

# **Practical Computing**

## **(UCS311)**

### **Assignment**

#### **4**



**Submitted By-**

**Name: Ananya Agarwal**

**Roll No: 102083036**

**Batch: 2CO14**

## Sol 1.

### /etc/exports file

The primary configuration for the NFS server is the /etc/exports file. This file is used to specify the directories that we want to share with the NFS clients.

For **example**, the following entry in the /etc/exports file would share the /usr/sharing/document directory with the NFS client- client01 (with the options of read-write) and the NFS client client02 (with the option of read-only):

```
# gedit /etc/exports
/usr/sharing/document client01(rw) client02(ro)
```

### Different options used in /etc/exports file, including these are:

- **root\_squash:** Prevents root users connected remotely from having root privileges and assigns them the user ID for the user **nfs-nobody**. This effectively "squashes"(moves) the power of the remote root user to the lowest local user, preventing unauthorized alteration of files on the remote server.
- **no\_root\_squash:** Map the root user and group account from the NFS client to the local root and group accounts.
- **rw:** Share as read-write. This is a default option.  
**Example:**  
**/another/exported/directory 192.168.0.57(rw,sync)**
- **ro:** Share as read-only. Remote hosts are not able to make changes to the data shared on the file system. To allow hosts to make changes to the file system, the read/write (**rw**) option must be specified.
- **sync:** The server waits before responding to the client until the request is written to disk or stable storage (e.g. disc drive) i.e. the respective write operations are being waited for thus less likely to result in data loss.
- **async:** It has the opposite functionality as of sync. It allows the server to reply to the NFS client as soon as it has processed the I/O request and sent it to the local file-system i.e. it does not wait for the data to be written to stable storage before responding to the NFS client. This can save time for I/O requests and improve performance. But it is more likely to result in data loss.
- **no\_subtree\_check:** This option disables subtree checking, which has mild security implications, but can improve reliability in some circumstances.
- **subtree\_check:** If a subdirectory of a filesystem is exported, but the whole filesystem isn't then whenever a NFS request arrives, the server must check not only that the

accessed file is in the appropriate filesystem (which is easy) but also that it is in the exported tree (which is harder).

- Additionally, other options are available where no default value is specified. These include **allow access from insecure ports**, and **allow insecure file locks** (necessary for certain early NFS client implementations).

```
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes      hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4       gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
/mnt/server_sharing 192.168.43.240(rw,sync,no_root_squash)
```

## Sol 2.

To check available NFS share on local & remote machine:

To check whether the installed package on the server is running or not:

```
server@server:~$ sudo -i
[sudo] password for server:
root@server:~# cd ..
root@server:/# service nfs-kernel-server status
● nfs-server.service - NFS server and services
   Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor p>
   Drop-In: /run/systemd/generator/nfs-server.service.d
            └─order-with-mounts.conf
   Active: active (exited) since Sun 2020-09-27 15:12:10 IST; 1h 13min ago
   Process: 622 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUC>
   Rhythmbox: 627 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSARGS (code=exited, statu>
   : 627 (code=exited, status=0/SUCCESS)

Sep 27 15:12:07 server systemd[1]: Starting NFS server and services...
Sep 27 15:12:07 server exportfs[622]: exportfs: /etc/exports [2]: Neither 'sub>
Sep 27 15:12:07 server exportfs[622]: Assuming default behaviour ('no_subtre>
Sep 27 15:12:07 server exportfs[622]: NOTE: this default has changed since n>
Sep 27 15:12:10 server systemd[1]: Finished NFS server and services.
lines 1-14/14 (END)
```

To check whether the nfs protocol is running or not on the client side i.e. to check whether the Kernel is accepting the sharing process or not:

```
root@client:/# service nfs-common status
● nfs-common.service
   Loaded: masked (Reason: Unit nfs-common.service is masked.)
   Active: inactive (dead)
root@client:/#
```

To check whether the server has given client the access of sharing the directory. If yes, then list the details of the respective directory by using the command: # showmount -e <server\_ip\_address>

```
root@client:/# showmount -e 192.168.43.206
Export list for 192.168.43.206:
/mnt/server_sharing 192.168.43.240
root@client:/#
```

