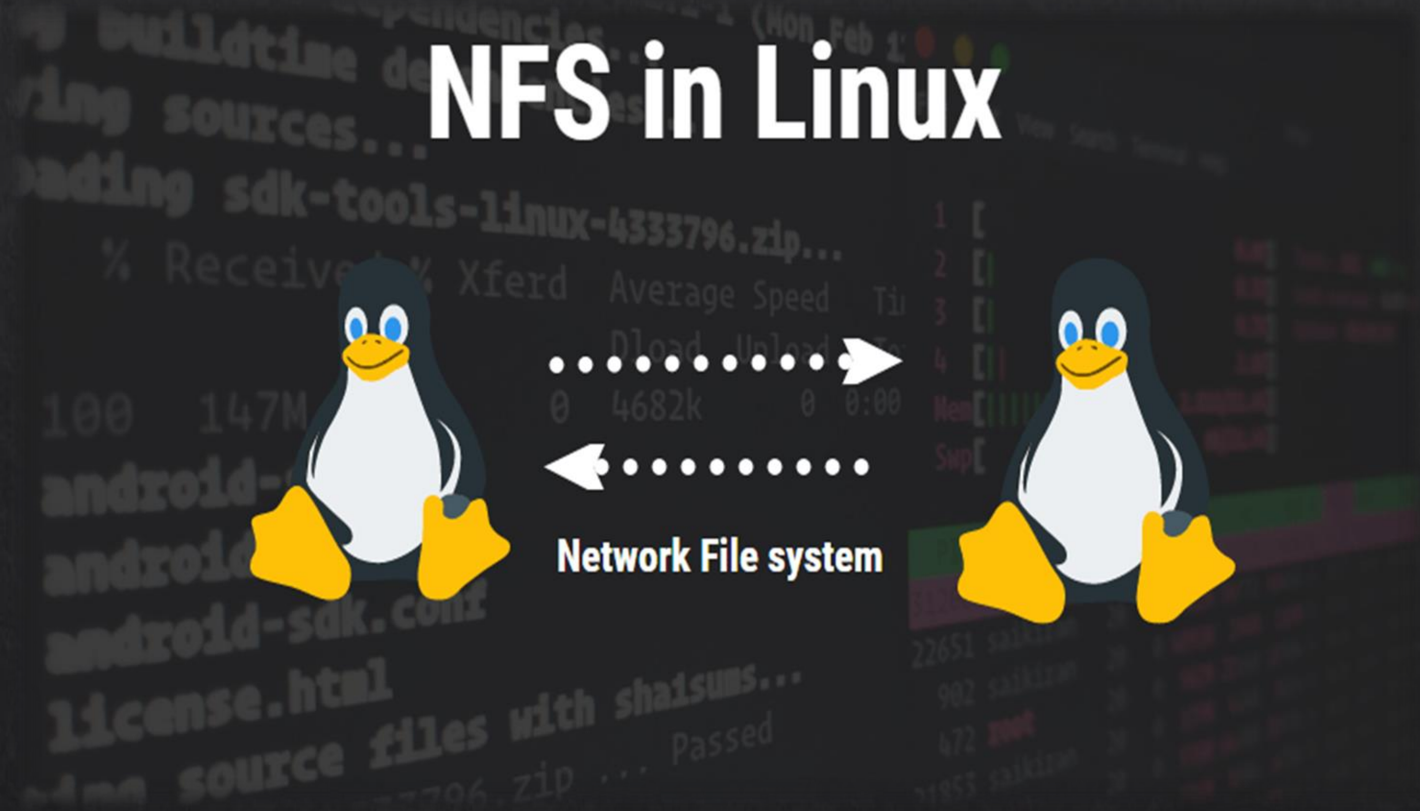
Aim :-

Setting up NFS server and client on Linux.



Created By :-

Name :- Chander Mohan Meena

Domain :- Btech cse (cloud computing and full stack

development)

College :- Poornima University

Network File System (NFS)

A Network File System (NFS) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally.
In Linux, NFS is a powerful protocol that lets you share directories between computers seamlessly. With NFS properly configured, moving files between computers becomes as easy as moving files around on the same machine.

Advantages of NFS :-

•**Centralized Data Storage**: NFS enables centralized management of shared files. Administrators can maintain data in one location, reducing workload.
•**Efficiency**: Easy setup using existing IP infrastructure. Users access remote files just like local files.
•**Reduced Disk Space**: Individual users require less local storage since software and files can be accessed remotely.
•**Granularity**: Fine-grained access control for files.

Disadvantages of NFS :-

•**Not Suitable for Sensitive Data**: NFS isn’t ideal for sharing sensitive data over public networks due to security concerns.
•**Lacks Hierarchical Storage Management**: NFS doesn’t support hierarchical storage management directly

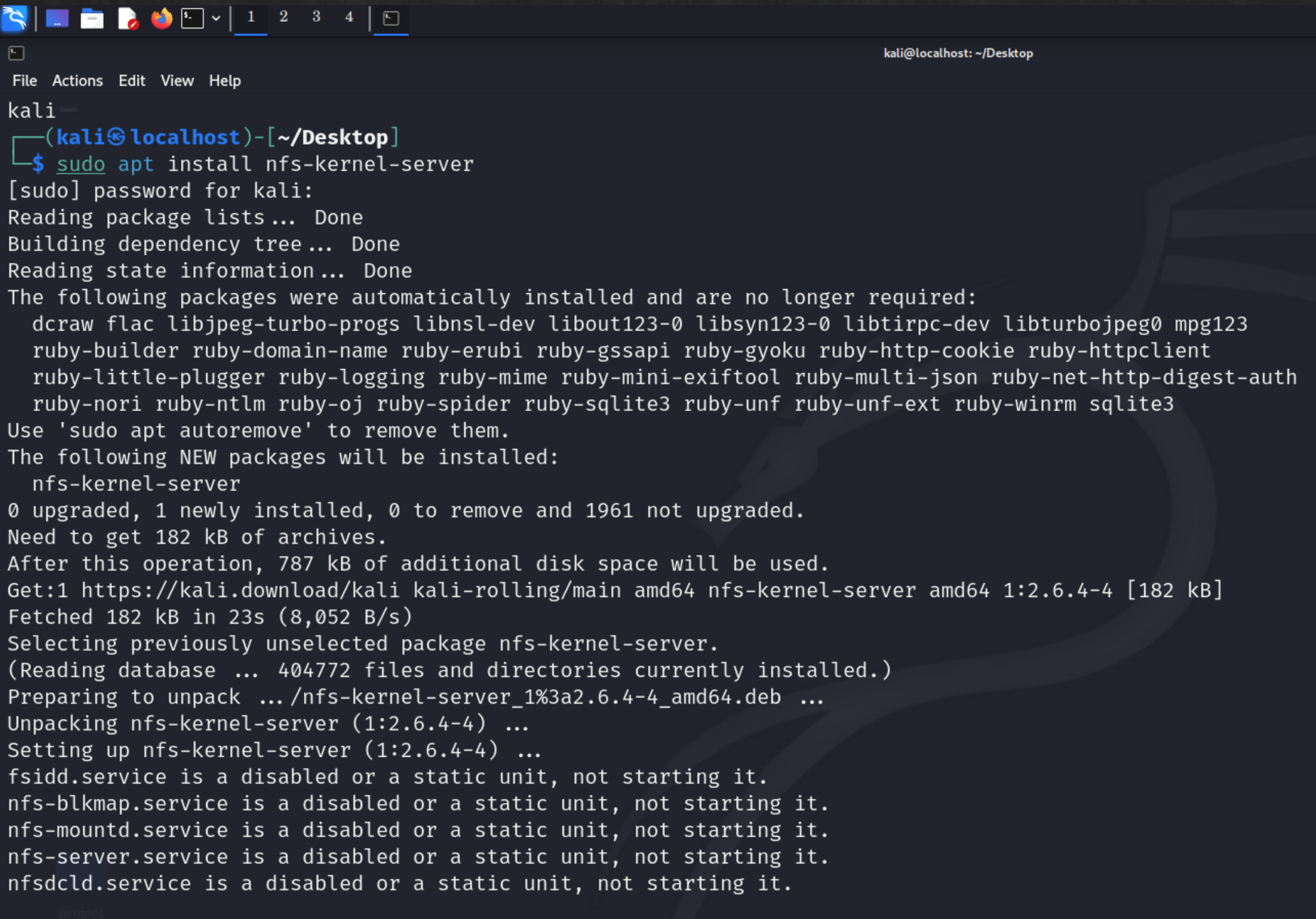
Procedure:-

- For setting up the NFS server here we have 2 Debian based system{kali linux and Ubuntu } which were connected with same IP
- Now we are considering kali linux as a server and Ubuntu an a clients

