Lecture 8

**Server Client-Model-NFS-Server**

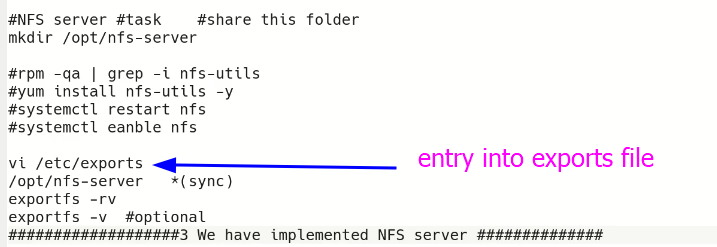
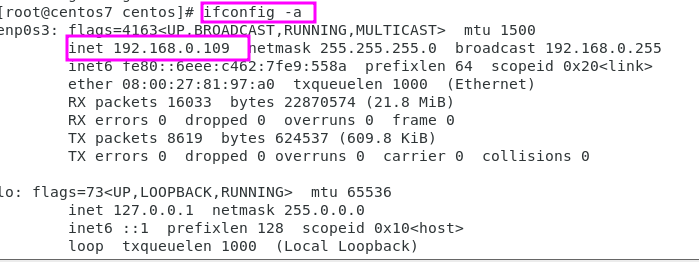
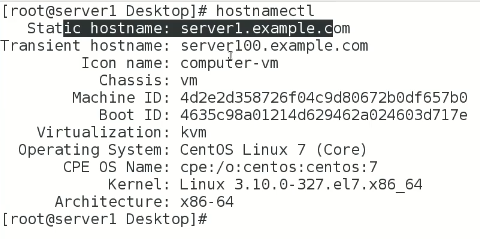
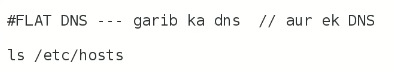
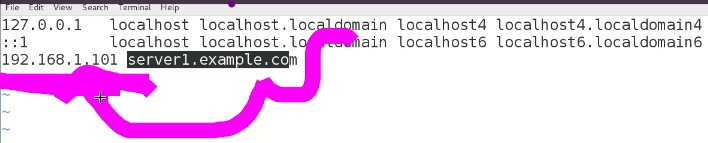
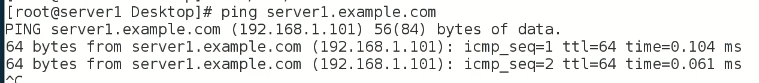
NFS 🡪 Network File Sharing or Network File Service

* In AWS its alternative is EFS
* Working of NFS is same as NAS (Network Attached Storage)
* Network-Attached Storage: This is a type of data storage that is connected to a computer network and provides file-level access to multiple clients. NAS devices typically use file-based protocols such as NFS or SMB/CIFS and can be accessed by multiple users at the same time.
* Port 2049
* Works on UDP Protocol
* Linux to Linux sharing
* Doesn’t ask to username & password (drawback)

Samba 🡪 file service

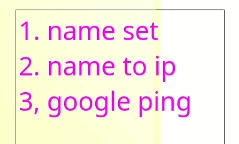
* + Port No. 137, 139, 335
  + Uses TCP Protocol
  + Linux to Linux or Linux to Windows
  + Requires Username & Password
  + Server CentOS 7 VM
  + Client CentOS 6 VM

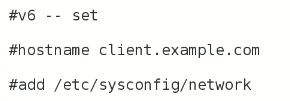
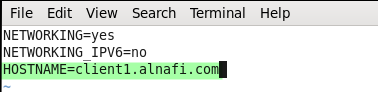
Server Configuration

* Tasks 🡪 share a folder over network,
  + Steps
* 
* 1 to 2 minutes time is required in Linux. 🡪 but in production environment ask for 1 to 2 days for complete setup and trouble shooting.
* Server 🡪 services (services run in server) 🡪 package is needed to be installed in server 🡪 then service is needed to be started.
* We need an IP address to create a server.
  + Private IP in this case
  + $ ifconfig -a
  + 
* How to set name for a machine /Server
* $ hostnamectl set-hostname server1.example.com - -static (in CentOS 7 & 8) (this step is not compulsory but it’s a standard way)
* $ cat /etc/hostname 🡪 to check
* 
* Point to be noted is that prompt is not yet changed as per new name because it is already loaded
* To check open a new terminal window 🡪 use $ su – to refresh active terminal.
* $ hostnamectl 🡪 for complete details of hostname
* 
* $ ping server1.example.com 🡪 ping is not yet working.
* We need to do DNS entry 🡪 DNS name to IP address resolution
* #BIND package in DNS topic
* For now,
* 
* Every application looks for IP in /etc/hosts file
* **$ vi /etc/hosts** 🡪 for entry of the IP
* 
* This is called IP mapping with server or domain name etc.
* Ping is now working
* 
* Also verify if **$ ping google.com** is working

