Network Services:

Network File System

This material is based on work supported by the National Science Foundation under Grant No. 0802551

National Science Foundation
WHERE DISCOVERIES BEGIN

Lesson Overview

Network File System (NFS) is a protocol that defines how one computer can access the files of a remote computer as if the files were stored locally.

Network File System is a popular service that is often run in a corporate environment and allows remote users to access centrally located files and folders.

Technicians and administrators entering the real world should have a basic understanding of the NSF protocol and services and should be able to configure and manage a Network File System. This lesson serves as a basic introduction to the Network File System.

Select **PLAY** below for a video on the lesson overview:

View Video VideoLesson9Overview (C4L9S2).mp4

Objective

You should know what will be expected of you when you complete this lesson. These expectations are presented as objectives. Objectives are short statements of expectations that tell you what you must be able to do, perform, learn, or adjust after reviewing the lesson.

Lesson Objective:

Given the need to mount shares, the student will be able to configure and mount NFS shares as per industry standards.

Lesson Outline

In this lesson, you will explore:

- Glossary & Resources
- NFS & Versions
- NFS Design
- NFS & Samba
- Installing and Configuring NFS
- Troubleshooting NFS

Resources and Notes

This lesson uses Ubuntu Linux for demonstration. To complete this lesson successfully, you must have access to:

- Ubuntu Linux on bare metal or as a virtual install
- 10 Gb of hard drive space dedicated to the operating system's use
- A command shell
- Internet for research
- Word processor

Use the resources on the right to configure your system for Ubuntu.

Resources:

- Download Virtualbox
- Using Virtualbox with Ubuntu
- Virtualbox for Linux Hosts
- Virtualbox manual

Glossary of Terms

View Video
VideoLesson9IntroToClonezilla
(C4L9S6).mp4

- **Bundling** - The process in which the client holds several changes to a file in local cache and then transfers the data back to the server in one group (bundle)
- **Bottlenecking** - The slowing down of data transfer on a network due to physical restrictions, hardware failure, or improper configuration
- **Centralized Server configuration** - A network configuration where data is stored (normally) on a single computer and clients access that information from remote locations. Advantages include reduced administrative costs and simplified management of large amounts of data since the data only needs to be managed on one machine. Disadvantages include security from unauthorized users and single point of failure (if the server fails authorized users cannot access data to do their work)
- **Clonezilla** - An open source Live CD that can backup an existing hard drive. Similar to Norton Ghost. You can learn more about Clonezilla on this video.
- **Directory Service** - A process that allows user friendly names to be associated to files and folders so that humans can read and associate with the folders and files easily
- **File Service** - The specifications of what a client computer is provided by the file system
- **File Server** - The process in which one or more computers provide file services to other computers whether remotely or locally