After mounting server side's directory onto the client so that client side can now access the server sides's specified folder, we have to check it by using df -h command:

```
root@client:/# mount 192.168.43.206:/mnt/server_sharing /mnt/client_sharing
root@client:/# df -h
Filesystem                                Size  Used Avail Use% Mounted on
udev                                     464M   0    464M   0% /dev
tmpfs                                    99M    1.4M  97M    2% /run
/dev/sda5                               9.3G   6.5G   2.4G   74% /
tmpfs                                    491M    0    491M   0% /dev/shm
tmpfs                                    5.0M   4.0K   5.0M   1% /run/lock
tmpfs                                    491M    0    491M   0% /sys/fs/cgroup
/dev/loop2                             241M   241M    0 100% /snap/gnome-3-34-1804
/24
/dev/loop1                              56M    56M    0 100% /snap/core18/1885
/dev/loop4                              63M    63M    0 100% /snap/gtk-common-them
es/1506
/dev/loop3                             256M   256M    0 100% /snap/gnome-3-34-1804
/36
/dev/loop0                              55M    55M    0 100% /snap/core18/1705
/dev/loop5                              50M    50M    0 100% /snap/snap-store/433
/dev/loop6                              50M    50M    0 100% /snap/snap-store/467
/dev/loop7                              28M    28M    0 100% /snap/snapd/7264
/dev/loop8                              31M    31M    0 100% /snap/snapd/9279
/dev/sda1                               511M   4.0K   511M   1% /boot/efi
tmpfs                                    99M    24K   99M   1% /run/user/1000
/dev/sr0                                58M    58M    0 100% /media/client/VBox_GA
s_6.1.12
192.168.43.206:/mnt/server_sharing      9.3G   6.4G   2.5G   73% /mnt/client_sharing
root@client:/#
```

### Sol 3.

To show all the directories of 'etc/export' file to be re-exported use the command:

**exportfs -r.**

**Reexport** all directories. It synchronizes /var/lib/nfs/etab with /etc/exports. It removes entries in /var/lib/nfs/etab which are deleted from /etc/exports, and remove any entries from the kernel export table which are no longer valid.

```
root@server:/# exportfs -r
exportfs: /etc/exports [2]: Neither 'subtree_check' or 'no_subtree_check' speci
fied for export "192.168.43.240:/mnt/server_sharing".
  Assuming default behaviour ('no_subtree_check').
  NOTE: this default has changed since nfs-utils version 1.0.x
```

### Sol 4.

- To mount an existing file system of one host into another, we use **NFS** which in the background work with **NFS protocol** and let you mount an existing file system on NFS Server to the Client Server.
- Now, after file-system has been mounted, there are several useful options. One of them is “**soft mount**” and “**hard mount**” which are used to define how the NFS client should handle NFS server crash/failure.
- The client uses the **nfstimeout option** value to determine how long to wait for an NFS system call to respond before timing out; this setting applies to hard and soft mounts. The default is 0 seconds.
- **nfsstat** command can be used to check the details of the currently mounted nfs file system.

<b><u>Parameters</u></b>	<b><u>Soft Mount</u></b>	<b><u>Hard Mount</u></b>
<b>Description</b>	Here, NFS tries repeatedly to contact the server until a connection is established when the <i>nfstimeout</i> value is reached.	Here, the gateway or server will continue to attempt to reconnect if the file system is unreachable and will write the changes as soon as the server is reachable.
<b>Working &amp; Application</b>	<p>NFS client background processes 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.</p> <p>For <i>read-only</i> file-systems it is safe to use soft mount because it won't block the application; only return no-data or an error which can be detected by an application.</p>	<p>Once the server is back online, the program will continue to execute undisturbed from the state where it was during server crash.</p> <p>We can use the mount option 'intr' which allows NFS requests to be interrupted if the server goes down or cannot be reached. To kill the repeated NFS server call(in case server is not reachable for a very long time) , it is recommended to use hard with intr option can interrupt to ensure no loss of data.</p>
<b>Advantages</b>	<ul style="list-style-type: none"> <li>-Fast responsiveness as it doesn't wait for the NFS server to respond.</li> <li>-Data integrity.</li> <li>-Server keeps running without any impact on gateway performance.</li> </ul>	<ul style="list-style-type: none"> <li>-No data corruption</li> <li>-Once reached, it will continue the transaction, assuring the integrity of the system and data.</li> </ul>
<b>Disadvantages</b>	<ul style="list-style-type: none"> <li>-Loss of data since there are multiple servers accessing the partition, and it could simply return timeout if connection fails.</li> <li>-If for some reason the gateway cannot reach the file system after a period set by the <i>nfstimeout</i> parameter, then it will receive an error. However, if the file system goes down during the transaction, the transaction will stop and there may be some data corruption.</li> </ul>	<ul style="list-style-type: none"> <li>-There is a performance impact with the constant connection.</li> </ul>

