

# 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 .....	5
3.3	Подключение каталогов к дереву NFS.....	7
3.4	Подключение каталогов для работы пользователей .....	8
3.5	Внесение изменений в настройки внутреннего окружения виртуальных машин	
	9	
4	Выводы.....	11

## 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`

```
Установка : sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64 7/
Запуск скриптлета: sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64 7/
Проверка : gssproxy-0.8.4-7.el9.x86_64 1/
Проверка : libev-4.33-6.el9.x86_64 2/
Проверка : libnfsidmap-1:2.5.4-38.el9.x86_64 3/
Проверка : libverto-libev-0.3.2-3.el9.x86_64 4/
Проверка : nfs-utils-1:2.5.4-38.el9.x86_64 5/
Проверка : rpcbind-1.2.6-7.el9.x86_64 6/
Проверка : sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64 7/

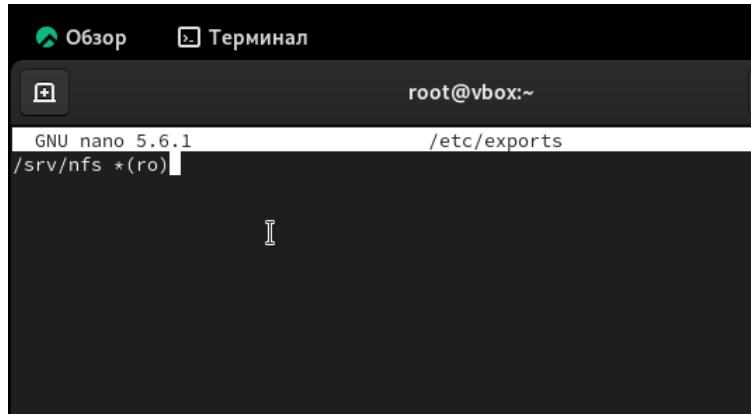
Установлен:
gssproxy-0.8.4-7.el9.x86_64           libev-4.33-6.el9.x86_64
libnfsidmap-1:2.5.4-38.el9.x86_64    libverto-libev-0.3.2-3.el9.x86_64
nfs-utils-1:2.5.4-38.el9.x86_64       rpcbind-1.2.6-7.el9.x86_64
sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64

Выполнено!
```

#### 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)`



#### 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
```

```
[root@vbox ~]# systemctl start nfs-server.service  
[root@vbox ~]# systemctl enable nfs-server.service  
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service →  
/usr/lib/systemd/system/nfs-server.service.  
[root@vbox ~]# firewall-cmd --add-service=nfs  
success  
[root@vbox ~]# firewall-cmd --add-service=nfs --permanent  
success  
[root@vbox ~]# firewall-cmd --reload  
success  
[root@vbox ~]#
```

### *Настройка межсетевого экрана*

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

```
Установка : sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64 7/  
Запуск скриптлета: sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64 7/  
Проверка : gssproxy-0.8.4-7.el9.x86_64 1/  
Проверка : libev-4.33-6.el9.x86_64 2/  
Проверка : libnfsidmap-1:2.5.4-38.el9.x86_64 3/  
Проверка : libverto-libev-0.3.2-3.el9.x86_64 4/  
Проверка : nfs-utils-1:2.5.4-38.el9.x86_64 5/  
Проверка : rpcbind-1.2.6-7.el9.x86_64 6/  
Проверка : sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64 7/  
  
Установлен:  
gssproxy-0.8.4-7.el9.x86_64 libev-4.33-6.el9.x86_64  
libnfsidmap-1:2.5.4-38.el9.x86_64 libverto-libev-0.3.2-3.el9.x86_64  
nfs-utils-1:2.5.4-38.el9.x86_64 rpcbind-1.2.6-7.el9.x86_64  
sssd-nfs-idmap-2.9.7-4.el9_7.1.x86_64  
  
Выполнено!
```

### *Установка пакетов*

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

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