Step1 :-

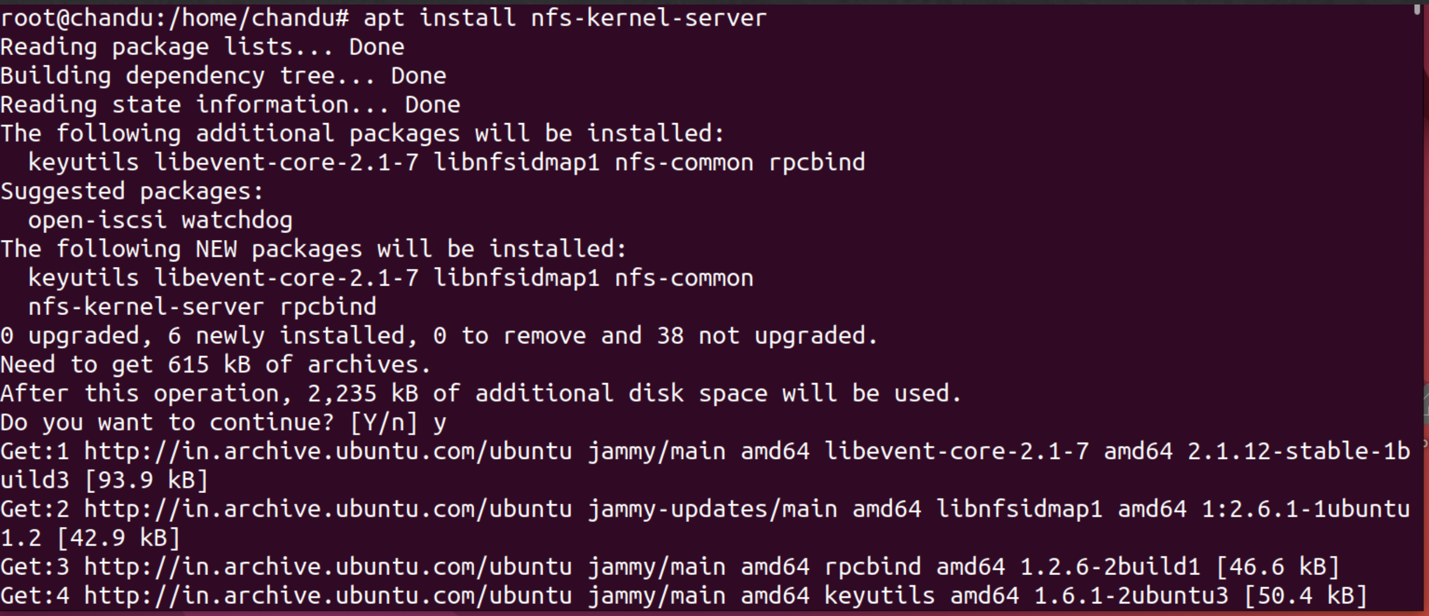
For linking the both of the system we have to install the services in our system both clients side and server side by using <<< sudo apt install nfs-kernel-server >>> command.

Server :-



```
kali
┌──(kali㉿localhost)-[~/Desktop]
└─$ sudo apt install nfs-kernel-server
[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  dcraw flac libjpeg-turbo-progs libnsl-dev libout123-0 libsyn123-0 libtirpc-dev libturbojpeg0 mpg123
  ruby-builder ruby-domain-name ruby-erubi ruby-gssapi ruby-gyoku ruby-http-cookie ruby-httpclient
  ruby-little-plugger ruby-logging ruby-mime ruby-mini-exiftool ruby-multi-json ruby-net-http-digest-auth
  ruby-nori ruby-ntlm ruby-oj ruby-spider ruby-sqlite3 ruby-unf ruby-unf-ext ruby-winrm sqlite3
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  nfs-kernel-server
0 upgraded, 1 newly installed, 0 to remove and 1961 not upgraded.
Need to get 182 kB of archives.
After this operation, 787 kB of additional disk space will be used.
Get:1 https://kali.download/kali kali-rolling/main amd64 nfs-kernel-server amd64 1:2.6.4-4 [182 kB]
Fetched 182 kB in 23s (8,052 B/s)
Selecting previously unselected package nfs-kernel-server.
(Reading database ... 404772 files and directories currently installed.)
Preparing to unpack .../nfs-kernel-server_1%3a2.6.4-4_amd64.deb ...
Unpacking nfs-kernel-server (1:2.6.4-4) ...
Setting up nfs-kernel-server (1:2.6.4-4) ...
fsidd.service is a disabled or a static unit, not starting it.
nfs-blkmap.service is a disabled or a static unit, not starting it.
nfs-mountd.service is a disabled or a static unit, not starting it.
nfs-server.service is a disabled or a static unit, not starting it.
nfsdcld.service is a disabled or a static unit, not starting it.
```

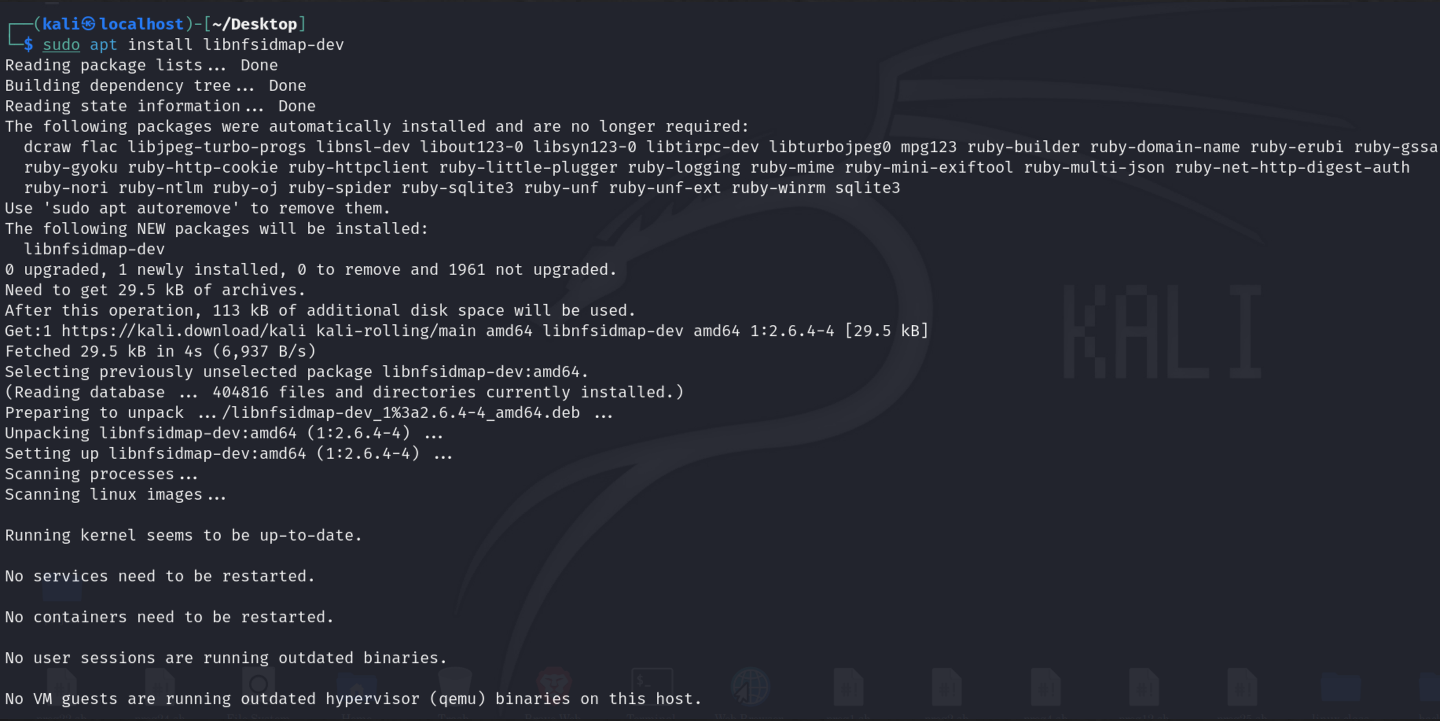
Client :-



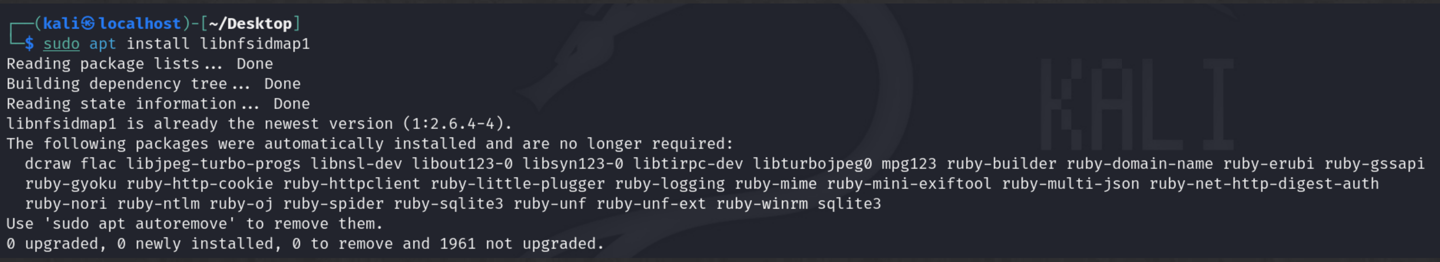
```
root@chandu:/home/chandu# apt install nfs-kernel-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  keyutils libevent-core-2.1-7 libnfsidmap1 nfs-common rpcbind
Suggested packages:
  open-iscsi watchdog
The following NEW packages will be installed:
  keyutils libevent-core-2.1-7 libnfsidmap1 nfs-common
  nfs-kernel-server rpcbind
0 upgraded, 6 newly installed, 0 to remove and 38 not upgraded.
Need to get 615 kB of archives.
After this operation, 2,235 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libevent-core-2.1-7 amd64 2.1.12-stable-1b
uild3 [93.9 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libnfsidmap1 amd64 1:2.6.1-1ubuntu
1.2 [42.9 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 rpcbind amd64 1.2.6-2build1 [46.6 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 keyutils amd64 1.6.1-2ubuntu3 [50.4 kB]
```