### Sol 5.

To add the entry:

110.168.2.10:/data, exported by NFS server and be added to client/etc/fstab/ file:

Open /etc/fstab file using any editor and then write the required data onto the fstab file.

```
root@client:/# nano /etc/fstab
root@client:/#
```

In the /etc/fstab:

```
GNU nano 4.8 /etc/fstab Modified
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda5 during installation
UUID=474a5692-52f6-459a-ac82-90189c74cb81 / ext4 errors=remou>
# /boot/efi was on /dev/sda1 during installation
UUID=0BF3-AF58 /boot/efi vfat umask=0077 0 1
/swapfile none swap sw >
110.168.2.10:/data /mnt/data nfs defaults, netdev 0 0
```

### Sol 6.

#### About 'Portmap':

- Portmap service maps RPC (Remote Procedure Call) 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 while finding the errors. When a client wishes to make an *RPC* call to a given program number, it will first contact portmap on the server machine to determine the port number where *RPC* packets should be sent. Portmap must be started before any *RPC* servers are invoked.
- The client system then contacts portmap service 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.
- NFS and NIS are the typical service to need this portmap.

To check the status use command-

Command: **# service portmap status**

Output: portmap (pid 8951) is running...

**For example-**

On server side:

```
root@server: /
root@server:/# service portmap status
● rpcbind.service - RPC bind portmap service
   Loaded: loaded (/lib/systemd/system/rpcbind.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-10-02 13:13:39 IST; 3h 29min ago
   TriggeredBy: ● rpcbind.socket
     Docs: man:rpcbind(8)
    Main PID: 474 (rpcbind)
      Tasks: 1 (limit: 2319)
     Memory: 2.0M
    CGroup: /system.slice/rpcbind.service
            └─474 /sbin/rpcbind -f -w

Oct 02 13:13:38 server systemd[1]: Starting RPC bind portmap service...
Oct 02 13:13:39 server systemd[1]: Started RPC bind portmap service.
lines 1-13/13 (END)
```

On client side:

```
root@client: /
root@client:/# service portmap status
● rpcbind.service - RPC bind portmap service
   Loaded: loaded (/lib/systemd/system/rpcbind.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-10-02 13:13:42 IST; 3h 28min ago
   TriggeredBy: ● rpcbind.socket
     Docs: man:rpcbind(8)
    Main PID: 454 (rpcbind)
      Tasks: 1 (limit: 2319)
     Memory: 2.0M
    CGroup: /system.slice/rpcbind.service
            └─454 /sbin/rpcbind -f -w

Oct 02 13:13:42 client systemd[1]: Starting RPC bind portmap service...
Oct 02 13:13:42 client systemd[1]: Started RPC bind portmap service.
lines 1-13/13 (END)
```



### Sol 7.

1. Auto-mount refers to the process of automatically mounting filesystems.
2. **autofs** is the program that controls the operation of the **automount** daemons (background processes).
3. **autofs** is a service in Linux like operating system which automatically mounts the file system and remote shares when it is accessed.

### Advantages-

- + No need to mount file system at all time, it is only mounted when it is in demand.
- + It offers better overall performance compared to static mounts via **fstab**.

### Steps:

#### Step -1:

Making directory name **db\_backup** in server and changing its owner along with changing permissions.

```
root@server: /mnt

root@server:/# cd /mnt
root@server:/mnt# mkdir db_backup
root@server:/mnt# ls
db_backup  file1.txt  server_sharing
root@server:/mnt# cd ..
root@server:/# chown nobody:nogroup db_backup
root@server:/# cd /mnt
root@server:/mnt#
root@server:/mnt# chown nobody:nogroup db_backup
root@server:/mnt# chmod 777 db_backup
root@server:/mnt# ls
db_backup  file1.txt  server_sharing
root@server:/mnt#
```