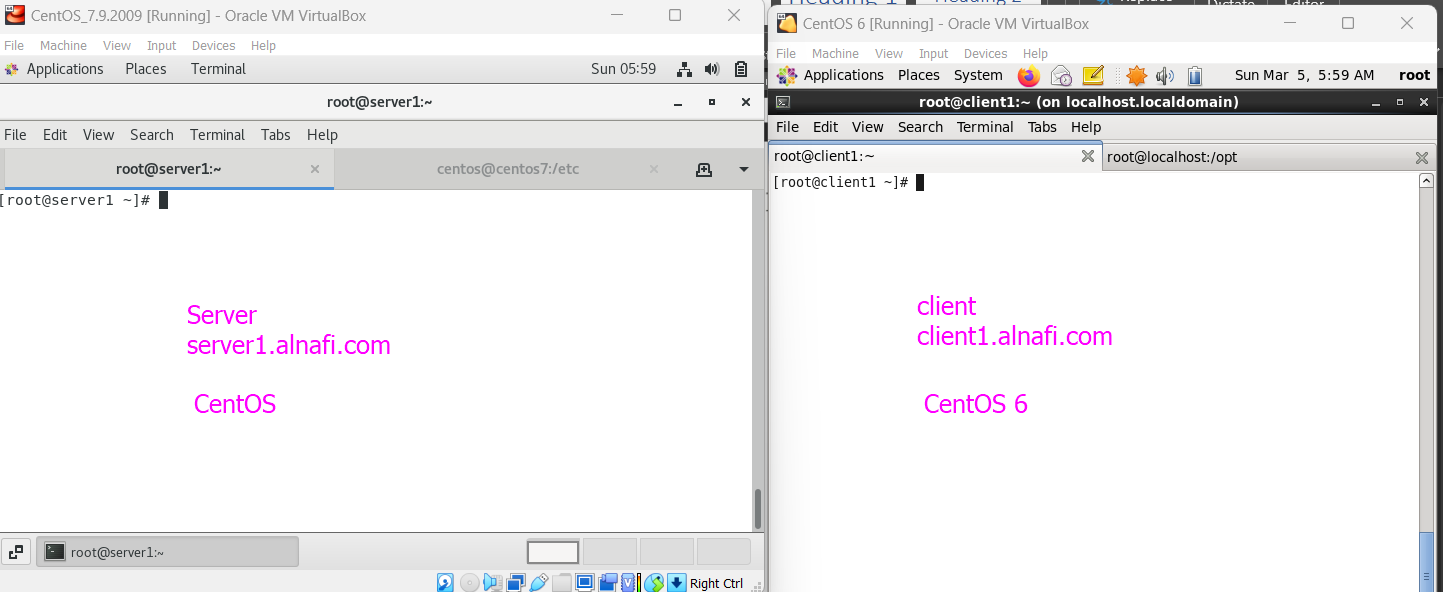
Client-side configuration (CentOS 6 VM)