After the installation will complete the we have also install libnfsidmap which holds multiple methods of mapping names to id’s and it mainly for NFSv4. by using <<< sudo apt instsll libnfsidmap-dev >>> and <<< sudo apt install libnfsidmap1 >>> command in both of the systems.

Server :-

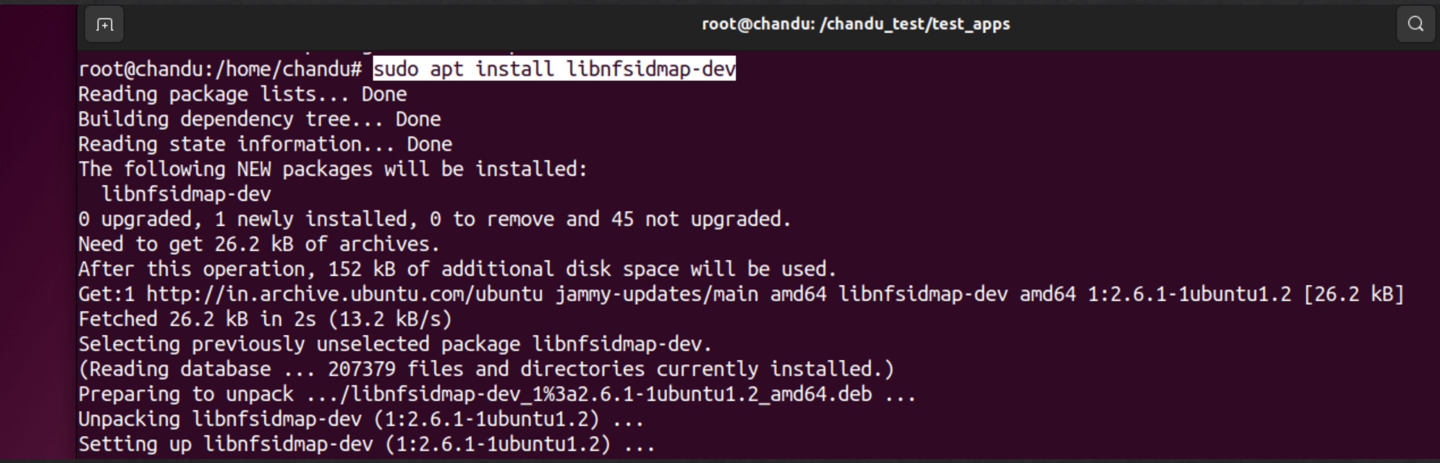


```
┌──(kali㉿localhost)-[~/Desktop]
└─$ sudo apt install libnfsidmap-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  dcraw flac libjpeg-turbo-progs libnsl-dev libout123-0 libsyn123-0 libtirpc-dev libturbojpeg0 mpg123 ruby-builder ruby-domain-name ruby-erubi ruby-gssa
  ruby-gyoku ruby-http-cookie ruby-httpclient ruby-little-plugger ruby-logging ruby-mime ruby-mini-exiftool ruby-multi-json ruby-net-http-digest-auth
  ruby-nori ruby-ntlm ruby-oj ruby-spider ruby-sqlite3 ruby-unf ruby-unf-ext ruby-winrm sqlite3
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  libnfsidmap-dev
0 upgraded, 1 newly installed, 0 to remove and 1961 not upgraded.
Need to get 29.5 kB of archives.
After this operation, 113 kB of additional disk space will be used.
Get:1 https://kali.download/kali kali-rolling/main amd64 libnfsidmap-dev amd64 1:2.6.4-4 [29.5 kB]
Fetched 29.5 kB in 4s (6,937 B/s)
Selecting previously unselected package libnfsidmap-dev:amd64.
(Reading database ... 404816 files and directories currently installed.)
Preparing to unpack .../libnfsidmap-dev_1%3a2.6.4-4_amd64.deb ...
Unpacking libnfsidmap-dev:amd64 (1:2.6.4-4) ...
Setting up libnfsidmap-dev:amd64 (1:2.6.4-4) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

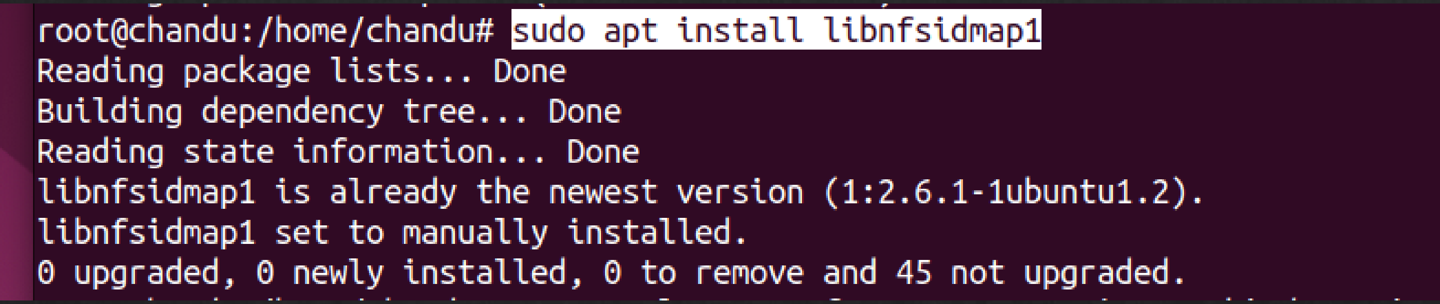


```
┌──(kali㉿localhost)-[~/Desktop]
└─$ sudo apt install libnfsidmap1
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libnfsidmap1 is already the newest version (1:2.6.4-4).
The following packages were automatically installed and are no longer required:
  dcraw flac libjpeg-turbo-progs libnsl-dev libout123-0 libsyn123-0 libtirpc-dev libturbojpeg0 mpg123 ruby-builder ruby-domain-name ruby-erubi ruby-gssapi
  ruby-gyoku ruby-http-cookie ruby-httpclient ruby-little-plugger ruby-logging ruby-mime ruby-mini-exiftool ruby-multi-json ruby-net-http-digest-auth
  ruby-nori ruby-ntlm ruby-oj ruby-spider ruby-sqlite3 ruby-unf ruby-unf-ext ruby-winrm sqlite3
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 1961 not upgraded.
```

Client :-



