster.com/company/legal.php>