**SAMBA**

“Samba is an Open Source/Free Software suite that provides seamless file and print services to SMB/CIFS clients." Samba is freely available, unlike other SMB/CIFS implementations, and allows for interoperability between Linux/Unix servers and Windows-based clients.

**Ports :-**

 UDP port 137 for netbiosns, the NETBIOS Name Service  
 UDP port 138 for netbiosdgm, the NETBIOS Datagram Service  
 TCP port 139 for netbiosssn, the NETBIOS session service  
 TCP port 445 for microsoftds, the Microsoft Domain Service

**Configuration file :**   /etc/samba/smb.conf

**SELinux security context used for SAMBA :** Chcon –t samba\_share\_t  <shared directory name>

**SMB PROTOCOL WORKS:**

Step 1 : SMB connection between a client and specific share on a server

Step 2: creates a session between the client and server by using a user name and password.

Step 3 : Makes a successful connection to shared resources and it serves request to authenticated users.

**what is nmbd daemon?**

This daemon handles all name registration and resolution requests. It is the primary vehicle involved in network browsing. It handles all UDP-based protocols. The nmbd daemon should be the first command started as part of the Samba startup process.

**What is smdb daemon?**

This daemon handles all TCP/IP-based connection services for file- and print-based operations. It also manages local authentication. It should be started immediately following the startup of nmbd.

**IPTABLES:**

-A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp --dport 137 -j ACCEPT

-A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp --dport 138 -j ACCEPT

-A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp --dport 139 -j ACCEPT

-A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp --dport 445 -j ACCEPT

Samba Consits of three sections :

[Global] :- Contains settings that determine overall behaviour

[global] section of smb.conf file:

workgroup = YOUR-DOMAIN-CONTROLLER  
netbios name = YOUR-SAMBA-SERVER-NAME  
password server = IP-ADDRESS-OF-YOUR-DOMAIN-CONTROLLER  
encrypt passwords = Yes  
preferred master = No  
domain master = No  
Where,

* Workgroup: This controls what workgroup your server will appear to be in when queried by clients.
* netbios name : This sets the NetBIOS name by which a Samba server is known.
* encrypt passwords : This boolean controls (YES or NO value) whether encrypted passwords will be used with the client. Note that Windows NT 4.0 SP3 and above and also Windows 98 will by default expect encrypted passwords unless a registry entry is changed. This is what you need to use for Window XP/2000/2003 systems.

[homes] :- Default share for providing a home directory for all users.

[printers] :- Default share for exporting all printers on host via CIFS

**Checking the Status:**

Smbclient –L Localhost(IPADDRESS) –U username

smbclient //<server>/<share> -U <username>

/etc/fstab

//IP\_ADDRESS\_OF\_SERVER/Shared   /shared    smbfs    noauto,defaults  0  0

**Checking connected status** : smbstatus

**Checking Configuration file** : Testparm

**Q: – Explain "force group" parameter used in smb.conf?**

It will define the group id to be used for all file access in the place of the user’s primary group.

Q: – **Explain "force user" parameter used in smb.conf?**

It will define the user id to be used for all file access.

Q: – **What are the SAMBA server Types ?**

– Primary Domain Controller (PDC)  
– Backup Domain Controller (BDC)  
– ADS Domain Controller

**Explain the parameter "wins support = Yes" used in smb.conf?**

**Ans:** If the Samba server was configured to provide WINS support ("wins support = Yes"), then the WINS server is able to provide name resolution for all of the hosts that are not listed in the /etc/hosts file or within the DNS. Making this adjustment in the Name Service Switch configuration file (/etc/nsswitch.conf) allows the Linux system to query the WINS server for local name resolution. This saves manual adjustments to host files

## Example of smb.conf config file:

# This is the main Samba configuration file. You should read the  
# smb.conf(5) manual page in order to understand the options listed  
# here. Samba has a huge number of configurable options (perhaps too  
# many!) most of which are not shown in this example  
#  
# For a step to step guide on installing, configuring and using samba,  
# read the Samba-HOWTO-Collection. This may be obtained from:  
# http://www.samba.org/samba/docs/Samba-HOWTO-Collection.pdf  
#  
# Many working examples of smb.conf files can be found in the  
# Samba-Guide which is generated daily and can be downloaded from:  
# http://www.samba.org/samba/docs/Samba-Guide.pdf  
#  
# Any line which starts with a ; (semi-colon) or a # (hash)  
# is a comment and is ignored. In this example we will use a #  
# for commentry and a ; for parts of the config file that you  
# may wish to enable  
#  
# NOTE: Whenever you modify this file you should run the command "testparm"  
# to check that you have not made any basic syntactic errors.  
#  
#======================= Global Settings =====================================  
[global]  
  