```
root@chandu: /chandu_test/test_apps
root@chandu:/home/chandu# sudo apt install libnfsidmap-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  libnfsidmap-dev
0 upgraded, 1 newly installed, 0 to remove and 45 not upgraded.
Need to get 26.2 kB of archives.
After this operation, 152 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libnfsidmap-dev amd64 1:2.6.1-1ubuntu1.2 [26.2 kB]
Fetched 26.2 kB in 2s (13.2 kB/s)
Selecting previously unselected package libnfsidmap-dev.
(Reading database ... 207379 files and directories currently installed.)
Preparing to unpack .../libnfsidmap-dev_1%3a2.6.1-1ubuntu1.2_amd64.deb ...
Unpacking libnfsidmap-dev (1:2.6.1-1ubuntu1.2) ...
Setting up libnfsidmap-dev (1:2.6.1-1ubuntu1.2) ...
```



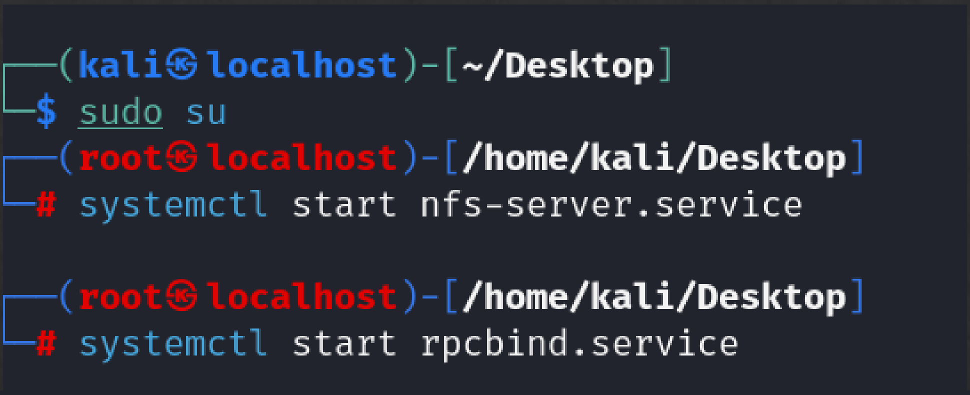
```
root@chandu:/home/chandu# sudo apt install libnfsidmap1
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libnfsidmap1 is already the newest version (1:2.6.1-1ubuntu1.2).
libnfsidmap1 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 45 not upgraded.
```

Step2 :-

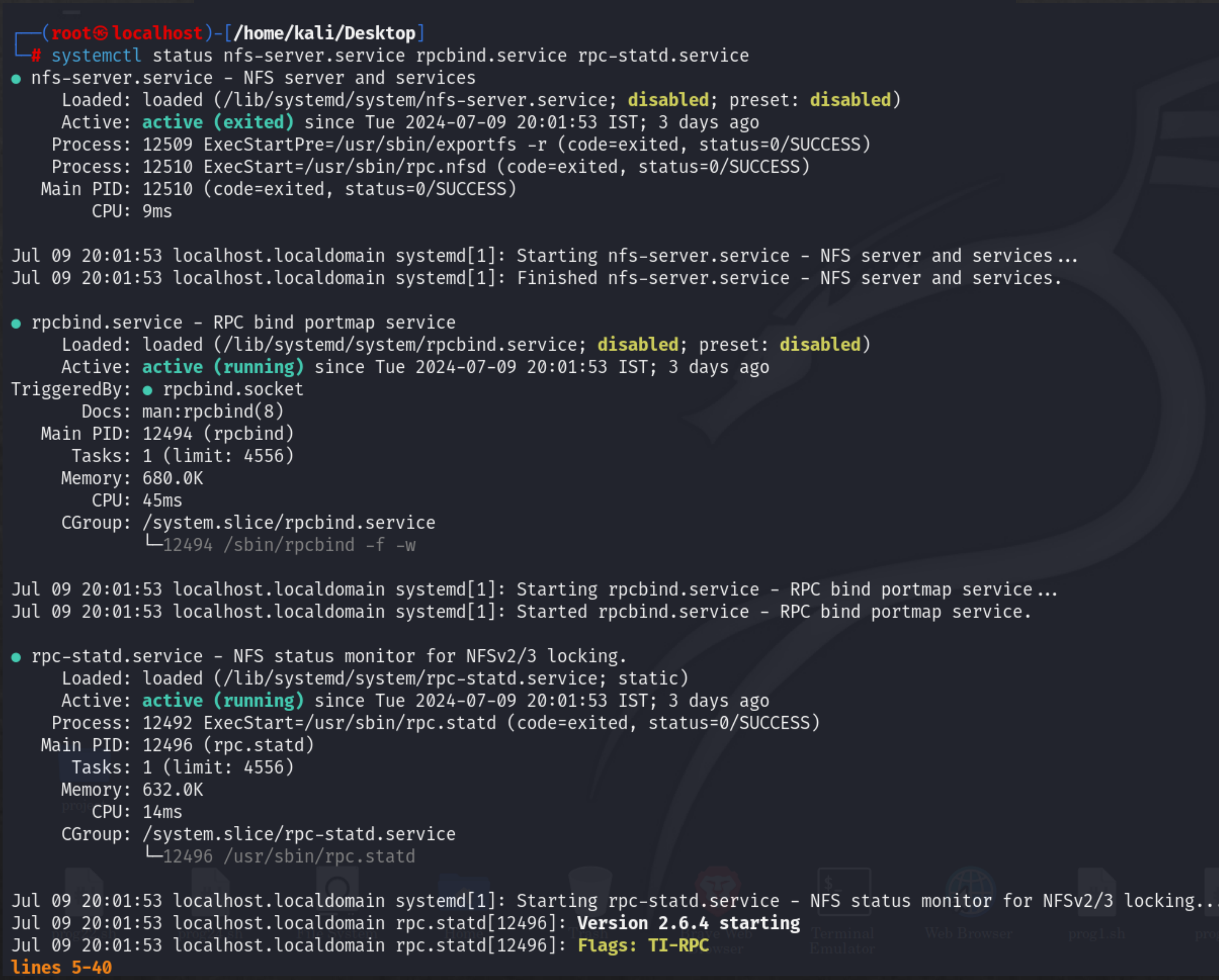
- Now , we have to enable all the services for establishing the connection in both clients and server side .
- For starting the services with

1. <<< sudo systemctl start nfs-server.service >>>
2. <<< sudo systemctl start rpcbind.service >>>
3. <<< sudo systemctl start nfs-idmapd.service >>>

Server :-



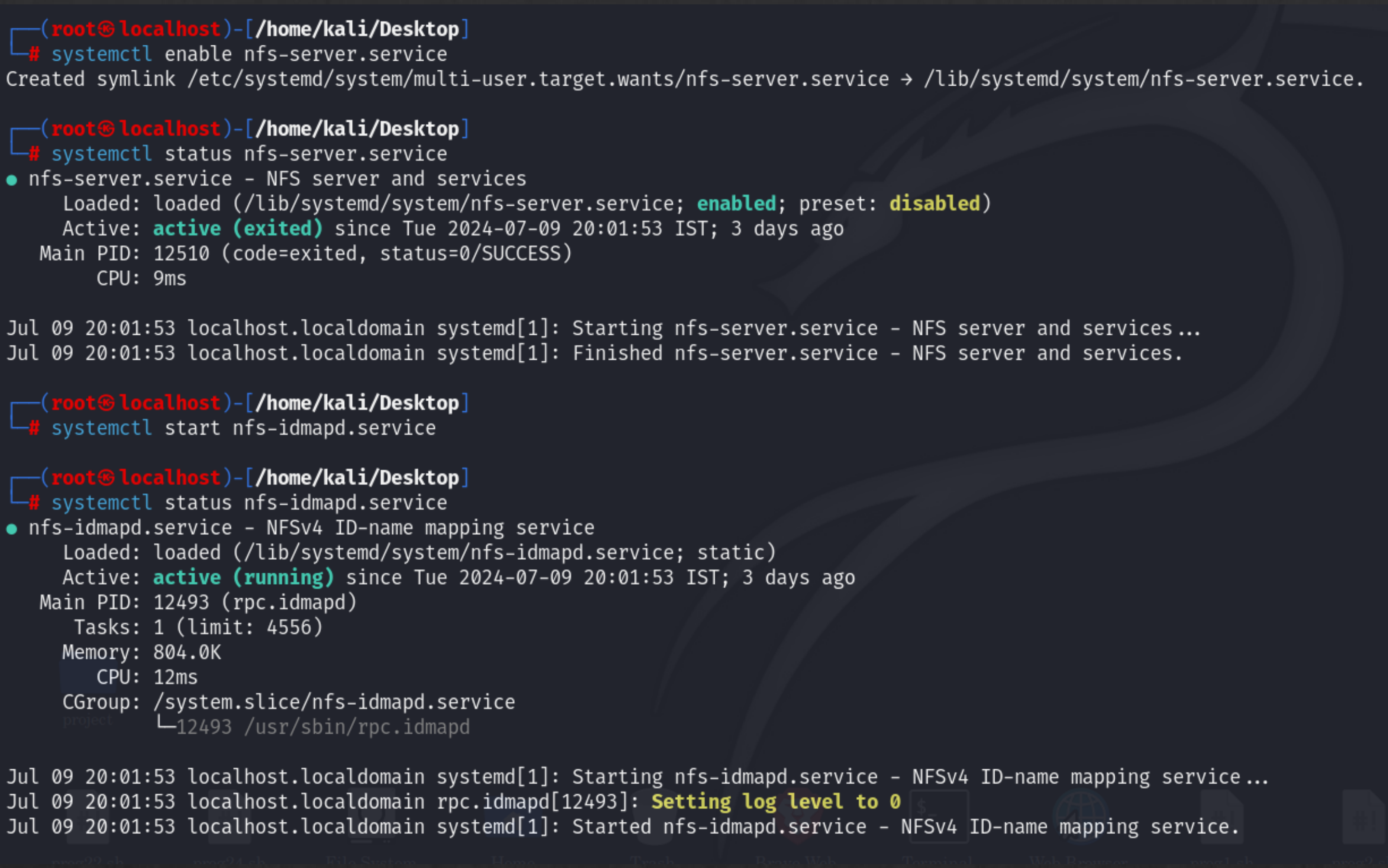
```
┌──(kali㉿localhost)-[~/Desktop]
└─$ sudo su
┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl start nfs-server.service

┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl start rpcbind.service
```



```
┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl status nfs-server.service rpcbind.service rpc-statd.service
● nfs-server.service - NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; disabled; preset: disabled)
     Active: active (exited) since Tue 2024-07-09 20:01:53 IST; 3 days ago
    Process: 12509 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
    Process: 12510 ExecStart=/usr/sbin/rpc.nfsd (code=exited, status=0/SUCCESS)
   Main PID: 12510 (code=exited, status=0/SUCCESS)
        CPU: 9ms

Jul 09 20:01:53 localhost.localdomain systemd[1]: Starting nfs-server.service - NFS server and services...
Jul 09 20:01:53 localhost.localdomain systemd[1]: Finished nfs-server.service - NFS server and services.