Steps

`- 

* 
* 🡪 make it persistent by entry into /etc/sysconfig/network
* 
* -

Machines are ready

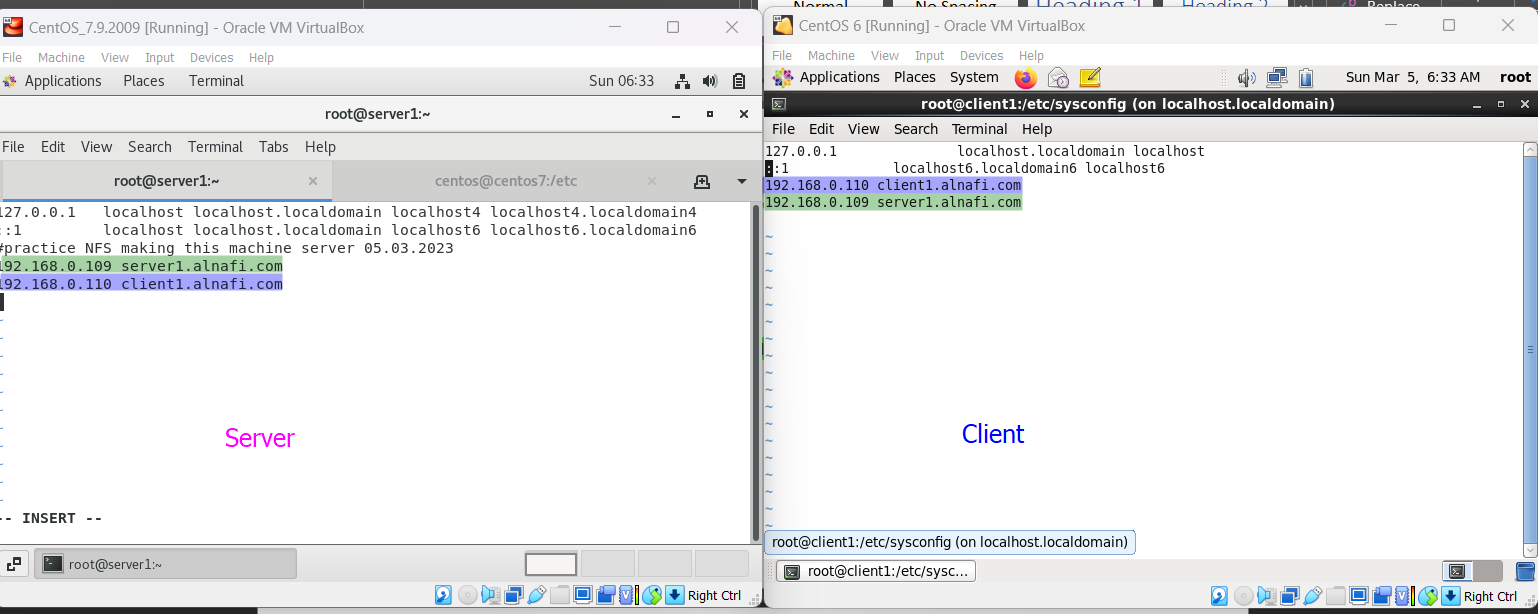


Right now,

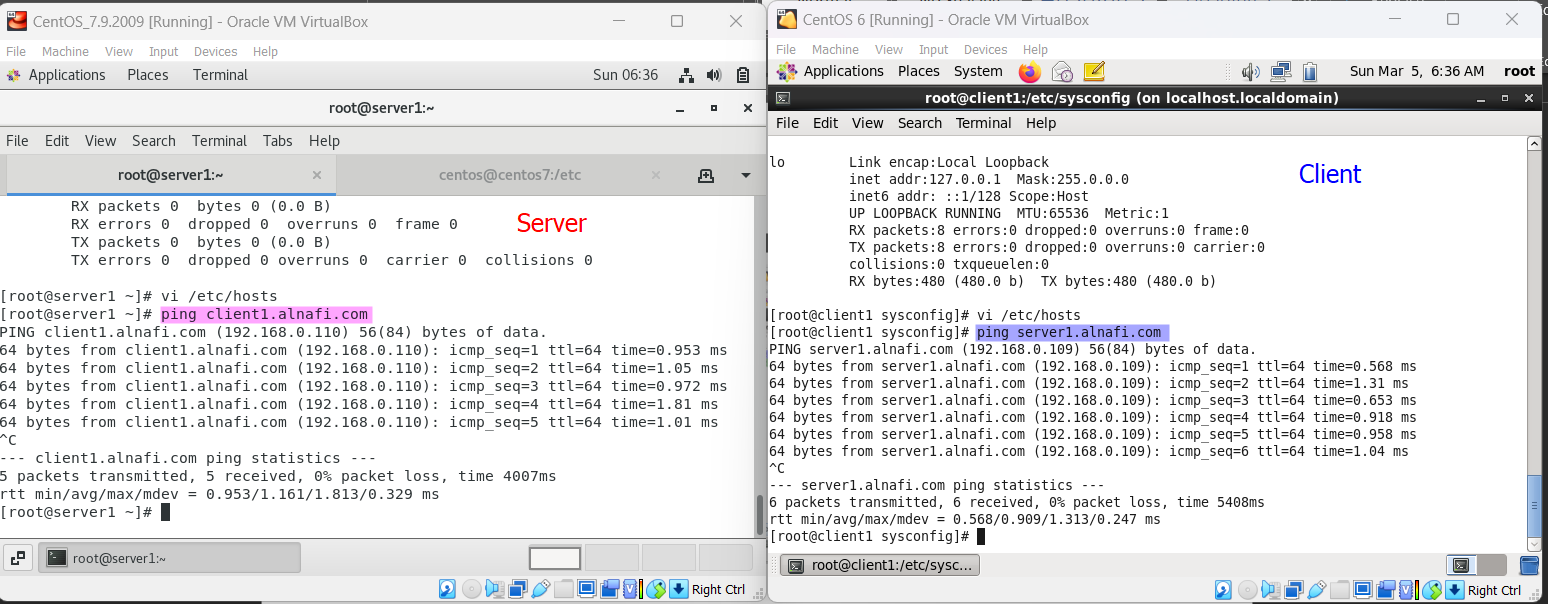
Both machines can’t ping each other

For this both machines /etc/hosts needed to be updated

* **Server should have Client entry.**
* **Client should have server entry.**

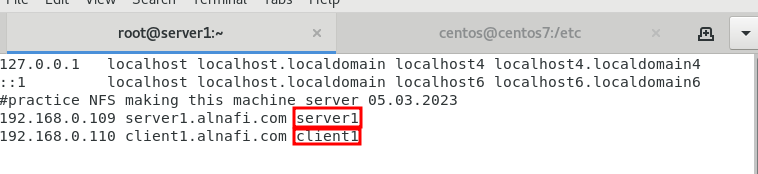


* Ping is working on both machines.



-

If we just want to use short name instead of full **server1 or client1 .alnafi.com**

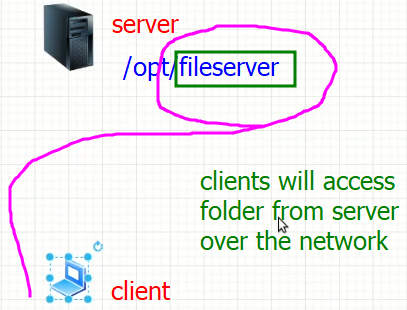


* **$ ping server1 or client1**
* Now all other servers can be initiated like NFS, apache, samba etc.

**On server-side**

$ mkdir **/opt/fileserver 🡪 make a directory in /opt named as “fileserver”.**

We r going to share this **“fileserver”** directory/folder on network.



* $ yum install nfs-utils 🡪 if NFS is not installed
* **$ systemctl restart rpcbind** 🡪 it is dependency of NFS
* **rpcbind** is a service that maps Remote Procedure Call (RPC) program numbers to network addresses. It is commonly used in Unix-based systems to allow different programs to communicate with each other over a network.
* When a program needs to communicate with another program over RPC, it needs to know the port number on which the RPC service is listening. This is where rpcbind comes in - it acts as a kind of directory service that provides the necessary information to the requesting program.
* Rpcbind maintains a database of registered RPC programs and their corresponding network addresses. When a program requests a particular RPC service, rpcbind consults its database to find the port number associated with that service and returns it to the requesting program. This allows the requesting program to establish a connection with the RPC service and exchange data over the network.