# workgroup = NT-Domain-Name or Workgroup-Name, eg: MIDEARTH  
workgroup = mygroup  
  
# server string is the equivalent of the NT Description field  
server string = Samba Server  
  
# Security mode. Defines in which mode Samba will operate. Possible  
# values are share, user, server, domain and ads. Most people will want  
# user level security. See the Samba-HOWTO-Collection for details.  
; security = user  
  
# This option is important for security. It allows you to restrict  
# connections to machines which are on your local network. The  
# following example restricts access to two C class networks and  
# the "loopback" interface. For more examples of the syntax see  
# the smb.conf man page  
; hosts allow = 192.168.1. 192.168.2. 127.  
  
# If you want to automatically load your printer list rather  
# than setting them up individually then you'll need this  
load printers = yes  
  
# you may wish to override the location of the printcap file  
; printcap name = /etc/printcap  
  
# on SystemV system setting printcap name to lpstat should allow  
# you to automatically obtain a printer list from the SystemV spool  
# system  
; printcap name = lpstat  
  
# It should not be necessary to specify the print system type unless  
# it is non-standard. Currently supported print systems include:  
# bsd, cups, sysv, plp, lprng, aix, hpux, qnx  
; printing = cups  
  
# This option tells cups that the data has already been rasterized  
cups options = raw  
  
# Uncomment this if you want a guest account, you must add this to /etc/passwd  
# otherwise the user "nobody" is used  
; guest account = pcguest  
  
# this tells Samba to use a separate log file for each machine  
# that connects  
log file = /var/log/samba/%m.log  
  
# Put a capping on the size of the log files (in Kb).  
max log size = 50  
  
# Use password server option only with security = server  
# The argument list may include:  
# password server = My\_PDC\_Name [My\_BDC\_Name] [My\_Next\_BDC\_Name]  
# or to auto-locate the domain controller/s  
# password server = \*  
; password server = <NT-Server-Name>  
  
# Use the realm option only with security = ads  
# Specifies the Active Directory realm the host is part of  
; realm = MY\_REALM  
  
# Backend to store user information in. New installations should  
# use either tdbsam or ldapsam. smbpasswd is available for backwards  
# compatibility. tdbsam requires no further configuration.  
; passdb backend = tdbsam  
  
# Using the following line enables you to customise your configuration  
# on a per machine basis. The %m gets replaced with the netbios name  
# of the machine that is connecting.  
# Note: Consider carefully the location in the configuration file of  
# this line. The included file is read at that point.  
; include = /usr/local/samba/lib/smb.conf.%m  
  
# Configure Samba to use multiple interfaces  
# If you have multiple network interfaces then you must list them  
# here. See the man page for details.  
; interfaces = 192.168.12.2/24 192.168.13.2/24  
  
# Browser Control Options:  
# set local master to no if you don't want Samba to become a master  
# browser on your network. Otherwise the normal election rules apply  
; local master = no  
  
# OS Level determines the precedence of this server in master browser  
# elections. The default value should be reasonable  
; os level = 33  
  
# Domain Master specifies Samba to be the Domain Master Browser. This  
# allows Samba to collate browse lists between subnets. Don't use this  
# if you already have a Windows NT domain controller doing this job  
; domain master = yes  
  
# Preferred Master causes Samba to force a local browser election on startup  
# and gives it a slightly higher chance of winning the election  
; preferred master = yes  
  
# Enable this if you want Samba to be a domain logon server for  
# Windows95 workstations.  
; domain logons = yes  
  
# if you enable domain logons then you may want a per-machine or  
# per user logon script  
# run a specific logon batch file per workstation (machine)  
; logon script = %m.bat  
# run a specific logon batch file per username  
; logon script = %U.bat  
  
# Where to store roving profiles (only for Win95 and WinNT)  
# %L substitutes for this servers netbios name, %U is username  
# You must uncomment the [Profiles] share below  
; logon path = \\%L\Profiles\%U  
  