● rpcbind.service - RPC bind portmap service
     Loaded: loaded (/lib/systemd/system/rpcbind.service; disabled; preset: disabled)
     Active: active (running) since Tue 2024-07-09 20:01:53 IST; 3 days ago
TriggeredBy: ● rpcbind.socket
       Docs: man:rpcbind(8)
   Main PID: 12494 (rpcbind)
      Tasks: 1 (limit: 4556)
     Memory: 680.0K
        CPU: 45ms
     CGroup: /system.slice/rpcbind.service
             └─12494 /sbin/rpcbind -f -w

Jul 09 20:01:53 localhost.localdomain systemd[1]: Starting rpcbind.service - RPC bind portmap service...
Jul 09 20:01:53 localhost.localdomain systemd[1]: Started rpcbind.service - RPC bind portmap service.

● rpc-statd.service - NFS status monitor for NFSv2/3 locking.
     Loaded: loaded (/lib/systemd/system/rpc-statd.service; static)
     Active: active (running) since Tue 2024-07-09 20:01:53 IST; 3 days ago
    Process: 12492 ExecStart=/usr/sbin/rpc.statd (code=exited, status=0/SUCCESS)
   Main PID: 12496 (rpc.statd)
      Tasks: 1 (limit: 4556)
     Memory: 632.0K
        CPU: 14ms
     CGroup: /system.slice/rpc-statd.service
             └─12496 /usr/sbin/rpc.statd

Jul 09 20:01:53 localhost.localdomain systemd[1]: Starting rpc-statd.service - NFS status monitor for NFSv2/3 locking..
Jul 09 20:01:53 localhost.localdomain rpc.statd[12496]: Version 2.6.4 starting
Jul 09 20:01:53 localhost.localdomain rpc.statd[12496]: Flags: TI-RPC
lines 5-40
```

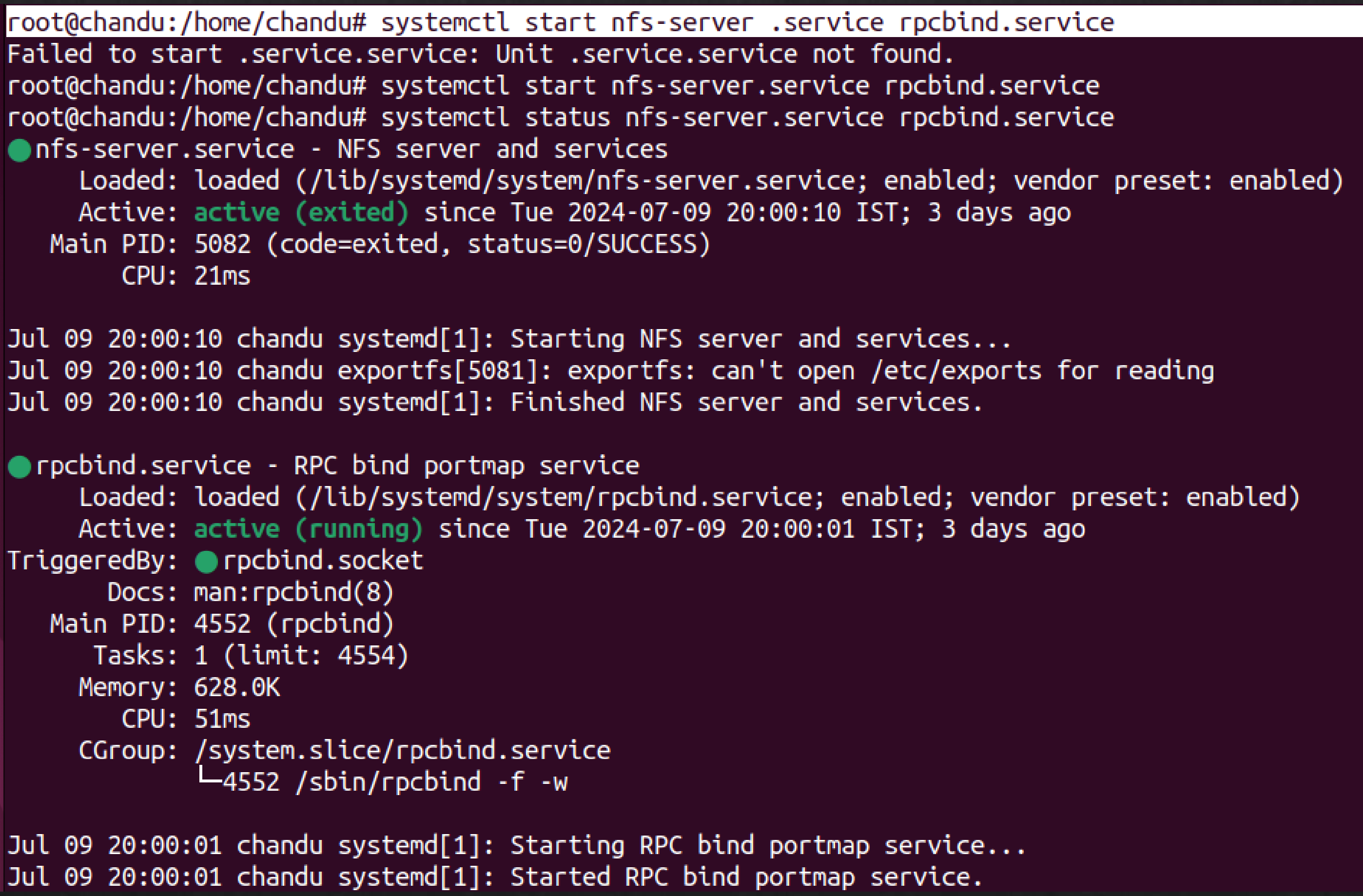
If any service was disable or stoped then we can start them and enable them. And check the status with

1. <<< sudo systemctl status nfs-server.service >>>
2. <<< sudo systemctl status rpcbind.service >>>
3. <<< sudo systemctl status rpc-statd.service >>>
4. <<< sudo systemctl status nfs-idmapd.service >>>



```
┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl enable nfs-server.service
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service → /lib/systemd/system/nfs-server.service.

┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl status nfs-server.service
● nfs-server.service - NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; preset: disabled)
     Active: active (exited) since Tue 2024-07-09 20:01:53 IST; 3 days ago
   Main PID: 12510 (code=exited, status=0/SUCCESS)
        CPU: 9ms

Jul 09 20:01:53 localhost.localdomain systemd[1]: Starting nfs-server.service - NFS server and services...
Jul 09 20:01:53 localhost.localdomain systemd[1]: Finished nfs-server.service - NFS server and services.

┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl start nfs-idmapd.service

┌──(root㉿localhost)-[/home/kali/Desktop]
└─# systemctl status nfs-idmapd.service
● nfs-idmapd.service - NFSv4 ID-name mapping service
     Loaded: loaded (/lib/systemd/system/nfs-idmapd.service; static)
     Active: active (running) since Tue 2024-07-09 20:01:53 IST; 3 days ago
   Main PID: 12493 (rpc.idmapd)
      Tasks: 1 (limit: 4556)
     Memory: 804.0K
        CPU: 12ms
     CGroup: /system.slice/nfs-idmapd.service
             └─12493 /usr/sbin/rpc.idmapd

Jul 09 20:01:53 localhost.localdomain systemd[1]: Starting nfs-idmapd.service - NFSv4 ID-name mapping service...
Jul 09 20:01:53 localhost.localdomain rpc.idmapd[12493]: Setting log level to 0
Jul 09 20:01:53 localhost.localdomain systemd[1]: Started nfs-idmapd.service - NFSv4 ID-name mapping service.
```

Client :-

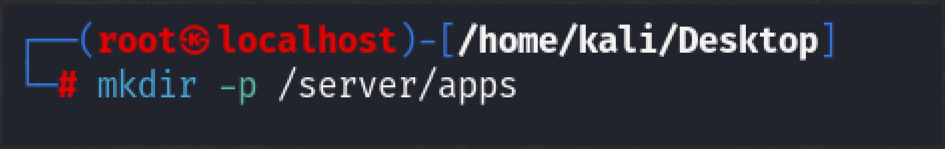


```
root@chandu:/home/chandu# systemctl start nfs-server .service rpcbind.service
Failed to start .service.service: Unit .service.service not found.
root@chandu:/home/chandu# systemctl start nfs-server.service rpcbind.service
root@chandu:/home/chandu# systemctl status nfs-server.service rpcbind.service
● nfs-server.service - NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor preset: enabled)
     Active: active (exited) since Tue 2024-07-09 20:00:10 IST; 3 days ago
   Main PID: 5082 (code=exited, status=0/SUCCESS)
        CPU: 21ms

Jul 09 20:00:10 chandu systemd[1]: Starting NFS server and services...
Jul 09 20:00:10 chandu exportfs[5081]: exportfs: can't open /etc/exports for reading
Jul 09 20:00:10 chandu systemd[1]: Finished NFS server and services.