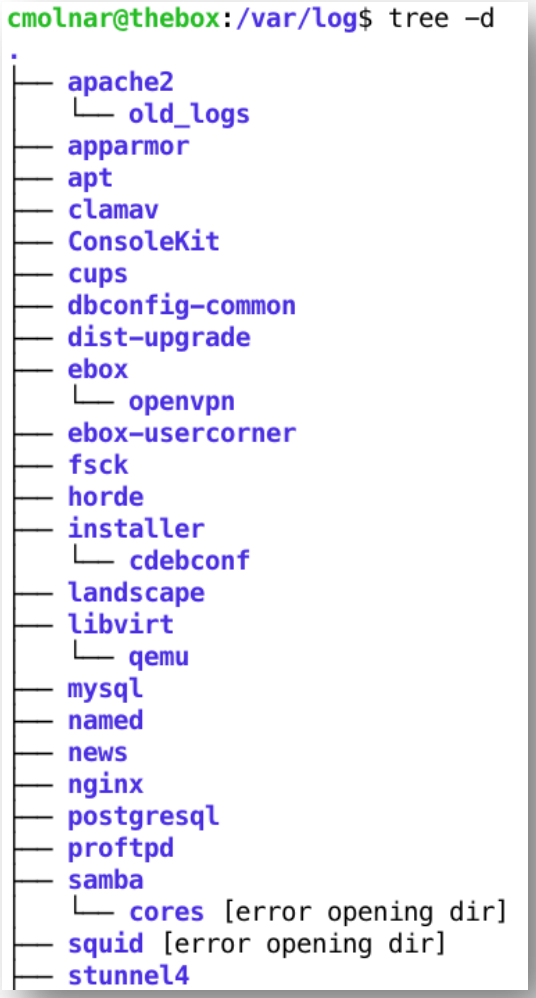
Glossary of Terms

- **Folder mapping** - A process in which a folder or its contents are made accessible over a network in a manner similar to that of the local directory
- **RPC** - **Remote Procedural Call** - The protocol used when one computer makes a request to another computer running a particular service. RPC allows the requestor to access the service without knowing the details of the network.
- **Stateful System** - Connection requests from the client to the server are stored on the server. This process can tie up server resources such as memory and CPU time but may lead to improved process efficiency since information is immediately available and does not need to be re-accessed.
- **Stateless System** - A stateless system is a process in which the client sends a request to the server and the server processes the request and returns some sort of result. In doing so, the server does not store any client specific information on the process (improves overall server efficiency by releasing resources).
- **TCP** - A reliable, connection based transport mechanism (protocol) that relies on segmentation and handshaking to ensure packets are delivered in a network environment. The segmentation and hand shaking increase reliability and data delivery.
- **UDP** - A connectionless transport mechanism (protocol) that is primarily used when quick communications is preferred. Items such as short query based applications (DNS) or media streaming are examples.

Helpful Links

Numerous websites provide helpful information on NFS that you should review to learn more about NFS. Some of the better options include:

- Linux NFS HowTo
- Chapter on NFS
- NFS and Linux
- Debugging NFS
- Troubleshooting NFS



Network File System (NFS)

The Network File System (NSF) is an RPC based, network service originally created by Sun in the 1980's that allowed remote computers transparent access to centralized folders and files on a server. By transparent access, we are referring to a process in which the local host recognizes the remote files as if they were localized.

NFS allowed the remote users to easily switch worker stations and still be granted access to the required folders and files to complete tasks and projects. Recognized as the most popular distributed file system used on modern Unix systems today, NFS can also be implemented on both Windows and Apple platforms that are best utilized using third party tools. (Usage on Windows and Mac systems is not common in most environments although it can be done. YouTube has several examples on setting these up.)

Common uses of NFS include storing a user's home directory on a centralized server so that it can be accessed remotely as if it were a local directory, storing information bulletins in a central location and mapping end user's to the information, rolling out software / development tools, and mail management / queuing.

Select **PLAY** below for a video on NFS.

View Video VideoLesson9WhatIsNfs (C4L9S9).mp4

NFS Versions

Version 1 was used internally by the Sun development team and there is little to no information easily available on this version (probably because it was a prototype).

Version 2 was the first publicly available version and primarily used UDP as the transport mechanism. As a 32 bit implementation of the protocol, there was a maximum file size limitation of 2GB and transmitted data size was limited to 8KB.

Ten years from the roll out of Version 2, Version 3 took advantages of several improvements in technology. TCP took over as the transmission mechanism (as long as both the server and client support it) which improved NFS performance.

Maximum data transfer size was increased from 8KB to 64KB (full 64-bit implementation) so file size limitations were eliminated. Bundling, *the process in which the client updates local cache initially when changes are made to a file then updates the server in "bundles,"* was introduced to improve network performance and reduce bottlenecking. File attributes are now automatically returned with each file request from the client.

Version 4 is primarily designed for Internet use and adds increased persistence, security and improved performance. However, administrative tasks to maintain this level of operation are more challenging and differ amongst the various Unix/Linux flavors.

NFS Design

The file system is responsible for the management of file access and organization. Some examples of these responsibilities include creating, naming, protecting and providing access to files and folders. The Network File System is capable of providing these basic services and many more. In reality, NFS can do just about anything our local file services can do but through a remote network connection instead of direct local disk access.

The main consideration when designing a Network File System is security. For the most part, the NFS service should only be implemented on a local area network and extreme caution should be considered when applying the service in a wide area network (Internet). However, the newer NFS version 4 does have some additional security features that are not offered in previous versions, but those features also add some additional steps in setup and can be complicated in administering. Therefore, NTFS version 3 is preferred by most administrators.

Another consideration when designing and implementing a Network File System is the potential for bottlenecks. As network traffic increases, data bottlenecks become a reality and may hamper network communications. The NFS protocol itself offers some minimization of this by providing efficient cache processes on the client machine. Designers should still pay close attention to network traffic during busy times and ensure proper contingency plans are in place to prevent bottlenecking.