# Windows Internet Name Serving Support Section:  
# WINS Support - Tells the NMBD component of Samba to enable it's WINS Server  
; wins support = yes  
  
# WINS Server - Tells the NMBD components of Samba to be a WINS Client  
# Note: Samba can be either a WINS Server, or a WINS Client, but NOT both  
; wins server = w.x.y.z  
  
# WINS Proxy - Tells Samba to answer name resolution queries on  
# behalf of a non WINS capable client, for this to work there must be  
# at least one WINS Server on the network. The default is NO.  
; wins proxy = yes  
  
# DNS Proxy - tells Samba whether or not to try to resolve NetBIOS names  
# via DNS nslookups. The default is NO.  
dns proxy = no  
username map = /etc/samba/smbusers  
; security = user  
; encrypt passwords = yes  
; guest ok = no  
; guest account = nobody  
  
# These scripts are used on a domain controller or stand-alone  
# machine to add or delete corresponding unix accounts  
; add user script = /usr/sbin/useradd %u  
; add group script = /usr/sbin/groupadd %g  
; add machine script = /usr/sbin/adduser -n -g machines -c Machine -d /dev/null -s /bin/false %u  
; delete user script = /usr/sbin/userdel %u  
; delete user from group script = /usr/sbin/deluser %u %g  
; delete group script = /usr/sbin/groupdel %g  
  
  
#============================ Share Definitions ==============================  
[homes]  
comment = Home Directories  
browseable = no  
writeable = yes  
  
# Un-comment the following and create the netlogon directory for Domain Logons  
; [netlogon]  
; comment = Network Logon Service  
; path = /usr/local/samba/lib/netlogon  
; guest ok = yes  
; writable = no  
; share modes = no  
  
  
# Un-comment the following to provide a specific roving profile share  
# the default is to use the user's home directory  
;[Profiles]  
; path = /usr/local/samba/profiles  
; browseable = no  
; guest ok = yes  
  
  
# NOTE: If you have a BSD-style print system there is no need to  
# specifically define each individual printer  
[printers]  
comment = All Printers  
path = /usr/spool/samba  
browseable = no  
# Set public = yes to allow user 'guest account' to print  
; guest ok = no  
; writeable = no  
printable = yes  
  
# This one is useful for people to share files  
;[tmp]  
; comment = Temporary file space  
; path = /tmp  
; read only = no  
; public = yes  
  
# A publicly accessible directory, but read only, except for people in  
# the "staff" group  
;[public]  
; comment = Public Stuff  
; path = /home/samba  
; public = yes  
; writable = yes  
; printable = no  
; write list = @staff  
  
# Other examples.  
#  
# A private printer, usable only by fred. Spool data will be placed in fred's  
# home directory. Note that fred must have write access to the spool directory,  
# wherever it is.  
;[fredsprn]  
; comment = Fred's Printer  
; valid users = fred  
; path = /homes/fred  
; printer = freds\_printer  
; public = no  
; writable = no  
; printable = yes  
  
# A private directory, usable only by fred. Note that fred requires write  
# access to the directory.  
;[fredsdir]  
; comment = Fred's Service  
; path = /usr/somewhere/private  
; valid users = fred  
; public = no  
; writable = yes  
; printable = no  
  
# a service which has a different directory for each machine that connects  
# this allows you to tailor configurations to incoming machines. You could  
# also use the %U option to tailor it by user name.  
# The %m gets replaced with the machine name that is connecting.  
;[pchome]  
; comment = PC Directories  
; path = /usr/pc/%m  
; public = no  
; writable = yes  
  
# A publicly accessible directory, read/write to all users. Note that all files  
# created in the directory by users will be owned by the default user, so  
# any user with access can delete any other user's files. Obviously this  
# directory must be writable by the default user. Another user could of course  
# be specified, in which case all files would be owned by that user instead.  
;[public]  
; path = /usr/somewhere/else/public  
; public = yes  
; only guest = yes  
; writable = yes  
; printable = no  
  
# The following two entries demonstrate how to share a directory so that two  
# users can place files there that will be owned by the specific users. In this  
# setup, the directory should be writable by both users and should have the  
# sticky bit set on it to prevent abuse. Obviously this could be extended to  
# as many users as required.  
;[myshare]  
; comment = Mary's and Fred's stuff  
; path = /usr/somewhere/shared  
; valid users = mary fred  
; public = no  
; writable = yes  
; printable = no  
; create mask = 0765  
  