* In summary, rpcbind is a crucial service that facilitates communication between different programs over a network by mapping RPC program numbers to network addresses.
* **$ systemctl restart nfs**
* .
* 

**$ system “enable”** **rpcbind** & **nfs** means to auto-start the specific service while booting. And to avoid above “restart” command again and again

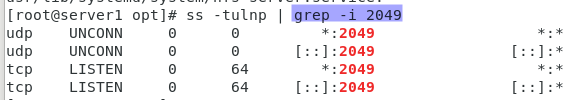
* As the command is executed port **No. 2049** opens
* To check
* **$ netstat -tulnp | grep 2049**

"netstat tulnp" is a command used in Linux and other Unix-based operating systems to display information about active network connections and related processes.

Here is a breakdown of each option in the command:

* "netstat": This command is used to display network connections.
* "-t": This option specifies that we want to display TCP connections.
* "-u": This option specifies that we want to display UDP connections.
* "-l": This option specifies that we only want to display listening sockets.
* "-n": This option specifies that we want to display numerical IP addresses instead of domain names.
* "-p": This option specifies that we want to display the process ID and name of the program that is using each connection.

So, the full command "netstat tulnp" will display a list of all TCP and UDP connections that are currently listening on the system, along with their numerical IP addresses, and the process ID and name of the program that is using each connection. This can be useful for troubleshooting network connectivity issues or identifying which programs are using network resources.

* **0r**
* **$ ss -tulnp | grep 2049**
* 

"ss -tulnp" is a command used in Unix-based operating systems to display information about active network connections and related processes.

