

# Лабораторная работа №13

## Настройка NFS

Сахно Никита НФИбд-02-23

### Содержание

|     |   |    |
|-----|---|----|
| 1   | Цель работы .....   | 1  |
| 2   | Задание .....   | 1  |
| 3   | Выполнение лабораторной работы.....   | 2  |
| 3.1 | Настройка сервера NFSv4.....  | 2  |
| 3.2 | Монтирование NFS на клиенте .....   | 4  |
| 3.3 | Подключение каталогов к дереву NFS.....                                     | 6  |
| 3.4 | Подключение каталогов для работы пользователей .....                        | 7  |
| 3.5 | Внесение изменений в настройки внутреннего окружения виртуальных машин<br>9 |    |
| 4   | Выводы.....   | 10 |

### 1 Цель работы

Приобрести навыки настройки сервера NFS для удалённого доступа к ресурсам.

### 2 Задание

1. Установить и настроить сервер NFSv4.
2. Подмонтировать удалённый ресурс на клиенте.
3. Подключить каталог с контентом веб-сервера к дереву NFS.
4. Подключить каталог для удалённой работы вашего пользователя к дереву NFS.
5. Написать скрипты для Vagrant, фиксирующие действия по установке и настройке сервера NFSv4 во внутреннем окружении виртуальных машин server и client. Соответствующим образом внести изменения в Vagrantfile.

## 3 Выполнение лабораторной работы

### 3.1 Настройка сервера NFSv4

На сервере установим необходимое программное обеспечение: `dnf -y install nfs-utils`

```
[dmbelicheva@server.dmbelicheva.net ~]$ sudo -i
[sudo] password for dmbelicheva:
[root@server.dmbelicheva.net ~]# dnf -y install nfs-utils
Extra Packages for Enterprise Linux 9 - x86_64 22 kB/s | 36 kB 00:01
Extra Packages for Enterprise Linux 9 - x86_64 5.7 MB/s | 20 MB 00:03
Rocky Linux 9 - BaseOS 2.3 kB/s | 4.1 kB 00:01
Rocky Linux 9 - BaseOS 2.2 MB/s | 2.2 MB 00:01
Rocky Linux 9 - AppStream 12 kB/s | 4.5 kB 00:00
Rocky Linux 9 - AppStream 6.5 MB/s | 7.4 MB 00:01
Rocky Linux 9 - Extras 8.4 kB/s | 2.9 kB 00:00
Dependencies resolved.
=====
Package Architecture Version Repository Size
=====
Installing:
nfs-utils x86_64 1:2.5.4-20.el9 baseos 425 k
Upgrading:
libipa_hbac x86_64 2.9.1-4.el9_3.1 baseos 38 k
libldb x86_64 2.7.2-2.el9 baseos 180 k
libsmclient x86_64 4.18.6-101.el9_3 baseos 74 k
libsmbmap x86_64 2.9.1-4.el9_3.1 baseos 93 k
```

#### Установка пакетов

На сервере создадим каталог, который предполагается сделать доступным всем пользователям сети (корень дерева NFS): `mkdir -p /srv/nfs`

В файле `/etc/exports` пропишем подключаемый через NFS общий каталог с доступом только на чтение: `/srv/nfs *(ro)`



```
GNU nano 5.6.1
/srv/nfs *(ro)
```

#### Редактирование файла

Для общего каталога зададим контекст безопасности NFS: `semanage fcontext -a -t nfs_t "/srv/nfs(/.*)?"`

Применим изменённую настройку SELinux к файловой системе: `restorecon -vR /srv/nfs`

Запустим сервер NFS:

```
systemctl start nfs-server.service
systemctl enable nfs-server.service
```

Настроим межсетевой экран для работы сервера NFS:

```
firewall-cmd --add-service=nfs
firewall-cmd --add-service=nfs --permanent
firewall-cmd --reload
```

```
on text -a -t nfs_t "/srv/nfs(/.*)?"
-vR /srv/nfs
var t:s0 to unconfined_u:object_r:nfs_t:s0
start nfs-server.service
enable nfs-server.service
.target.wants/nfs-server.service -> /usr/lib/systemd/system/nfs-server.s
d --add-service=nfs
d --add-service=nfs --permanent
d --reload
```