NFS Design (Contd)

As with a local file system, proper folder and file names should be used in the directory service to minimize network accesses, searches, and traffic. Whenever possible, consideration should be given when implementing an NFS where the files will be changed on a frequent basis.

Frequent changes in files and folders will add additional traffic to the network and could lead to network bottlenecks and restricted data throughput. The ideal implementation of a Network File System is for files that are immutable (never changing).

NFS uses port 2049 for communications and NSF version 2 uses Universal Datagram Protocol (UDP) while NFS version 3 uses Transmission Control Protocol. The general thought is that UDP minimized transmission delays since no flow control is used, but TCP allows for a more reliable delivery of data. After much debate, Version 3, featuring TCP, is still used today.

NFS and Samba

The Network File System was designed and implemented to allow Unix based systems the ability to map external drives so that the users can access the remote drives in a manner similar to the way they access their local file system. NFS can be implemented on systems that do contain Windows-based hosts but third party tools and extra administration is required for proper operation.

Samba was designed to allow Unix-based computers the ability to share information with (primarily) Windows-based hosts. Samba is considered to be a little more difficult to administer, especially in later versions of Ubuntu, but GUI based tools are available in most package managers.

Both NFS and Samba are a combination of services and protocols designed for practicality and efficiency. Both systems are easy to implement and administer initially but get more complex as additional security measures are added. One distinct difference in NSF as compared to Samba is that NSF is built directly into the Kernel in NSF Version 3 and above.

Select **PLAY** below for a video on the differences between NFS and Samba.

View Video VideoLesson9nfsVsSamba(C4L9S13).mp4

How Does NSF Work?

As previously stated, the Network File System is a series of protocols and services that allow remote machines to access files and folders on a centralized server and display the contents of the server on the local machine in a manner similar to the local file system. Simply stated, the Network File System allows us to map network drives as if they were local drives.

The Network File System operates in a manner that is referred to as "stateless." That is, when a client makes a query to a remote server, the server carries out that request and returns the value or data to the client. Once the data is sent to the client, the server "forgets" about the client request and retains no status or history of the request. This stateless operation adds some advantages to our server performance which is listed below:

- If the server crashes and then restores itself, the client connection does not need to be restored since there was no state to maintain.
- Since there is no client server state maintained, additional resources are not required to keep the connection on the server.
- There is no limit on the numbers of files that can be open at any one time on the server.
- If the client crashes there is no side effect on the server.

Recommended Reading

- Linux NFS Howto

How Does NSF Work? (Contd)

If the client makes a request to an NFS server and the server crashes during the request, the client simply keeps sending the request until the server is back online and "answers" or until the client's retry attempts exceed the maximum number of attempts allowed.

Another key benefit of the NFS process is the use of client cache. The information requested by the client is transferred to client cache from the server disk and then stored on the client disk. If and when changes are made to the file, the changes are first applied to the cache and then stored to the client disk. Client caching was designed to reduce the number of remote calls on the network.

There are downsides to a stateless system, which include it being a very noisy process, inefficient and slow. Data that is saved from the client to the server must be written synchronously to the server's disk, and if the client crashes before the synchronization takes place, data on the client is lost since no state is maintained by the server.

Recommended Reading

- NFS Prereq

Install NFS

The Network File System is supported by most common package managers, and these packages managers should be used if possible.

Package managers are normally extremely reliable in managing any additional dependencies that may be required by the original application the administrator intends to install.

Searching for and locating the NFS application through most package managers is easy.

1. Select **Systems**>**Administration**>**Synaptic Package Manager**
2. Enter **nfs** in the search string selection box

Select **Search** and after a few seconds, look for following two packages:

- *nfs-common*
- *nfs-kernel-server*

Select these two packages by marking and applying them as with previous package installations.

Select **PLAY** below for a video on installing NFS.

View Video
VideoLesson9InstallNfs
(C4L9S16).mp4

Required Reading

- Ubuntu & NFS

Install NFS (Contd)

As with all Linux distributions, savvy administrators may want to use apt-get from a terminal as root to install the server:

#sudo apt-get install nfs-kernel-server nfs-common portmap

The client application can be installed from a terminal as root by using:

#apt-get install nfs-common

Note:
The Network File System also uses port mapping to map services to the ports on which they are listening. Make sure you have the portmapper service installed and that the firewall is properly configured for the service (port 111 UDP and TCP, NSF requires port 2049 UDP and TCP).