## Step -2:

- Making directory name dbstuff2 in client
- Installing autofs so as to start automounting
- Opening files auto.master and auto.nfsdb

```
root@client:/mnt# mkdir dbstuff2
root@client:/mnt# ls
client_sharing dbstuff1 dbstuff2
root@client:/mnt# apt-get install autofs
Reading package lists... Done
Building dependency tree
Reading state information... Done
autofs is already the newest version (5.1.6-2ubuntu0.1).
0 upgraded, 0 newly installed, 0 to remove and 348 not upgraded.
root@client:/mnt# cd ..
root@client:/# nano /etc/auto.master
root@client:/# nano /etc/auto.nfsdb
root@client:/# /etc/init.d/autofs start
Starting autofs (via systemctl): autofs.service.
```

```
GNU nano 4.8 /etc/auto.master
# two configuration items - one /etc/auto.master.d/extra.autofs file
# (using the same line format as the auto.master file)
# and a separate mount map (e.g. /etc/auto.extra or an auto.extra NIS map)
# that is referred to by the extra.autofs file.
#
+dir:/etc/auto.master.d
#
# If you have fedfs set up and the related binaries, either
# built as part of autofs or installed from another package,
# uncomment this line to use the fedfs program map to access
# your fedfs mounts.
#/nfs4 /usr/sbin/fedfs-map-nfs4 nobind
#
# Include central master map if it can be found using
# nsswitch sources.
#
# Note that if there are entries for /net or /misc (as
# above) in the included master map any keys that are the
# same will not be seen as the first read key seen takes
# precedence.
#
+auto.master
/mnt/dbstuff2 /etc/auto.nfsdb --timeout=180
```

```
GNU nano 4.8 /etc/auto.nfsdb
/mnt/db_backup -fstype=nfs4,rw,soft,intr 192.168.43.240:/mnt/db_backup
```

### Step -3:

To check whether the server has given client the access of sharing the directory. If yes, then list the details of the respective directory:

```
root@client:/# showmount -e 192.168.43.206
Export list for 192.168.43.206:
/mnt/db_backup      192.168.43.240
/mnt/server_sharing 192.168.43.240
```

### Step -4:

To check how many free partitions and how many free partitions are available:

```
root@client:/mnt/dbstuff2# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            967M   0    967M   0% /dev
tmpfs           199M  1.4M  198M   1% /run
/dev/sda5       9.3G  6.7G  2.2G  77% /
tmpfs           994M   0    994M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           994M   0    994M   0% /sys/fs/cgroup
/dev/loop0      55M   55M   0 100% /snap/core18/1705
/dev/loop1      56M   56M   0 100% /snap/core18/1885
/dev/loop2     256M  256M   0 100% /snap/gnome-3-34-1804/36
/dev/loop4      63M   63M   0 100% /snap/gtk-common-themes/15
06
/dev/loop3     218M  218M   0 100% /snap/gnome-3-34-1804/60
/dev/loop5      50M   50M   0 100% /snap/snap-store/433
/dev/loop6      50M   50M   0 100% /snap/snap-store/467
/dev/loop7      28M   28M   0 100% /snap/snapd/7264
/dev/loop8      31M   31M   0 100% /snap/snapd/9279
/dev/sda1       511M  4.0K  511M   1% /boot/efi
tmpfs           199M  28K  199M   1% /run/user/1000
/dev/sr0        58M   58M   0 100% /media/client/VBox_GAs_6.1
.12
192.168.43.206:/mnt/db_backup 9.3G  6.7G  2.2G  76% /mnt/dbstuff2
root@client:/mnt/dbstuff2#
```

### Sol 8.

- DNS (**Domain Name System**) translates human readable domain names (for example, www.amazon.com) to machine readable IP addresses (for example, 192.0.2.44), that computers use to connect to each other.
- DNS on a name server consists of a configuration file called named.conf, a resolver file named resolv.conf etc.

Path	Content and Purpose of File
/etc/named.conf	The configuration file specifies the type of server it is running on and the zones that it serves as a 'Master' or 'Slave'. It also defines security and logging options.
/etc/resolv.conf	This file resides on every DNS client (including DNS servers) and designates the servers, the client that queries for DNS information.
/etc/nsswitch.conf	The /etc/nsswitch.conf file defines the order in which to contact different name services
/etc/hosts	It is an operating system file that translate hostnames or domain names to IP addresses

## **SOL 9.**

- Synchronous or asynchronous mode can be set when the filesystem is mounted on the clients by simply putting sync or async on the mount command line or in the file /etc/fstab for the NFS filesystem. To change the option, first unmount the NFS filesystem, change the option, then remount the filesystem.
- If **option sync** is specified, the server waits before responding to the client until the request is written to disk or stable storage (e.g. disc drive) i.e. the respective write operations are being waited for. It follows the NFS protocol.