## Настройка межсетевого экрана

На клиенте установим необходимое для работы NFS программное обеспечение: `dnf -y install nfs-utils`

```
Extra Packages for Enterprise Linux 9 - x86_64 17 kB/s |
Extra Packages for Enterprise Linux 9 - x86_64 4.5 MB/s |
Rocky Linux 9 - BaseOS 1.4 kB/s |
Rocky Linux 9 - BaseOS 3.0 MB/s |
Rocky Linux 9 - AppStream 10 kB/s |
Rocky Linux 9 - AppStream 3.4 MB/s |
Rocky Linux 9 - Extras 7.8 kB/s |
Dependencies resolved.
=====
Package Architecture Version Repository
=====
Installing:
nfs-utils x86_64 1:2.5.4-20.el9 baseos
Upgrading:
libnina_bbac x86_64 2.9.1-4.el9_3.1 baseos
```

## Установка пакетов

На клиенте попробуем посмотреть имеющиеся подмонтированные удалённые ресурсы (вместо user укажите свой логин):

```
showmount -e server.nvsakhno.net
```

```
clnt_create: RPC: Unable to receive
```

## Просмотр подмонтированных удалённых ресурсов

Попробуем на сервере остановить сервис межсетевого экрана: `systemctl stop firewalld.service`

Затем на клиенте вновь попробуем подключиться к удалённо смонтированному ресурсу: `showmount -e server.nvsakhno.net`

На сервере запустим сервис межсетевого экрана `systemctl start firewalld`

На сервере посмотрим, какие службы задействованы при удалённом монтировании:

```
lsof | grep TCP
lsof | grep UDP
```

```

Output information may be incomplete.
systemd 1 root 92u IPv4 38904 0t0 TCP *:sunrpc (LISTEN)
systemd 1 root 94u IPv6 38922 0t0 TCP *:sunrpc (LISTEN)
cupsd 861 root 6u IPv6 19573 0t0 TCP localhost:ipp (LISTEN)
cupsd 861 root 7u IPv4 19574 0t0 TCP localhost:ipp (LISTEN)
sshd 871 root 3u IPv4 19727 0t0 TCP *:down (LISTEN)
sshd 871 root 4u IPv6 19734 0t0 TCP *:down (LISTEN)
sshd 871 root 5u IPv4 19738 0t0 TCP *:ssh (LISTEN)

```

*Задействованные службы при удаленном монтировании по протоколу TCP*

```

systemd 1 root 93u IPv4 38913 0t0 UDP *:sunrpc
systemd 1 root 95u IPv6 38931 0t0 UDP *:sunrpc
avahi-daemon 563 avahi 12u IPv4 18794 0t0 UDP *:mdns
avahi-daemon 563 avahi 13u IPv6 18795 0t0 UDP *:mdns
avahi-daemon 563 avahi 14u IPv4 18796 0t0 UDP *:34106
avahi-daemon 563 avahi 15u IPv6 18797 0t0 UDP *:34135
chronyd 590 chrony 5u IPv4 18692 0t0 UDP localhost:323
chronyd 590 chrony 6u IPv6 18693 0t0 UDP localhost:323
chronyd 590 chrony 7u IPv4 18694 0t0 UDP *:ntp
named 901 named 6u IPv4 21448 0t0 UDP server.dmbelicheva.net:domain

```

*Задействованные службы при удаленном монтировании по протоколу UDP*

Добавим службы rpc-bind и mountd в настройки межсетевого экрана на сервере:

```

firewall-cmd --get-services
firewall-cmd --add-service=mountd --add-service=rpc-bind
firewall-cmd --add-service=mountd --add-service=rpc-bind --permanent
firewall-cmd --reload