```
[root@vbox ~]# showmount -e server.nvsakhno.net  
clnt_create: RPC: Unknown host  
[root@vbox ~]#
```

### *Просмотр подмонтированных удаленных ресурсов*

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

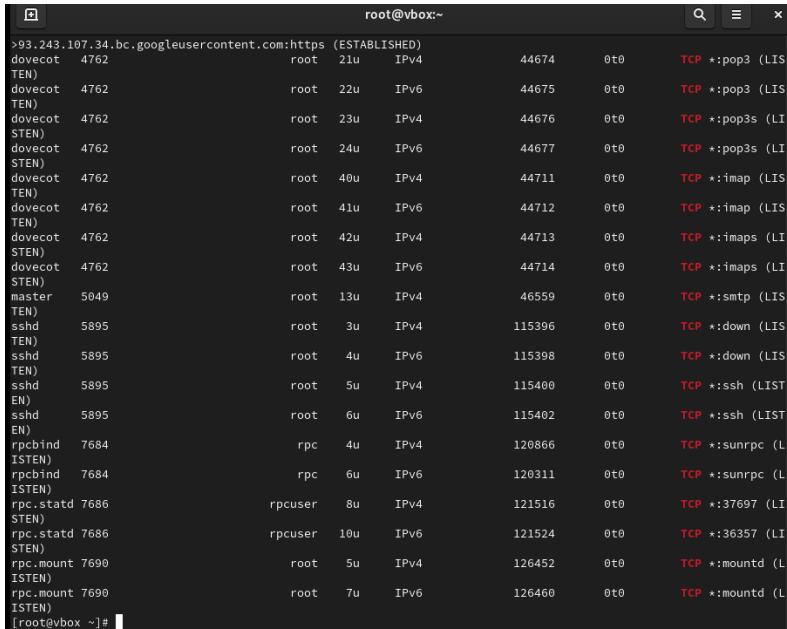
Затем на клиенте вновь попробуем подключиться к удалённо смонтированному ресурсу: `showmount -e server.nvsakhno.net`

На сервере запустим сервис межсетевого экрана `systemctl start firewalld`

На сервере посмотрим, какие службы задействованы при удалённом монтировании:

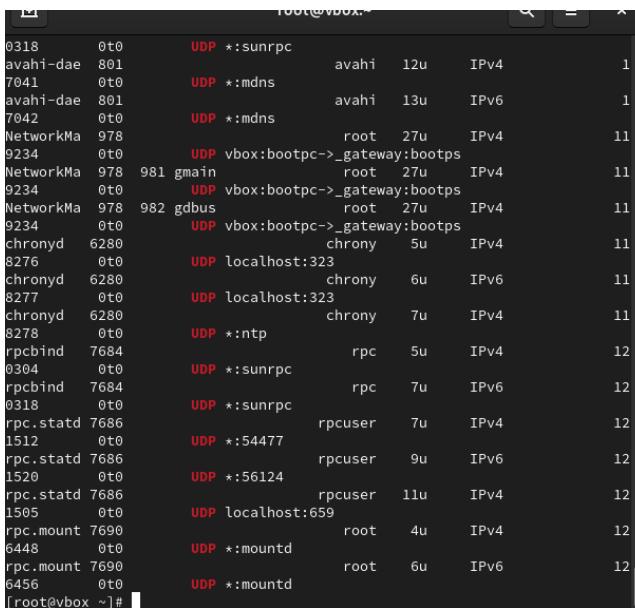
```
lsof | grep TCP
```

```
lsof | grep UDP
```



```
root@vbox:~# lsof | grep TCP
>93.243.107.34.bc.googleusercontent.com:https (ESTABLISHED)
dovecot 4762 root 2lu IPv4 44674 0t0 TCP *:pop3 (LIS
TEN)
dovecot 4762 root 22u IPv6 44675 0t0 TCP *:pop3 (LIS
TEN)
dovecot 4762 root 23u IPv4 44676 0t0 TCP *:pop3s (LI
STEN)
dovecot 4762 root 24u IPv6 44677 0t0 TCP *:pop3s (LI
STEN)
dovecot 4762 root 40u IPv4 44711 0t0 TCP *:imap (LIS
TEN)
dovecot 4762 root 41u IPv6 44712 0t0 TCP *:imap (LIS
TEN)
dovecot 4762 root 42u IPv4 44713 0t0 TCP *:imaps (LI
STEN)
dovecot 4762 root 43u IPv6 44714 0t0 TCP *:imaps (LI
STEN)
master 5849 root 13u IPv4 46559 0t0 TCP *:smtp (LIS
TEN)
sshd 5895 root 3u IPv4 115396 0t0 TCP *:down (LIS
TEN)
sshd 5895 root 4u IPv6 115398 0t0 TCP *:down (LIS
TEN)
sshd 5895 root 5u IPv4 115400 0t0 TCP *:ssh (LIST
EN)
sshd 5895 root 6u IPv6 115402 0t0 TCP *:ssh (LIST
EN)
rpcbind 7684 rpc 4u IPv4 120866 0t0 TCP *:sunrpc (L
ISTEN)
rpcbind 7684 rpc 6u IPv6 120311 0t0 TCP *:sunrpc (L
ISTEN)
rpc.statd 7686 rpcuser 8u IPv4 121516 0t0 TCP *:37697 (LI
STEN)
rpc.statd 7686 rpcuser 10u IPv6 121524 0t0 TCP *:36357 (LI
STEN)
rpc.mount 7690 root 5u IPv4 126452 0t0 TCP *:mountd (L
ISTEN)
rpc.mount 7690 root 7u IPv6 126460 0t0 TCP *:mountd (L
ISTEN)
[root@vbox ~]#
```

*Задействованные службы при удаленном монтировании по протоколу TCP*