● rpcbind.service - RPC bind portmap service
     Loaded: loaded (/lib/systemd/system/rpcbind.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2024-07-09 20:00:01 IST; 3 days ago
TriggeredBy: ● rpcbind.socket
       Docs: man:rpcbind(8)
   Main PID: 4552 (rpcbind)
      Tasks: 1 (limit: 4554)
     Memory: 628.0K
        CPU: 51ms
     CGroup: /system.slice/rpcbind.service
             └─4552 /sbin/rpcbind -f -w

Jul 09 20:00:01 chandu systemd[1]: Starting RPC bind portmap service...
Jul 09 20:00:01 chandu systemd[1]: Started RPC bind portmap service.
```

Step 3 :-

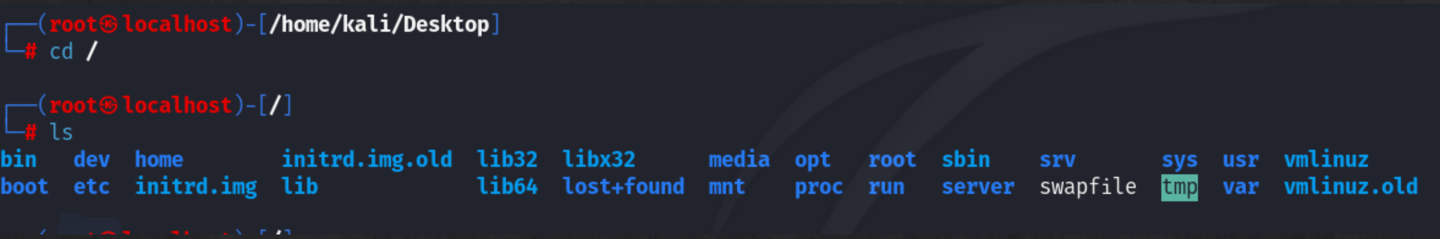
Server :-

Now we have to create an directory in server side with the root access so we can make the directory in root terminal using <<< mkdir –p /server/apps >>> command.



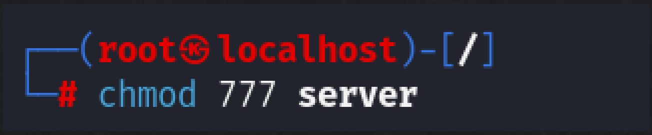
```
┌──(root㉿localhost)-[/home/kali/Desktop]
└─# mkdir -p /server/apps
```

So now we can acces those directories , for listing the directories of root files by using <<< cd / >>> (this command helps to chande the current directoryto the root directory) the simply use <<< ls >>> for listing the existing directory.



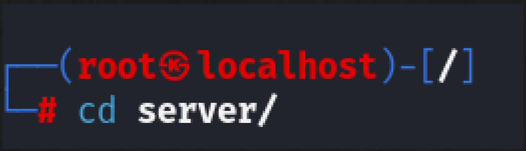
```
┌──(root㉿localhost)-[/home/kali/Desktop]
└─# cd /

┌──(root㉿localhost)-[/]
└─# ls
bin   dev  home        initrd.img.old  lib32  libx32      media  opt   root  sbin    srv       sys  usr  vmlinuz
boot  etc  initrd.img  lib             lib64  lost+found  mnt    proc  run   server  swapfile  tmp  var  vmlinuz.old
```

- Now we have to give the permission to server and apps directory.
- By using <<< chmod777 server >>>(here chmod777 command is used to set full read, write and execute permission on it .)



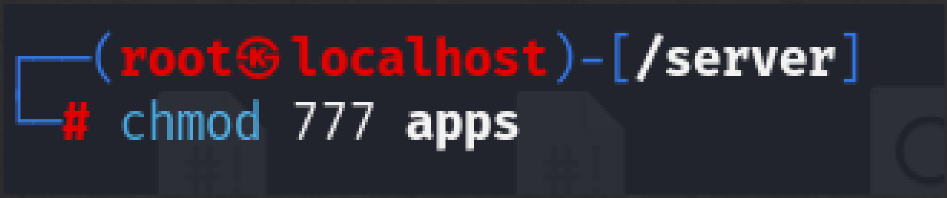
```
┌──(root㉿localhost)-[/]
└─# chmod 777 server
```

Then we have to give permission to apps directory by changing the current directory to server directory by using <<< cd server/ >>> command.



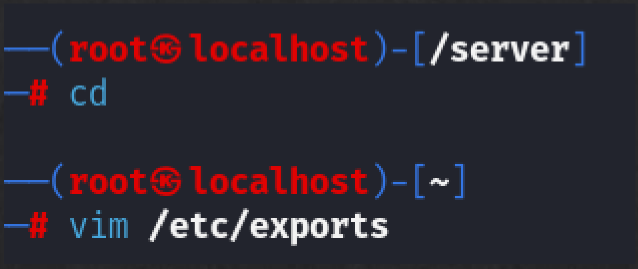
```
┌──(root㉿localhost)-[/]
└─# cd server/
```

Then give the permission to apps directory by <<< chmod 777 apps >>> command.



```
┌──(root㉿localhost)-[/server]
└─# chmod 777 apps
```

- Afte giving all the permissions then we have to modify the /etc/exports file and also add the new shared files.
- Then we have to make an new rule and also have to define the new directory in /etc/exports. By using <<< vim /etc/exports >>> command.
- Where The /etc/exports file plays a crucial role in configuring Network File System (NFS) exports on a Linux server.



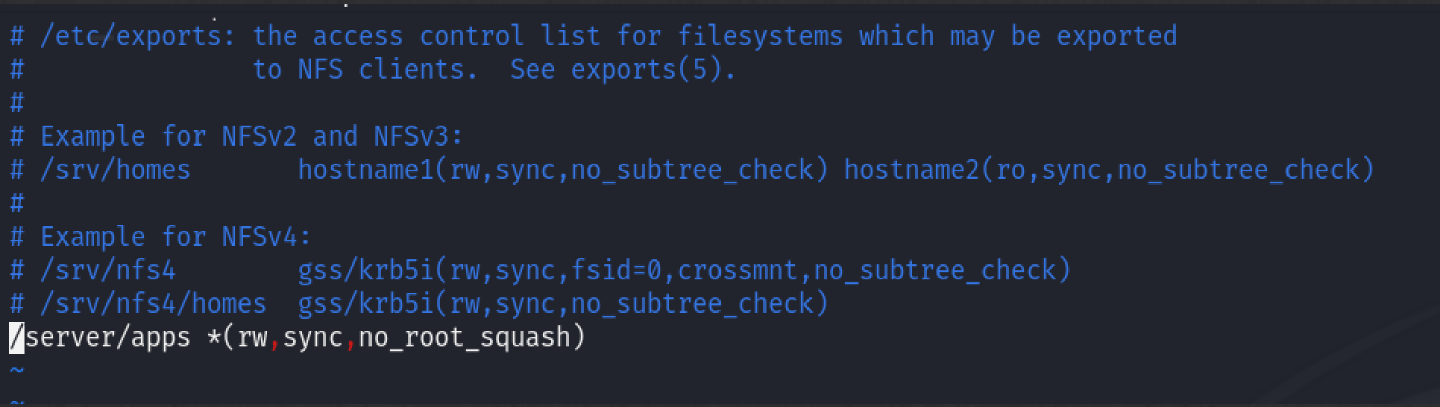
```
──(root㉿localhost)-[/server]
─# cd

──(root㉿localhost)-[~]
─# vim /etc/exports
```

After running the above command then there we have to define the new rule and new directory in vim editor with given command<<< /server/apps *(rw,sync,no_root_squash) >>>

Where *(rw,sync,no_root_squash):

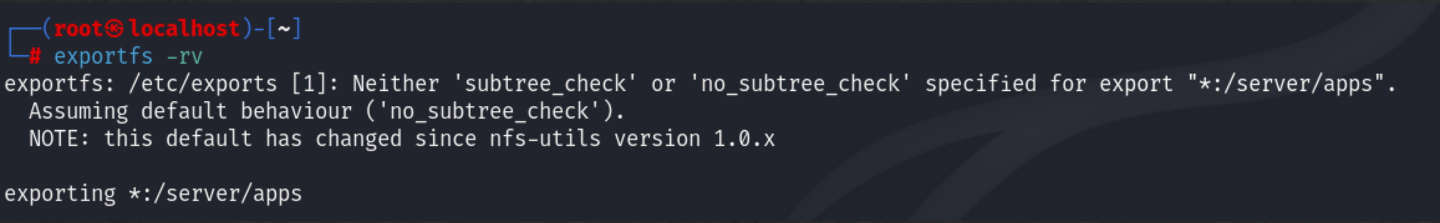
- *: This wildcard allows any client to access the exported directory.
- rw: Grants read and write permissions to clients.
- no_root_squash: Ensures that the root user on the client machine is treated as the root user on the NFS server (i.e., root squashing is disabled).
- sync: Data is written synchronously to the server’s disk.

These options allow clients to read and write data to /server/apps without any restrictions.