```

```

bacula-client bb bgp bitcoin bitcoin-rpc bitcoin-testnet bitcoin-testnet-rpc bittorrent-lsd ceph ceph-mon cfengine cl
eckmk-agent cockpit collectd condor-collector cratedb ctdb dhcp dhcpv6 dhcpv6-client distcc dns dns-over-tls docker-r
egistry docker-swarm dropbox-lansync elasticsearch etcd-client etcd-server finger foreman foreman-proxy freeipa-4 fr
eipa-lldap freeipa-ldaps freeipa-replication freeipa-trust ftp galera ganglia-client ganglia-master git gpsd grafana g
re high-availability http http3 https ident imap imaps ipfs ipp ipp-client ipsec irc ircs iscsi-target isns jellyfin
jenkins kadmin kdeconnect kerberos kibana klogind kpasswd kprop kshell kube-api kube-apiserver kube-control-plane kube
-control-plane-secure kube-controller-manager kube-controller-manager-secure kube-nodeport-services kube-scheduler ku
be-scheduler-secure kube-worker kubelet kubelet-readonly kubelet-worker lldap ldaps libvirt libvirt-tls lightning-net
ork llmnr llmnr-tcp llmnr-udp managiesieve matrix mdns memcached minidlna mongodb mosh mountd mqtt mqtt-tls ms-wbt msscl
l murmur mysql nbd netbios-ns netdata-dashboards nfs nfs3 nmap nmap-0183 nrpe ntp nut openvpn ovirt-imageio ovirt-storagec
nsole ovirt-vmconsole plex pmcd pmproxy pmwebapi pmwebapis pop3 pop3s postgresql privoxy prometheus prometheus-node-e
xporter proxy-dhcp ps3netmgr ptp pulseaudio puppetmaster quassel radius rdp redis redis-sentinel rpc-bind rquotad rsync
rsyncd rtsp salt-master samba samba-client samba-dc sane sip sipsec sip smtp smtp-submission smtps snmp snmptls snmptl
s-trap snmptrap spideroak-lansync spotify-sync squid sssd ssh ssh-custom steam-streaming svdrp svn syncthing syncthr
g-gui synergy syslog syslog-tls telnet tentacle tftp tile38 tinc tor-socks transmission-client upnp-client vdsms vnc-s
erver wdem-http wdem-https wireguard ws-discovery ws-discovery-client ws-discovery-tcp ws-discovery-udp wsman wsmans
xdmcp xmpp-bosh xmpp-client xmpp-local xmpp-server zabbix-agent zabbix-server zerotier

```

*Настройка межсетевого экрана на сервере*

На клиенте проверим подключение удалённого ресурса: `showmount -e server.nvsakhno.net`

### 3.2 Монтирование NFS на клиенте

На клиенте создадим каталог, в который будет монтироваться удалённый ресурс, и подмонтируем дерево NFS:

```

mkdir -p /mnt/nfs
mount server.dmbelicheva.net:/srv/nfs /mnt/nfs

```

Проверим, что общий ресурс NFS подключён правильно: `mount`

```

proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime,seclabel)
devtmpfs on /dev type devtmpfs (rw,nosuid,nodev,seclabel,size=4096k,nr_inodes=114507,mode=755,inode64)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,seclabel,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,seclabel,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,seclabel,size=194916k,nr_inodes=819200,mode=755,inode64)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,seclabel,nsdelegate,memory_recursiveprot)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime,seclabel)
bpf on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)
/dev/sdal on / type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,noquota)
selinuxfs on /sys/fs/selinux type selinuxfs (rw,nosuid,noexec,relatime)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=29,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,inode=17475)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel,pagesize=2M)
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime,seclabel)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime,seclabel)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime,seclabel)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
none on /run/credentials/systemd-sysctl.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
none on /run/credentials/systemd-tmpfiles-setup-dev.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
none on /run/credentials/systemd-tmpfiles-setup.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
vagrant on /vagrant type vboxsf (rw,nodev,relatime,iocharset=utf8,uid=1000,gid=1000)
vagrant on /vagrant type vboxsf (rw,nodev,relatime,iocharset=utf8,uid=1000,gid=1000,_netdev)
tmpfs on /run/user/1001 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=97456k,nr_inodes=24364,mode=700,uid=1000,gid=1001,inode64)
gvfsd-fuse on /run/user/1001/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,relatime,user_id=1001,group_id=1001)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
server.dmbelicheva.net:/srv/nfs on /mnt/nfs type nfs4 (rw,relatime,vers=4.2,rsize=131072,wsz=131072,namlen=255,