Select **PLAY** below for videos on configuring NFS.

View Video VideoLesson9nfsUbuntu (C4L9S17v1).mp4

Ubuntu

View Video VideoLesson9nfsDebian(C4L9s17v2)mp4

Debian

View Video VideoLesson9NfsServer(C4L9S17v3)mp4

NFS server

Using NFS

NFS operation has recently been simplified with the addition of GUI driven packages that allow for easy configuration. NFS can also be managed by simply changing the /etc/exports file on your Linux based system.

Do not forget to restart the NFS server every time you change the /etc/exports file. The easiest way to reread the configuration is to call:

 # /etc/init.d/nfs-kernel-server start

Note: The /etc/exports file is very space sensitive. Make sure that only one space character is used between text groupings. If your exports are not working correctly, this would be an item to check initially!

Setting up an export file is simple… .you just need to specify three attributes in the file.

1. The directory to be shared (exported)
2. The computer, NIS group, hostname, domain name, or subnet allowed to access that directory
3. Any options, such as ro or rw, or several other options

Required Reading

- /etc/exports manpage

Sample /etc/exports File

```
# sample /etc/exports file
      /           master(rw) trusty(rw,no_root_squash)
      /projects      proj*.local.domain(rw)
      /usr         *.local.domain(ro) @trusted(rw)
      /home/joe      pc001(rw,all_squash,anonuid=150,anongid=100)
      /pub         (ro,insecure,all_squash)
      /pub/private   (noaccess)
```

The first line exports the entire filesystem to the *master* and *musty* machines. In addition to write access, all UID squashing is turned off for the *trusty* host. The second and third entries show examples for wildcard hostnames and netgroups (this is the entry '@trusted').

The fourth line shows the entry for the PC/NFS client discussed above. Line 5 exports the public FTP directory to every host in the world, executing all requests under the nobody account. The insecure option in this entry also allows clients with NFS implementations that do not use a reserved port for NFS. The last line denies all NFS clients access to the private directory.

Note: The server determines the type of access allowed to the server's files. The /etc/exports file lists the file systems the server will make available for clients to mount and use. By default, the *mountd* will disallow access to all server directories and the /etc/exports file will give permissions for the remote client to access the service on those folders.

Required Reading

- /etc/exports example
- /etc/exports config

Troubleshooting NFS

Errors	Description
Too many levels of remote in path	Attempting to mount a file system that has already been mounted
Permission denied	User is denied access. The user on the client may not exist on the server or an improperly configured /etc/exports file
No such host	Typographical or DNS configuration error in the name of the server
No such file or directory	Typographical error in the name of the file or directory: they don't exist
NFS server is not responding	The server could be overloaded or down. Network could be inaccessible. Local DNS could be damaged
Stale file handle	A file that was previously accessed by the client was deleted on the server before the client closed it.
Fake hostname	Forward and reverse DNS entries don't exist for the NFS client

Troubleshooting NFS

Troubleshooting tips taken and summarized from : NFS Gotchas

Cause category	Symptom
The portmap or nfs daemons are not running	Typically, failure to mount
Syntax error on client mount command or server's /etc/exports	Typically, failure to mount or failure to write enable. A space between the mount point and the (rw) causes the share to be read-only -- a frustrating and hard to diagnose problem. (Whitespace is tough to troubleshoot.)
Problems with permissions, uid's and gid's	Mounts OK, but access to the data is impossible or not as specified. Possible directory error
Firewalls filtering packets necessary for NFS	Mount failures, timeouts, excessively slow mounts, or intermittent mounts
Bad DNS on server	Mount failures, timeouts, excessively slow mounts, or intermittent mounts

Troubleshooting Tips

1. CHECK THE DAEMONS ON THE SERVER

Perform the following two commands on the server:
ps ax | grep portmap
ps ax | grep nfs

If either shows nothing (or if it shows just the *grep* command), that server is not running. Investigate why. Start by seeing if the server is even set to run at boot:

/sbin/chkconfig --list portmap
/sbin/chkconfig --list nfs

Each command will output a line showing the run levels at which the command is on. If either one is not on at any runlevel between 3 and 5 inclusive, turn it on with one or both of these commands:

/sbin/chkconfig portmap on
/sbin/chkconfig nfs on

The preceding commands set it to fire at boot but do not run the daemon. You must run them manually:
service portmap restart
service nfs restart

Continued on next . . .