**Advantage:** Less chances of crash and data being lost or corrupted data if the NFS server goes down or network connectivity is lost.

- Whereas Asynchronous mode (async) allows the server to reply to the NFS client as soon as it has processed the I/O request and sent it to the local filesystem i.e. it does not wait for the data to be written to stable storage before responding to the NFS client. This can save time for I/O requests and improve performance.
- If we have a copy of the data somewhere, we can perhaps run asynchronously for better performance. If we don't have copies or the data cannot be easily or quickly reproduced, then perhaps synchronous mode is the better option.

### **For Example-**

Here a directory server\_sharing present in the mnt directory is ready to be shared with client side wherein the client side has permission of both reading and writing onto the file along with sync option wherein the respective write options are first written onto the disk and thus are being waited for. Moreover, no\_root\_squash option is used here to map the root user and group account from the NFS client to the local root and group accounts.

```
GNU nano 4.8 /etc/exports

# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4 gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
/mnt/server_sharing 192.168.43.240(rw,sync,no_root_squash)
```

### Sol 10.

We can use these different options at different steps to trace out the flaw while mounting a nfs share:

- All the detail which could be shared is in the export file and thus it is notified from the server side to the client side:

```
root@server:/# nano /etc/exports
root@server:/# exportfs -a
exportfs: /etc/exports [2]: Neither 'subtree_check' or 'no_subtree_check' speci
fied for export "192.168.43.240:/mnt/server_sharing".
Assuming default behaviour ('no_subtree_check').
NOTE: this default has changed since nfs-utils version 1.0.x
```

- To check whether the installed package on the server is running or not:

```
server@server:~$ sudo -i
[sudo] password for server:
root@server:~# cd ..
root@server:/# service nfs-kernel-server status
● nfs-server.service - NFS server and services
   Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor p>
   Drop-In: /run/systemd/generator/nfs-server.service.d
            └─order-with-mounts.conf
   Active: active (exited) since Sun 2020-09-27 15:12:10 IST; 1h 13min ago
   Process: 622 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUC>
            627 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSDARGS (code=exited, statu>
            627 (code=exited, status=0/SUCCESS)

Sep 27 15:12:07 server systemd[1]: Starting NFS server and services...
Sep 27 15:12:07 server exportfs[622]: exportfs: /etc/exports [2]: Neither 'sub>
Sep 27 15:12:07 server exportfs[622]: Assuming default behaviour ('no_subtre>
Sep 27 15:12:07 server exportfs[622]: NOTE: this default has changed since n>
Sep 27 15:12:10 server systemd[1]: Finished NFS server and services.
lines 1-14/14 (END)
```

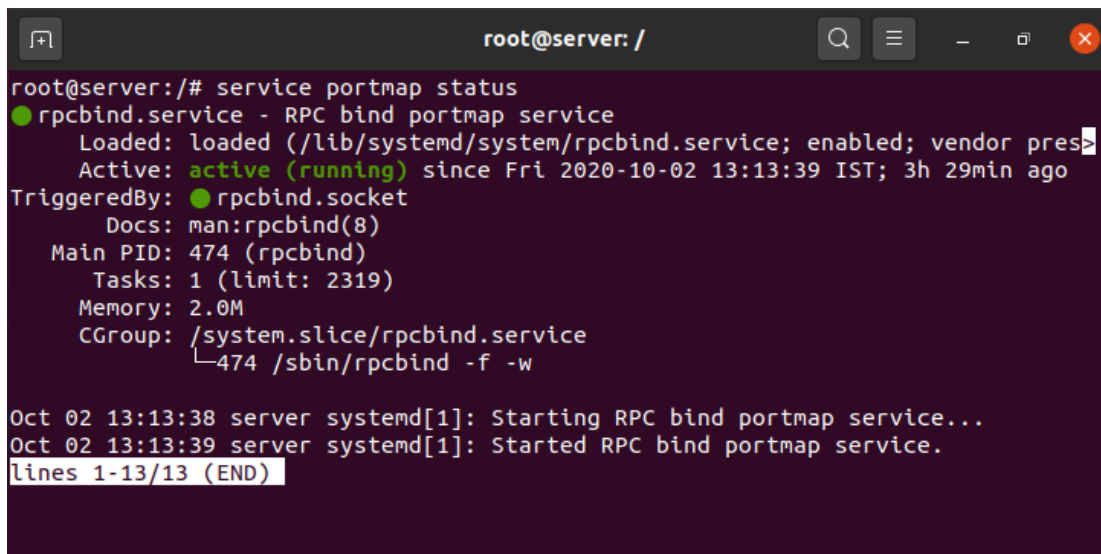
- To check whether the nfs protocol is running or not on the client side i.e. to check whether the Kernel is accepting the sharing process or not:

```
root@client:/# service nfs-common status
● nfs-common.service
   Loaded: masked (Reason: Unit nfs-common.service is masked.)
   Active: inactive (dead)
root@client:/#
```