```

## Монтирование NFS на клиенте

На клиенте в конце файла `/etc/fstab` добавим следующую запись:

```
server.nvsakhno.net:/srv/nfs /mnt/nfs nfs _netdev 0 0
```

```

#
UUID=d648795f-bd46-4875-b95d-2593e1087e22 / xfs
/swapfile none swap defaults 0 0
#VAGRANT-BEGIN
# The contents below are automatically generated by Vagrant. Do not modify
vagrant /vagrant vboxsf uid=1000,gid=1000,_netdev 0 0
#VAGRANT-END

```

## Редактирование файла

На клиенте проверим наличие автоматического монтирования удалённых ресурсов при запуске операционной системы: `systemctl status remote-fs.target`

```

● remote-fs.target - Remote File Systems
   Loaded: loaded (/usr/lib/systemd/system/remote-fs.target; enabled; preset: enabled)
   Active: active since Sat 2023-12-23 01:51:16 MSK; 26min ago
     Until: Sat 2023-12-23 01:51:16 MSK; 26min ago
    Docs: man:systemd.special(7)

```

## Проверка наличия автоматического монтирования удалённых ресурсов

Перезапустим клиент и убедимся, что удалённый ресурс подключается автоматически.

```
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime,seclabel)
devtmpfs on /dev type devtmpfs (rw,nosuid,seclabel,size=4096k,nr_inodes=114507,mode=755,inode64)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,seclabel,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,seclabel,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,seclabel,size=194916k,nr_inodes=819200,mode=755,inode64)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,seclabel,nsdelegate,memory_recursiveprot)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime,seclabel)
bpf on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)
/dev/sdal on / type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,noquota)
selinuxfs on /sys/fs/selinux type selinuxfs (rw,nosuid,noexec,relatime)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=29,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,inode=17478)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel,pagesize=2M)
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime,seclabel)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime,seclabel)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime,seclabel)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
none on /run/credentials/systemd-sysctl.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
none on /run/credentials/systemd-tmpfiles-setup-dev.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
none on /run/credentials/systemd-tmpfiles-setup.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
vagrant on /vagrant type vboxsf (rw,nodev,relatime,ioccharset=utf8,uid=1000,gid=1000)
server.dmbelicheva.net:/srv/nfs on /mnt/nfs type nfs4 (rw,relatime,vers=4.2,rsize=131072,wsz=131072,namlen=255,proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr=192.168.1.30,local_lock=none,addr=192.168.1.1,_netdev)
```

*Проверка*

### 3.3 Подключение каталогов к дереву NFS

На сервере создадим общий каталог, в который затем будет подмонтирован каталог с контентом веб-сервера: `mkdir -p /srv/nfs/www`

Подмонтируем каталог web-сервера: `mount -o bind /var/www/ /srv/nfs/www/`

На сервере проверим, что отображается в каталоге `/srv/nfs`.

```
mount -o bind /var/www/ /srv/nfs/www/
cd /srv/nfs
ls
```

*Содержимое каталога*

На клиенте посмотрим, что отображается в каталоге `/mnt/nfs`.

```
cd /mnt/nfs
$ ls
```

*Содержимое каталога*

На сервере в файле `/etc/exports` добавим экспорт каталога веб-сервера с удалённого ресурса: `/srv/nfs/www 192.168.0.0/16(rw)`

```
GNU nano 5.6.1
/srv/nfs *(ro)
/srv/nfs/www 192.168.0.0/16(rw)
```

*Редактирование файла*

Экспортируем все каталоги, упомянутые в файле /etc/exports: `exportfs -r`

Проверим на клиенте каталог /mnt/nfs.

```
cd /mnt/nfs
ls
```

### *Содержимое каталога*

На сервере в конце файла /etc/fstab добавим следующую запись: `/var/www /srv/nfs/www none bind 0 0`

```
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
UUID=d648795f-bd46-4875-b95d-2593e1087e22 / xfs default
/swapfile none swap defaults 0 0
#VAGRANT-BEGIN
# The contents below are automatically generated by Vagrant. Do not modify.
vagrant /vagrant vboxsf uid=1000,gid=1000,_netdev 0 0
#VAGRANT-END
/var/www /srv/nfs/www none bind 0 0
```

### *Редактирование файла*

Повторно экспортируем каталоги, указанные в файле /etc/exports: `exportfs -r`

На клиенте проверим каталог /mnt/nfs.