```
# /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)
/server/apps *(rw,sync,no_root_squash)
~
```

With the help of <<< exportfs -rv >>> command enables NFS server to maintain the table of local file system accessable to NFS client.

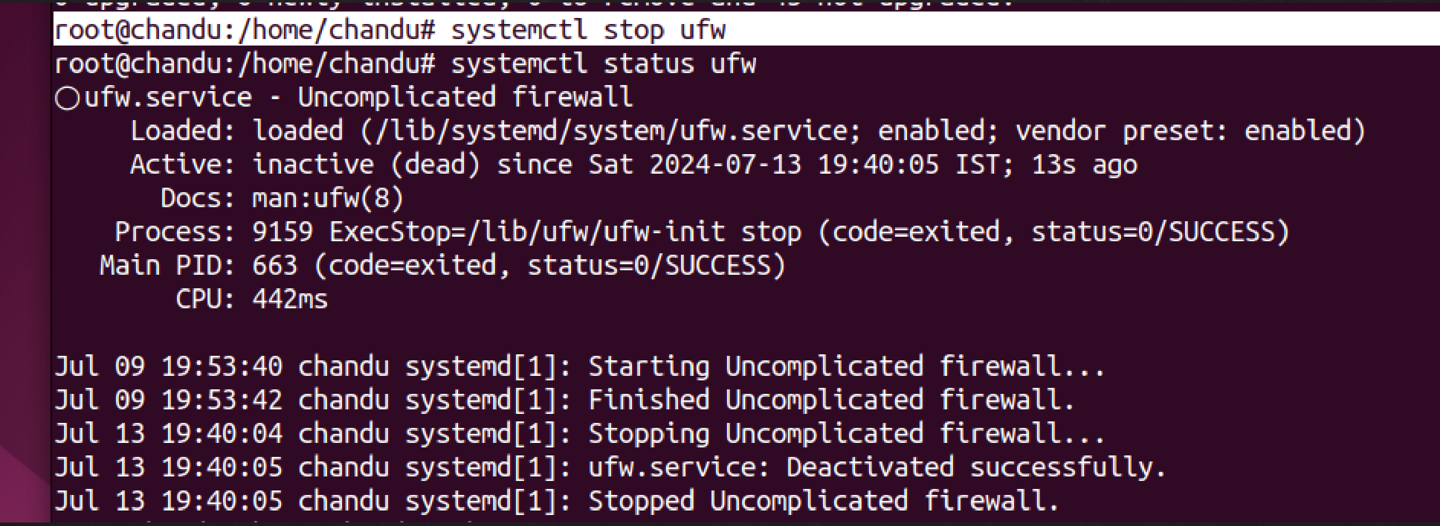


```
┌──(root㉿localhost)-[~]
└─# exportfs -rv
exportfs: /etc/exports [1]: Neither 'subtree_check' or 'no_subtree_check' specified for export "*:/server/apps".
  Assuming default behaviour ('no_subtree_check').
  NOTE: this default has changed since nfs-utils version 1.0.x

exporting *:/server/apps
```

Step 4 :-

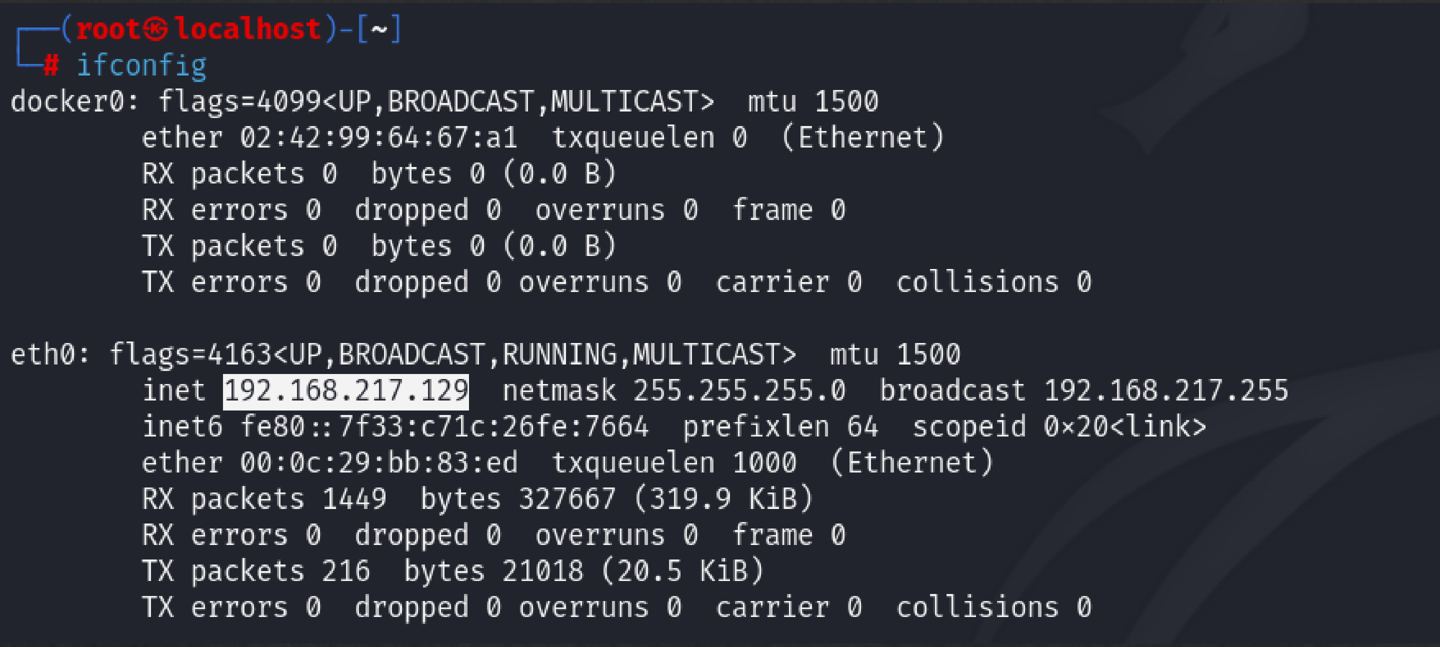
Client :-

From the client side we have to turn off the firewall settings by using <<< systemctl stop ufw >>> command



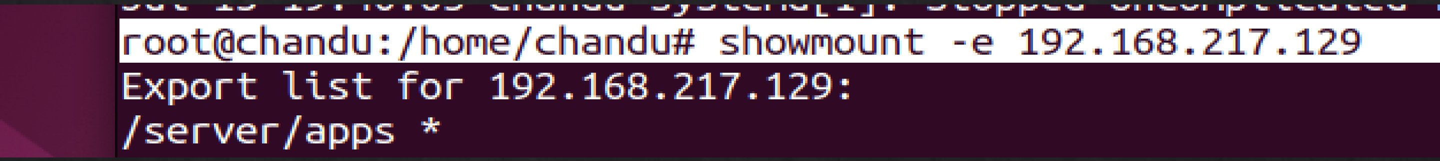
```
root@chandu:/home/chandu# systemctl stop ufw
root@chandu:/home/chandu# systemctl status ufw
○ufw.service - Uncomplicated firewall
     Loaded: loaded (/lib/systemd/system/ufw.service; enabled; vendor preset: enabled)
     Active: inactive (dead) since Sat 2024-07-13 19:40:05 IST; 13s ago
       Docs: man:ufw(8)
    Process: 9159 ExecStop=/lib/ufw/ufw-init stop (code=exited, status=0/SUCCESS)
   Main PID: 663 (code=exited, status=0/SUCCESS)
        CPU: 442ms

Jul 09 19:53:40 chandu systemd[1]: Starting Uncomplicated firewall...
Jul 09 19:53:42 chandu systemd[1]: Finished Uncomplicated firewall.
Jul 13 19:40:04 chandu systemd[1]: Stopping Uncomplicated firewall...
Jul 13 19:40:05 chandu systemd[1]: ufw.service: Deactivated successfully.
Jul 13 19:40:05 chandu systemd[1]: Stopped Uncomplicated firewall.
```

- Now we have to take the server IP address by using <<< ifconfig >>> command.