Troubleshooting Tips

Continued from previous . . .

Always restart the portmap daemon before restarting the NFS daemon, because NFS needs the portmapper to function. If either of those commands fails or produces an error message, investigate.

IMPORTANT NOTE:

Even if the daemons were both running when you investigated, restart them both anyway. First, you might see an error message. Second, it's always nice to achieve a known state. Restarting these two daemons should take about a minute. That one minute is a small price to pay for the peace of mind you achieve knowing there are no undiscovered problem with the daemons.

If NFS fails to start, investigate the syntax in /etc/exports and possibly comment out everything in that file, and try another restart. If that changes the symptom, divide and conquer. If restarting NFS takes a long time, investigate the server's DNS.

Troubleshooting Tips

2: CHECK AND DOUBLE CHECK THE SYNTAX

If the daemons work, review the syntax of the mount command on the client and the /etc/exports file on the server. Obviously, if you use the wrong syntax (or wrong IP addresses or directories) in your mount command, the mount will fail. You do not need to take a great deal of time—just verify that the syntax is correct and that the IP address is correct. Do the same for directories and mount points. Correct as necessary, and retest.

3: CAREFULLY READ ERROR MESSAGES AND DEVELOP A SYMPTOM DESCRIPTION

The first two steps were general maintenance—educated guesses designed to yield quick solutions. If they didn't work, it's time to troubleshoot. The first step is to read the error message and learn more about it. You might want to check the system logs (start with /var/log/messages) in case relevant messages were written.

Try several mounts and unmounts and note the exact malfunction:

- Does the mount produce an error message?
- Does the mount time out?
- Does the mount appear to hang forever (more than 5 minutes)?
- Does the mount appear to succeed, but the data can't be seen, read, or written as expected?
- Does the symptom change over time, or with reboots?

The more you learn and record about the symptom, the better your chances of solving the problem.

Troubleshooting Tips

4: THE DAEMON MOUNTS BUT YOU CANNOT ACCESS FILES, CHECK PERMISSIONS, OR VIEW THE GIDS AND UIDS

Generally speaking, the permissions on the server do not affect the mounting or unmounting of the NFS share. But they affect whether such a share can be seen, executed, read, or written. Often the cause is obvious. If the directory is owned by root or permissioned as 700, it obviously cannot be read and written by the user. This type of problem is easy to diagnose and fix.

By far, the toughest problems are caused by non-matching UIDs and GIDs. Let's say you share your home directory on the server, and you log in as yourself on the client and mount that share. It will mount without problems assuming you used **su -c** or **sudo** to mount it, but you will not be able to read the data because permission will be denied!

5: IF THERE ARE STILL PROBLEMS, DISABLE FIREWALLS OR LOG FIREWALLS

Many supposed NFS problems are really problems with the firewall. In order for your NFS server to successfully serve NFS shares, its firewall must enable the following:

- ICMP Type 3 packets
- Port 111, the portmap daemon
- Port 2049, NFS
- The port(s) assigned to the mountd daemon

Troubleshooting Tips

6: If there are still problems, investigate the server's DNS, host name resolution, etc

Bad forward and reverse name resolution can mess up any server app, including NFS. Like other apps, bad DNS most often results in very slow performance or timeouts.

Be sure to check your /etc/resolv.conf and make sure you're querying the correct DNS server. Check your DNS server with DNS walk or DNS lint or another suitable utility.

Recommended Review

- NFS Gotchas

Lesson Summary

The Network File System is one of the most popular Linux-based services in use today. NFS offers local users access to remote files and folders by implementing simple and proven protocols via a network infrastructure. This centralized environment can reduce administrative responsibilities, limit resources, and improve overall operational efficiency.

This lesson served as in introduction to the Network File Service and provided an overview for setting up and accessing a Network File System. Students also discussed potential real world scenarios through the case studies presented in the Forum discussions. A bonus lab was created to demonstrate how NFS is used from the popular Linux cloning tool called Clonezilla.

Given the overall popularity of the Network File System and associated services, students must be familiar with NFS concepts prior to entering the real world environment.

Select **PLAY** below for a summary video on NFS.

View Video VideoLesson9NfsSummary (C4L9S27)mp4