```
root@vbox:~# lsof | grep UDP
0318 0t0 UDP *:sunrpc avahi 12u IPv4 1
avahi-dae 801 0t0 UDP *:mdns avahi 13u IPv6 1
7041 0t0 UDP *:mdns avahi-dae 801 0t0 UDP *:mdns
7042 0t0 UDP *:mdns NetworkMa 978 0t0 UDP vbox:bootpc->_gateway:bootps
9234 0t0 UDP vbox:bootpc->_gateway:bootps NetworkMa 978 981 gmain 0t0 UDP vbox:bootpc->_gateway:bootps
9234 0t0 UDP vbox:bootpc->_gateway:bootps NetworkMa 978 982 gibus 0t0 UDP vbox:bootpc->_gateway:bootps
9234 0t0 UDP chrony 6280 0t0 UDP localhost:323
chronyd 6280 0t0 UDP localhost:323 chronyd 6280 0t0 UDP localhost:323
8276 0t0 UDP chrony 6277 0t0 UDP localhost:323
chronyd 6280 0t0 UDP localhost:323 chronyd 6280 0t0 UDP localhost:323
8278 0t0 UDP *:ntp rpcbind 7684 0t0 UDP *:sunrpc
0304 0t0 UDP *:sunrpc rpcbind 7684 0t0 UDP *:sunrpc
0318 0t0 UDP *:sunrpc rpc.statd 7686 0t0 UDP *:54477
1512 0t0 UDP *:54477 rpc.statd 7686 0t0 UDP *:56124
1520 0t0 UDP *:56124 rpc.statd 7686 0t0 UDP localhost:659
1505 0t0 UDP localhost:659 rpc.mount 7690 6448 0t0 UDP *:mountd
6448 0t0 UDP *:mountd rpc.mount 7690 6456 0t0 UDP *:mountd
6456 0t0 UDP *:mountd
[root@vbox ~]#
```

*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
```

```
RH-Satellite-6 RH-Satellite-6-capsule afp amanda-client amanda-k5-client a
mqp amqps apcupsd audit ausweisapp2 bacula-client bareos-director b
areos-filedaemon bareos-storage bb bgp bitcoin bitcoin-rpc bitcoin-testnet
bitcoin-testnet-rpc bittorrent-lsd ceph ceph-exporter ceph-mon cfengine c
heckmk-agent cockpit collectd condor-collector cratedb ctdb dds dds-multic
ast dds-unicast dhcp dhcpcv6 dhcpcv6-client distcc dns dns-over-tls docker-r
egistry docker-swarm dropbox-lansync elasticsearch etcd-client etcd-server
finger foreman foreman-proxy freeipa-4 freeipa-ldap freeipa-ldaps freeipa
-replication freeipa-trust ftp galera ganglia-client ganglia-master git gp
sd grafana gre high-availability http http3 https ident imap imaps ipfs ip
p ipp-client ipsec irc ircs iscsi-target isns jenkins kadmin kdeconnect ke
rberos kibana klogin kpasswd kprop kshell kube-api kube-apiserver kube-con
trol-plane kube-control-plane-secure kube-controller-manager kube-controll
er-manager-secure kube-nodeport-services kube-scheduler kube-scheduler-sec
ure kube-worker kubelet kubelet-readonly kubelet-worker ldap ldaps libvirt
libvirt-tls lightning-network llmnr llmnr-client llmnr-tcp llmnr-udp mana
gesieve matrix mdns memcache minidlna mongodb mosh mountd mqtt mqtt-tls ms
-fwbt mssql murmur mysql nbd nebula netbios-ns netdata-dashboard nfs nfs3 n
ne-0183 nrpe ntp nut opentelemetry openvpn ovirt-imageio ovirt-storagecon
sole ovirt-vmconsole plex pmcd pmproxy pmwebapi pmwebapis pop3 pop3s postg
resql privoxy prometheus prometheus-node-exporter proxy-dhcp ps2link ps3ne
tsrv rtp pulseaudio puppetmaster quassel radius rdp redis redis-sentinel r
pote rpc-bind rquotad rsh rsyncd rtsp salt-master samba samba-client samba
dc sane sip sips slp smtp smtp-submission smtps snmp snmp tls snmptrap
snmptrap spideroak-lansync spotify-sync squid ssdp ssh ssh-custom steam-s
reaming svdrp svn syncthing syncthing-gui syncthing-relay synergy syslog
syslog-tls telnet tentacle tftp tile38 tinc tor-socks transmission-client
upnp-client vdsd vnc-server warpinator wbem-http wbem-https wireguard ws-d
iscovery ws-discovery-client ws-discovery-tcp ws-discovery-udp wsman wsman
s xdmcp xmpp-bosh xmpp-client xmpp-local xmpp-server zabbix-agent zabbix-s
erver zerotier
[root@vbox ~]#
```