Here is a breakdown of each option in the command:

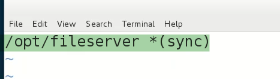
* "ss": This command is used to display **socket statistics.**
* "-t": This option specifies that we want to display TCP connections.
* "-u": This option specifies that we want to display UDP connections.
* "-l": This option specifies that we only want to display listening sockets.
* "-n": This option specifies that we want to display numerical IP addresses instead of domain names.
* "-p": This option specifies that we want to display the process ID and name of the program that is using each connection.

So, the full command "ss -tulnp" will display a list of all TCP and UDP connections that are currently listening on the system, along with their numerical IP addresses, and the process ID and name of the program that is using each connection. This can be useful for troubleshooting network connectivity issues or identifying which programs are using network resources.

Compared to "netstat", "ss" is a newer and more efficient tool for displaying socket statistics. It provides more detailed information and is capable of displaying a wider range of socket types, making it a more powerful alternative to "netstat".

* Now we need to share “fileserver” in opt/fileserver.
* NFS has a major file “/etc**/exports**”
* The "**/etc/exports**" file is used in Unix-based operating systems to configure the directories that can be shared with remote systems using the NFS (Network File System) protocol.
* This file contains a list of directory paths and the options that define how those directories are shared with remote systems. Each entry in the file specifies a directory to be shared, the IP address or hostname of the systems that are allowed to access that directory, and the permissions that are granted to those systems.
* Here's an example of what an entry in the "/etc/exports" file might look like:

**/path/to/shared/directory 192.168.1.100(rw,sync) 192.168.1.101(ro)**

* n this example, the directory "/path/to/shared/directory" is being shared with two remote systems. The IP address "192.168.1.100" is allowed to read and write to the directory (rw), while the IP address "192.168.1.101" is only allowed to read from the directory (ro). The "sync" option specifies that data should be written to the disk before it is considered to be written, ensuring data consistency between the client and the server.
* Once changes have been made to the "/etc/exports" file, the "exportfs" command can be used to apply those changes and make the shared directories available to remote systems over NFS.
* In our case
* 
* $ ls -d /opt/fileserver
* The "**ls -d**" command is used in Unix-based operating systems to list information about a directory without showing the contents of that directory. The "-d" option instructs the "ls" command to only list information about the directory itself, rather than listing the contents of the directory.
* 
* It will load the exports file into the RAM

The **"exportfs -rv"** command is used in Unix-based operating systems to update the list of directories that are shared with remote systems using the NFS (Network File System) protocol.

Here is a breakdown of each option in the command:

* **"exportfs"**: This command is used to manage the list of directories that are shared using NFS.
* "-r": This option instructs the "exportfs" command to re-export all directories that are currently listed in the "/etc/exports" file.
* "-v": This option instructs the "exportfs" command to display verbose output, including any errors or warnings that may occur during the export process.

So, the full command "exportfs -rv" will update the list of shared directories based on the configuration in the "/etc/exports" file and display any errors or warnings that occur during the export process. This command is typically used after making changes to the "/etc/exports" file to apply those changes and make the shared directories available to remote systems over NFS.

Note that the "exportfs" command requires root privileges to run, so you may need to use the "sudo" command or log in as the root user in order to run this command. Additionally, it's important to ensure that the NFS service is running on the system before attempting to share directories using NFS.

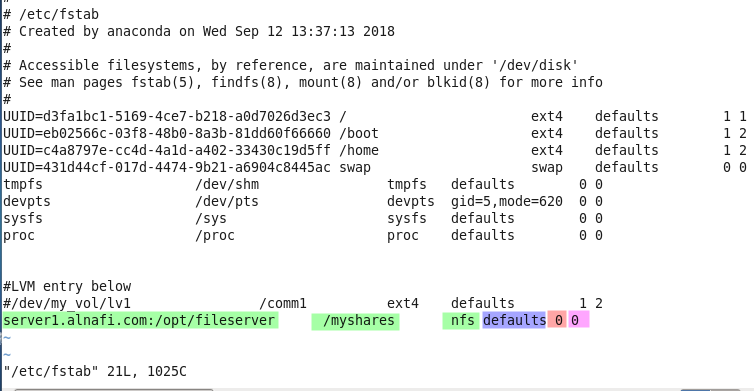
Top of Form

After these steps on server1 the NFS Server is established

* Disable firewall on server-side
* $ systemctl stop firewalld
* To resolve this error
  + “clnt\_create: RPC: Port mapper failure - Unable to receive: errno 113 (No route to host)”