- To check whether the server has given client the access of sharing the directory. If yes, then list the details of the respective directory by using the command: # showmount -e <server\_ip\_address>

```
root@client:/# showmount -e 192.168.43.206
Export list for 192.168.43.206:
/mnt/server_sharing 192.168.43.240
root@client:/#
```

- Portmap service maps RPC (Remote Procedure Call) 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 while finding the errors:



```
root@server: /
root@server:/# service portmap status
● rpcbind.service - RPC bind portmap service
   Loaded: loaded (/lib/systemd/system/rpcbind.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-10-02 13:13:39 IST; 3h 29min ago
     TriggeredBy: ● rpcbind.socket
       Docs: man:rpcbind(8)
    Main PID: 474 (rpcbind)
       Tasks: 1 (limit: 2319)
      Memory: 2.0M
     CGroup: /system.slice/rpcbind.service
            └─474 /sbin/rpcbind -f -w

Oct 02 13:13:38 server systemd[1]: Starting RPC bind portmap service...
Oct 02 13:13:39 server systemd[1]: Started RPC bind portmap service.
lines 1-13/13 (END)
```

```
root@client: /
root@client:/# service portmap status
● rpcbind.service - RPC bind portmap service
   Loaded: loaded (/lib/systemd/system/rpcbind.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-10-02 13:13:42 IST; 3h 28min ago
     TriggeredBy: ● rpcbind.socket
       Docs: man:rpcbind(8)
    Main PID: 454 (rpcbind)
      Tasks: 1 (limit: 2319)
     Memory: 2.0M
    CGroup: /system.slice/rpcbind.service
            └─454 /sbin/rpcbind -f -w

Oct 02 13:13:42 client systemd[1]: Starting RPC bind portmap service...
Oct 02 13:13:42 client systemd[1]: Started RPC bind portmap service.
lines 1-13/13 (END)
```

- **Ufw status (unified firewall) is the command to check for any unwanted service which is not in knowledge of server / Kernel because of which there will be a problem in mounting a nfs share. If these firewalls are active, clients's service request will not be processed by the server.**

```
root@server: ~
root@server:~# ufw status
Status: inactive
root@server:~# ufw enable
Firewall is active and enabled on system startup
```

If the ufw is enabled the client will stop responding.

```
root@client: ~
root@client:~# showmount -e 192.168.43.206
.....system hang...not responding.....
```

Again ufw is disabled so that the client's request are processed correctly by the server.

```
root@server: ~
root@server:~# ufw disable
Firewall stopped and disabled on system startup
root@server:~#
```

We are now able to see the files shared from the server side to the client side.

```
root@client:~# showmount -e 192.168.43.206
.....system hang...not responding.....Export list for 192.168.43.20
6:
/mnt/db_backup      192.168.43.240
/mnt/server_sharing 192.168.43.240
root@client:~# .....system hang...not responding.....
.....system: command not found
root@client:~# showmount -e 192.168.43.206
Export list for 192.168.43.206:
/mnt/db_backup      192.168.43.240
/mnt/server_sharing 192.168.43.240
root@client:~# █
```

- Sometimes the NFS server may only support NFSv3 connections. By default, the mount command uses NFSv4, which may result in the error. To avoid this, specify the NFSv3 while mounting the share.
- The IP addresses of the client & server systems must be mentioned correctly everywhere to make the proper network share amongst them.

END