## **3.4 Подключение каталогов для работы пользователей**

На сервере под пользователем dmbelicheva в его домашнем каталоге создадим каталог common с полными правами доступа только для этого пользователя, а в нём файл dmbelicheva@server.txt:

```
mkdir -p -m 700 ~/common
cd ~/common
touch nvsakhno@server.txt
```

На сервере создадим общий каталог для работы пользователя nvsakhno по сети: `mkdir -p /srv/nfs/home/user`

Подмонтируем каталог common пользователя nvsakhno в NFS: `mount -o bind /home/user/common /srv/nfs/home/user`

```

-p -m 700 ~/common
common
touch dmbelicheva@server.txt
mkdir -p /srv/nfs/home/dmbelicheva
: Permission denied
sudo -i

/nfs/home/dmbelicheva
/home/user/common /srv/nfs/home/dmbelicheva
ce /home/user/common does not exist.
it systemd still uses
'reload' to reload.

s

```

### *Подключение каталогов для работы пользователей*

Подключим каталог пользователя в файле `/etc/exports`, прописав в нём (вместо user укажите свой логин): `/srv/nfs/home/user 192.168.0.0/16(rw)`

```

root@server:/etc
GNU nano 5.6.1 exports
/srv/nfs *(ro)
/srv/nfs/www 192.168.0.0/16(rw)
/srv/nfs/home/dmbelicheva 192.168.0.0/16(rw)

```

### *Редактирование файла*

Внесем изменения в файл `/etc/fstab` (вместо user укажите свой логин):  
`/home/user/common /srv/nfs/home/user none bind 0 0`

Повторно экспортируем каталоги: `exportfs -r`

На клиенте проверим каталог `/mnt/nfs`.

На клиенте под пользователем user перейдем в каталог `/mnt/nfs/home/user` и попробуем создать в нём файл `user@client.txt` и внести в него какие-либо изменения:

```

cd /mnt/nfs/home/user
touch user@client.txt

```

```

nfs/home/dmbelicheva
$ touch test.txt
system
$ sudo -i

e/dmbelicheva
test.txt
system

$ ls
$

```

### *Создание файла*

Безуспешно.



Попробуем это проделать под пользователем root.

Безуспешно.

На сервере посмотрим, появились ли изменения в каталоге пользователя /home/user/common.

Не появились, все тщетно.

### 3.5 Внесение изменений в настройки внутреннего окружения виртуальных машин

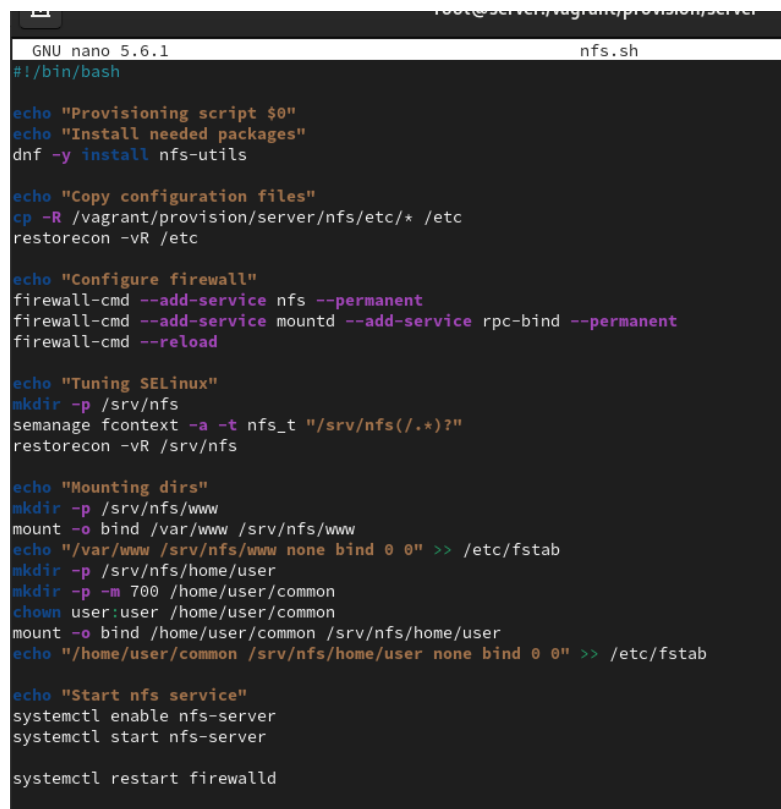
На виртуальной машине server перейдем в каталог для внесения изменений в настройки внутреннего окружения /vagrant/provision/server/, создадим в нём каталог nfs, в который поместим в соответствующие подкаталоги конфигурационные файлы:

```
cd /vagrant/provision/server
mkdir -p /vagrant/provision/server/nfs/etc
cp -R /etc/exports /vagrant/provision/server/nfs/etc/
```