  
[smb\_share]  
path = /home/smb\_share  
writeable = yes  
browseable = yes  
valid users = veronica  
comment = Share Directory For All Users

Linux uses a combination of kernel-level support and continuously running daemon processes to provide NFS file sharing, however, NFS support must be enabled in the Linux kernel in order to function. NFS uses *Remote Procedure Calls (RPC)* to route requests between clients and servers, meaning that the **portmap** **service** must be enabled and active at the proper runlevels for NFS communication to occur. Working with **portmap**, the following processes ensure that a given NFS connection is allowed and may proceed without error:

## NFS and portmap

NFS relies upon remote procedure calls (RPC) to function. The **portmap** **service** is **required** to map RPC requests to the correct services. RPC processes notify **portmap** when they start, revealing the port number they are monitoring and the RPC program numbers they expect to serve. The client system then contacts **portmap** on the server with a particular RPC program number. **portmap** then redirects the client to the proper port number to communicate with its intended **service**.

Because RPC-based services rely on **portmap** to make all connections with incoming client requests, **portmap** must be available before any of these services start. If, for some reason, the **portmap** **service** unexpectedly quits, restart **portmap** and any services running when it was started.

The **portmap** **service** can be used with TCP wrappers' hosts access files (/etc/hosts.allow and /etc/hosts.deny) to control which remote systems are permitted to use RPC-based services on the server. See [Chapter 15 *TCP Wrappers and xinetd*](http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/ch-tcpwrappers.html) for more information. Access control rules for **portmap** will affect all RPC-based services. Alternatively, it is possible to specify each of the NFS RPC daemons to be affected by a particular access control rule. The man pages for rpc.mountd and rpc.statd contain information regarding the precise syntax for these rules.

The rpcinfo command shows each RPC-based **service** with its port number, RPC program number, version, and IP protocol type (TCP or UDP).

o make sure the proper NFS RPC-based services are enabled for **portmap**, use the rpcinfo -p command:

|  |
| --- |
| program vers proto port  100000 2 tcp 111 portmapper  100000 2 udp 111 portmapper  100024 1 udp 1024 status  100024 1 tcp 1024 status  100011 1 udp 819 rquotad  100011 2 udp 819 rquotad  100005 1 udp 1027 mountd  100005 1 tcp 1106 mountd  100005 2 udp 1027 mountd  100005 2 tcp 1106 mountd  100005 3 udp 1027 mountd  100005 3 tcp 1106 mountd  100003 2 udp 2049 nfs  100003 3 udp 2049 nfs  100021 1 udp 1028 nlockmgr  100021 3 udp 1028 nlockmgr  100021 4 udp 1028 nlockmgr |

The -p option probes the portmapper on the specified host or defaults to localhost if no specific host is listed. Other options are available from the rpcinfo man page.

From this output, it is apparent that various NFS services are running. If one of the NFS services does not start up correctly, **portmap** will be unable to map RPC requests from clients for that **service** to the correct port. In many cases, restarting NFS as root (/sbin/**service** nfs restart) will cause those **service** to correctly register with **portmap** and begin working.

**The portmapper keeps a list of what services are running on what ports.**

**This list is used by a connecting machine to see what ports it wants to talk to**

**access certain services.**

**1. portmap: (111 TCP/UDP) (on nfs server)**  
It tells the RPC request, which port should be used for communication.  
/etc/rpc             <--this file contains all the rpc program numbers (program identifier)  
rpcinfo              <--this command can be used for workability checking  
  
  
**2.mountd: (random TCP) (on nfs server)**  
After the files, directories and/or filesystems have been exported, an NFS client must explicitly mount them before it can use them. It is handled by the mountd daemon. It answers RPC requests, it checks /etc/xtab file to find out what are exported. It shows the currently mounted filesystems, with the help of the command showmount. Port can be specified in /etc/services.  
  
  
**3. nfsd (2049 TCP) (on nfs server)**  
Services client requests for filesystem operations. Once a client's mount request has been validated by mountd, it is allowed to request various filesystem operations. These requests are handled on the server side by nfsd. If the /etc/exports file does not exist, the nfsd and the rpc.mountd daemons will not start. You can get around this by creating an empty/etc/exports file.  
  