```
┌──(root㉿localhost)-[~]
└─# ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 02:42:99:64:67:a1  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.217.129  netmask 255.255.255.0  broadcast 192.168.217.255
        inet6 fe80::7f33:c71c:26fe:7664  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:bb:83:ed  txqueuelen 1000  (Ethernet)
        RX packets 1449  bytes 327667 (319.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 216  bytes 21018 (20.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

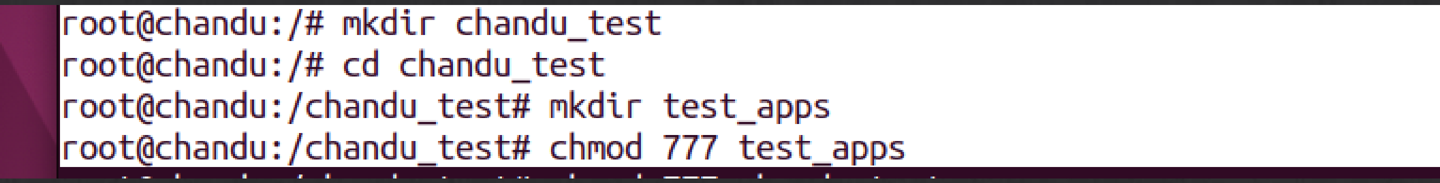
- Then , by using showmount command we can check the mount information on an NFS server . By using <<< showmount –e 129.168.***.*** >>>



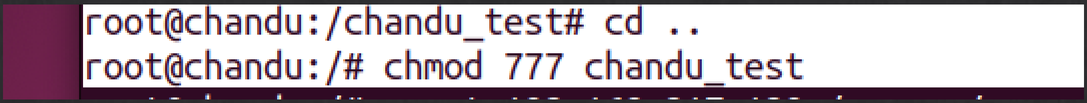
```
root@chandu:/home/chandu# showmount -e 192.168.217.129
Export list for 192.168.217.129:
/server/apps *
```

Now we have to create directory in client's side where the receved date will be stored and also give the permission by following commands.

1. <<< mkdir chandu_txt >>>
2. <<< cd chandu_txt >>>
3. <<< mkdir test_apps >>>
4. <<< chmod 777 test_apps>>>
5. <<< cd ..>>>
6. <<< chmod 777 chandu_txt>>>

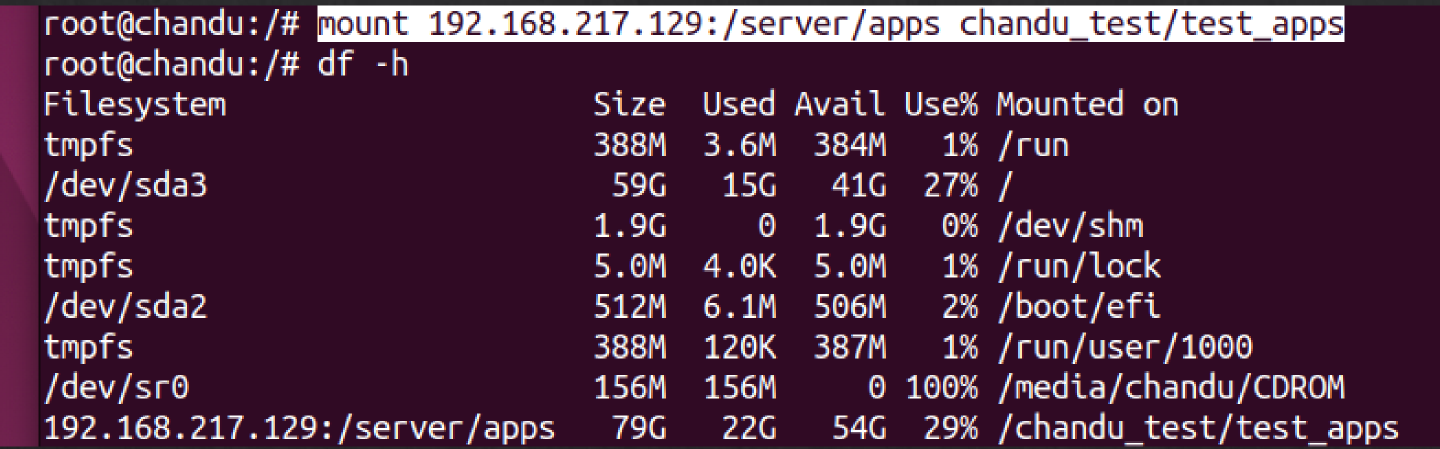


```
root@chandu:/# mkdir chandu_test
root@chandu:/# cd chandu_test
root@chandu:/chandu_test# mkdir test_apps
root@chandu:/chandu_test# chmod 777 test_apps
```



```
root@chandu:/chandu_test# cd ..
root@chandu:/# chmod 777 chandu_test
```

By using <<< mount (ip):/server/apps chandu_txt/test_apps >>> Command we can mount the client side directory to the server side directory with the given server IP.



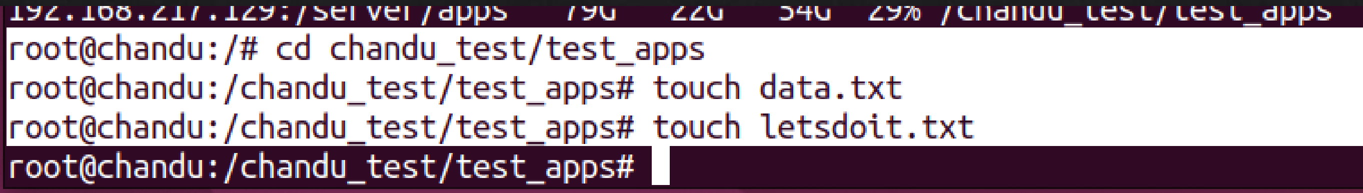
```
root@chandu:/# mount 192.168.217.129:/server/apps chandu_test/test_apps
root@chandu:/# df -h
Filesystem                    Size  Used Avail Use% Mounted on
tmpfs                         388M  3.6M  384M   1% /run
/dev/sda3                      59G   15G   41G  27% /
tmpfs                         1.9G     0  1.9G   0% /dev/shm
tmpfs                         5.0M  4.0K  5.0M   1% /run/lock
/dev/sda2                     512M  6.1M  506M   2% /boot/efi
tmpfs                         388M  120K  387M   1% /run/user/1000
/dev/sr0                      156M  156M     0 100% /media/chandu/CDROM
192.168.217.129:/server/apps   79G   22G   54G  29% /chandu_test/test_apps
```

Step 5:-

- Here , most of the work was done now we can simply use the services .
- First we have to change the directory and go to the mounted directory which was chandu_test/test_apps by using

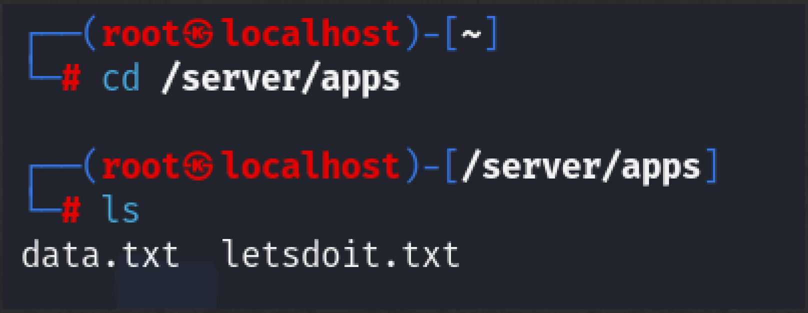
<<< cd chandu_test/test_apps >>> command.

Now I have made some files in the mounted directory named as <<< touch data.txt letsdoit.txt >>>



```
root@chandu:/# cd chandu_test/test_apps
root@chandu:/chandu_test/test_apps# touch data.txt
root@chandu:/chandu_test/test_apps# touch letsdoit.txt
root@chandu:/chandu_test/test_apps#
```

After the client will make the files then we can simply able to access the files in the mounted /server/apps directory and with the help of <<< ls >>> command we can access those files which was created in the client system .



```
┌──(root㉿localhost)-[~]
└─# cd /server/apps

┌──(root㉿localhost)-[/server/apps]
└─# ls
data.txt  letsdoit.txt
```

Here I have made this step-by-step guide which will help you to perform to set up the NFS server in your linux system.

Thank you