В каталоге /vagrant/provision/server создадим исполняемый файл nfs.sh:

```
cd /vagrant/provision/server
touch nfs.sh
chmod +x nfs.sh
```

Открыв его на редактирование, пропишем в нём следующий скрипт:



```
GNU nano 5.6.1                                nfs.sh
#!/bin/bash

echo "Provisioning script $0"
echo "Install needed packages"
dnf -y install nfs-utils

echo "Copy configuration files"
cp -R /vagrant/provision/server/nfs/etc/* /etc
restorecon -vR /etc

echo "Configure firewall"
firewall-cmd --add-service nfs --permanent
firewall-cmd --add-service mountd --add-service rpc-bind --permanent
firewall-cmd --reload

echo "Tuning SELinux"
mkdir -p /srv/nfs
semanage fcontext -a -t nfs_t "/srv/nfs(/.*)?"
restorecon -vR /srv/nfs

echo "Mounting dirs"
mkdir -p /srv/nfs/www
mount -o bind /var/www /srv/nfs/www
echo "/var/www /srv/nfs/www none bind 0 0" >> /etc/fstab
mkdir -p /srv/nfs/home/user
mkdir -p -m 700 /home/user/common
chown user:user /home/user/common
mount -o bind /home/user/common /srv/nfs/home/user
echo "/home/user/common /srv/nfs/home/user none bind 0 0" >> /etc/fstab

echo "Start nfs service"
systemctl enable nfs-server
systemctl start nfs-server

systemctl restart firewalld
```

*Редактирование файла*

На виртуальной машине client перейдем в каталог для внесения изменений в настройки внутреннего окружения /vagrant/provision/client/: `cd /vagrant/provision/client`

В каталоге /vagrant/provision/client создадим исполняемый файл nfs.sh:

```
cd /vagrant/provision/client
touch nfs.sh
chmod +x nfs.sh
```

Открыв его на редактирование, пропишем в нём следующий скрипт:



```
GNU nano 5.6.1                                nfs.sh
#!/bin/bash

echo "Provisioning script $0"

echo "Install needed packages"
dnf -y install nfs-utils

echo "Mounting dirs"
mkdir -p /mnt/nfs
mount server.user.net:/srv/nfs /mnt/nfs
echo "server.user.net:/srv/nfs /mnt/nfs nfs _netdev 0 0" >> /etc/fstab
restorecon -vR /etc
```

### *Редактирование файла*

Для отработки созданных скриптов во время загрузки виртуальных машин server и client в конфигурационном файле Vagrantfile необходимо добавить в соответствующих разделах конфигураций для сервера и клиента:

```
server.vm.provision "server nfs",
  type: "shell",
  preserve_order: true,
  path: "provision/server/nfs.sh"

client.vm.provision "client nfs",
  type: "shell",
  preserve_order: true,
  path: "provision/client/nfs.sh"
```

## **4 Выводы**

В процессе выполнения данной лабораторной работы я приобрел навыки настройки сервера NFS для удалённого доступа к ресурсам.