  
**4. lockd and statd (both on server and client)**  
lockd (file locking) and statd (file lock recovery at crash) are on both server and client, and they run as a team. The lockd daemon on the client sends lock requests to the server lock daemon through the RPC. The lockd daemon then asks the statd (status monitor) daemon for monitor service. The statd daemon interacts with the lockd daemon to provide crash and recovery functions for the locking services  
  
The status monitor maintains information about the location of connections and the status in the /var/statmon/sm, /var/statmon/sm.bak and the /var/statmon/state file. When statd restarted it queries these files and tries to reestablish the connection it had prior to termination. To restart the statd daemon, and subsequently the lockd daemon, without prior knowledge of existing locks or status, delete these files before restarting the statd daemon.  
  
The statd daemon should always be started before the lockd daemon.  
  
  
**5. biod (on nfs client)**  
Not needed anymore, it no longer plays an active role in management of the NFS client subsystem, because an NFS client internally manages its I/O operations to NFS servers. (The biod daemon is retained for compatibility reasons, becuase earlier versions might have scripts that invoke biod.) The biod daemon might be removed in future AIX  releases. (man biod)  
  
(chnfs and biod has a parameter: NumberofBiod, which specifies the number of biod threads on the client. This option has no effect and should not be used. If needed number of biod threads should be set as a mount option: mount -o biods=16 ...)

**Port No 2049**

**Portmapper 111**

## exports file 🡪  **/etc/exports**

**‘/etc/sysconfig/nfs**‘ is the file through which we can fix ports for **RQUOTAD\_PORT, MOUNTD\_PORT, LOCKD\_TCPPORT, LOCKD\_UDPPORT and STATD\_PORT**

**SHARE PATH : CLIENT(OPTIONS):-**

ro: The directory is shared read only; the client machine will not be able to write to it. This is the default.  
rw: The client machine will have read and write access to the directory.  
root\_squash: By default, any file request made by user root on the client machine is treated as if it is made by user nobody on the server. (Exactly which UID the request is mapped to depends on the UID of user “nobody” on the server, not the client.)  
no\_root\_squash : if this option is used , then root on the client machine will have the same level of access to the files on the system as root on the server. This can have serious security implications, although it may be necessary if you want to perform any administrative work on the client machine that involves the exported directories. You should not specify this option without a good reason.  
no\_subtree\_check : If only part of a volume is exported, a routine called subtree checking verifies that a file that is requested from the client is in the appropriate part of the volume. If the entire volume is exported, disabling this check will speed up transfers.  
sync : Replies to the NFS request only after all data has been written to disk. This is much safer than async, and is the default in all nfsutils versions after 1.0.0.  
async : Replies to requests before the data is written to disk. This improves performance, but results in lost data if the server goes down.  
no\_wdelay : NFS has an optimization algorithm that delays disk writes if NFS deduces a likelihood of a related write request soon arriving. This saves disk writes and can speed performance  
wdelay : Negation of no\_wdelay , this is default  
nohide : Normally, if a server exports two filesystems one of which is mounted on the other, then the client will have to mount both filesystems explicitly to get access to them. If it just mounts the parent, it will see an empty directory at the place where the other filesystem is mounted. That filesystem is “hidden”. Setting the nohide option on a filesystem causes it not to be hidden, and an appropriately authorised client will be able to move from the parent to that filesystem without noticing the change.  
hide : Negation of nohide This is the default

all\_squash : Treats as Anonymous user

Examples :-

|  |  |
| --- | --- |
| /home/nfs/ 10.1.1.55(rw,sync) | export /home/nfs directory for host with an IP address 10.1.1.55 with read, write permissions, and synchronized mode |
| /home/nfs/ 10.1.1.0/24(ro,sync) | export /home/nfs directory for network 10.1.1.0 with netmask 255.255.255.0 with read only permissions and synchronized mode |
| /home/nfs/ 10.1.1.55(rw,sync) 10.1.1.10(ro,sync) | export /home/nfs directory for host with IP 10.1.1.55with read, write permissions, synchronized mode, and also export /home/nfs directory for another host with an IP address 10.1.1.10 with read only permissions and synchronized mode |
| /home/nfs/ 10.1.1.55(rw,sync,no\_root\_squash) | export /home/nfs directory for host with an IP address 10.1.1.55with read, write permissions, synchronized mode and the remote root user will be treated as a root and will be able to change any file and directory. |
| /home/nfs/ \*(ro,sync) | export /home/nfs directory for any host with read only permissions and synchronized mode |
| /home/nfs/ \*.linuxcareer.com(ro,sync) | export /home/nfs directory for any host within linuxconfig.org domain with a read only permission and synchronized mode |
| /home/nfs/ foobar(rw,sync) | export /home/nfs directory for hostname foobar with read, write permissions and synchronized mode |