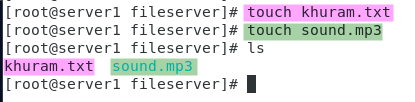
**Clint-side**

* Check ping “server1”.
* $ ping server1
* $ showmount -e server1.alnafi.com
* 
* After firewall is stopped on server1
* 
* This error is due to firewall running on server-side
* $ mkdir /myshares
* $ mount server1.alnafi.com:/opt/fileserver /myshares
* $ df -h 🡪 to show the shared directory.
* Table

  Description automatically generated
* **To make it persistent add its entry to “/etc/fstab”**
* 

Check if files are being shared

**Create few files in shared folder in server1.**



**Now access them in client1**



**Troubleshooting**

***Sir Kazim told that in future mostly there would be problem in server. Keep it in mind always.***

**After restart my network went down,**

* There was a change in IP after restart because I didn’t set static IP on server, so I set the static IP that was mapped in /etc/hosts file
* I restarted rpcbind and nfs services.
* I disabled firewall on Server.
* For a safe side I also restarted NetworkManager on server
* .
* **On client machine,**
* I checked the ping it was not working (possible because of change in IP at server)
* I restarted the nfs service.
* After troubleshooting at server, client started ping to server.
* I executed mount -a command to mount the shared partition
* After these steps 🡪 network worked perfectly and files were available on shared directory.

**-Server-side troubleshooting**

If NFS is not working on a CentOS 7 server, here are some troubleshooting steps that you can follow:

1. Check if the NFS service is running on the server:

**systemctl status nfs-server**

If the service is not running, start it using:

**systemctl start nfs-server**

1. Check if the exported filesystems are correctly specified in the /etc/exports file:

**cat /etc/exports**

If the exported filesystems are not correctly specified, edit the /etc/exports file to fix them.

1. Check if the NFS ports are open on the server:

**firewall-cmd --list-ports | grep nfs**

If the ports are not open, configure the firewall to allow NFS traffic.

1. Check if the correct permissions are set on the exported filesystems:

**ls -ld /exported/filesystem**

If the permissions are incorrect, change them using chmod and chown commands.

1. Check if the NFS server is exporting the required filesystems:

**showmount -e localhost**

If the required filesystems are not being exported, configure the NFS server to export them.

1. Check if SELinux is blocking NFS traffic:

**getsebool -a | grep nfs**

If SELinux is blocking NFS traffic, configure SELinux to allow it.

1. Check if the kernel module for NFS is loaded:

**lsmod | grep nfs**

If the module is not loaded, load it using:

**modprobe nfs**

Once you have identified the issue and taken the necessary steps to resolve it, restart the NFS service:

**systemctl restart nfs-server**

This should allow the clients to access the exported filesystems on the server. If you are still experiencing issues, you may need to check the system logs for any NFS-related errors:

**ournalctl -u nfs-server**

This should help you diagnose and troubleshoot any remaining issues with NFS on your CentOS 7 server.

**-Client-side troubleshooting**

There are several reasons why NFS service may not be working on a client CentOS 6 system. Here are some troubleshooting steps you can take:

1. Check if NFS service is running on the client system:

**service nfs status**

If the service is not running, start it using:

**service nfs start**

1. Check if the NFS mount points are correctly specified in /etc/fstab:

**cat /etc/fstab | grep nfs**

If the mount points are not correctly specified, edit /etc/fstab to fix them.

1. Check if the NFS server is reachable from the client system:

**ping <nfs\_server\_ip>**

If the NFS server is not reachable, check network connectivity between the client and server.

1. Check if NFS ports are open on the server:

**nmap -p 111 <nfs\_server\_ip> nmap -p 2049 <nfs\_server\_ip>**

If the ports are not open, configure the firewall to allow NFS traffic.

1. Check if the NFS server is exporting the required filesystems:

**showmount -e <nfs\_server\_ip>**

If the required filesystems are not being exported, configure the NFS server to export them.

1. Check if the correct permissions are set on the exported filesystems:

**ls -ld /exported/filesystem**

If the permissions are incorrect, change them using chmod and chown commands.

1. Check if SELinux is blocking NFS traffic:

**getsebool -a | grep nfs**

If SELinux is blocking NFS traffic, configure SELinux to allow it.

Once you have identified the issue and taken the necessary steps to resolve it, restart NFS service on the client system:

**service nfs restart**

and mount the NFS shares again:

**mount -a**

This should allow the client to access the NFS shares on the server.