## Restart NFS daemon

# /etc/init.d/nfs restart

If you later decide to add more NFS exports to the /etc/exports file, you will need to either restart NFS daemon or run command exportfs:

# exportfs -ra

Mount remote file system on client

First we need to create a mount point:

# mkdir /home/nfs\_local

If you are sure that the NFS client and mount point are ready, you can run the mount command to mount exported NFS remote file system:

# mount 10.1.1.50:/home/nfs /home/nfs\_local

In case that you need to specify a filesystem type you can do this by:

# mount -t nfs 10.1.1.50:/home/nfs /home/nfs\_local

You may also get and an error message:

mount: mount to NFS server failed: timed out (retrying).

This may mean that your server supports higher NFS version and therefore you need to pass one extra argument to your nfs client mount command. In this example we use nfs version 3:

# mount -t nfs -o nfsvers=3 10.1.1.50:/home/nfs /home/nfs\_local

In any case now you should be able to access a remote /home/nfs directory locally on your NFS client.

# ls /home/nfs\_local/

nfs-test-file

# cd /home/nfs\_local/

# ls

nfs-test-file

# touch test

touch: cannot touch `test': Read-only file system

The above output proves that a remote NFS export is mounted and that we can access it by navigating to a local /home/nfs\_local/ directory. Please notice that the touch command reports that the filesystem is mounted as read-only which was exactly our intention.

6. Configure automount

To make this completely transparent to end users, you can automount the NFS file system every time a user boots a Linux system, or you can also use PAM modules to mount once a user logs in with a proper username and password. In this situation just edit /etc/fstab to mount system automatically during a system boot. You can use your favorite editor and create new line like this within /etc/fstab:

10.1.1.50:/home/nfs /home/nfs\_local/ nfs defaults 0 0

as before you also use echo command to do that:

# echo "10.1.1.50:/home/nfs /home/nfs\_local/ nfs defaults 0 0" >> /etc/fstab

# tail -1 /etc/fstab

10.1.1.50:/home/nfs /home/nfs\_local/ nfs defaults 0 0

##### **Important commands for NFS**

1. showmount -e : Shows the available shares on your local machine
2. showmount -e <server-ip or hostname>: Lists the available shares at the remote server
3. showmount -d : Lists all the sub directories
4. exportfs -v : Displays a list of shares files and options on a server
5. exportfs -a : Exports all shares listed in /etc/exports, or given name
6. exportfs -u : Unexports all shares listed in /etc/exports, or given name
7. exportfs -r : Refresh the server’s list after modifying /etc/exports

* **Soft Mount :** Consider we have mounted a NFS share using ‘soft mount’ . When a program or application requests a file from the NFS filesystem, NFS client daemons will try to retrieve the data from the NFS server. But, if it doesn’t get any response from the NFS server (due to any crash or failure of NFS server), the NFS client will report an error to the process on the client machine requesting the file access. The advantage of this mechanism is “fast responsiveness” as it doesn’t wait for the NFS server to respond. But, the main disadvantage of this method is data corruption or loss of data. So, this is not a recommended option to use.
* **Hard Mount :** Suppose we have mounted the NFS share using hard mount, it will repeatedly retry to contact the server. Once the server is back online the program will continue to execute undisturbed from the state where it was during server crash. We can use the mount option “intr” which allows NFS requests to be interrupted if the server goes down or cannot be reached. Hence the recommended settings are hard and intr options.

**How to check nfs server version ?**

Ans: **‘nfsstat -o all’** command shows all information about active versions of NFS.

**What is portmap?**

Ans: The**portmapper** keeps a list of what services are running on what ports. This list is used by a connecting machine to see what ports it wants to talk to access certain services.

**How to reexport all the directories of ‘/etc/exports’ file ?**

**exportfs –r**