### *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.nvsakhno.net:/srv/nfs /mnt/nfs
```

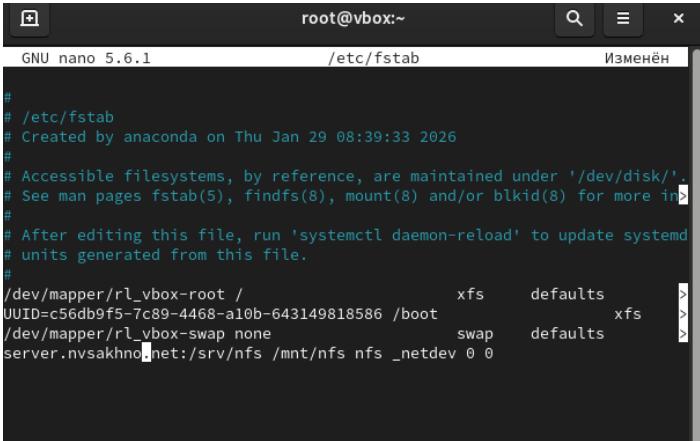
Проверим, что общий ресурс NFS подключен правильно: `mount`

```
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime,seclabel)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
none on /run/credentials/systemd-tmpfiles-setup-dev.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
none on /run/credentials/systemd-sysctl.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
/var/lib/snapd/snaps/octave_306.snap on /var/lib/snapd/snap/octave/306 type squashfs (ro,nodev,relatime,context=system_u:object_r:snappy_snap_t:s0,errors=continue,x-gdu.hide)
/var/lib/snapd/snaps/snapd_25935.snap on /var/lib/snapd/snap/snapd/25935 type squashfs (ro,nodev,relatime,context=system_u:object_r:snappy_snap_t:s0,errors=continue,x-gdu.hide)
/var/lib/snapd/snaps/core18_2979.snap on /var/lib/snapd/snap/core18/2979 type squashfs (ro,nodev,relatime,context=system_u:object_r:snappy_snap_t:s0,errors=continue,x-gdu.hide)
/dev/sdal on /boot type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,noquota)
none on /run/credentials/systemd-tmpfiles-setup.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=174760k,nr_inodes=43690,mode=700,uid=1000,gid=1000,inode64)
gvfsd-fuse on /run/user/1000/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,relatime,user_id=1000,group_id=1000)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,nosuid,nodev,noexec,relatime)
portal on /run/user/1000/doc type fuse.portal (rw,nosuid,nodev,relatime,user_id=1000,group_id=1000)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
nfsd on /proc/fs/nfsd type nfsd (rw,relatime)
root@vbox:~#
```

### Монтиране NFS на клиенте

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

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



```
root@vbox:~# 
root@vbox:~# nano /etc/fstab
GNU nano 5.6.1
/etc/fstab
Изменён

#
# /etc/fstab
# Created by anaconda on Thu Jan 29 08:39:33 2026
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# 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.
#
/dev/mapper/rl_vbox-root /           xfs    defaults      >
UUID=c56dbbf5-7c89-4468-a10b-643149818586 /boot   xfs      >
/dev/mapper/rl_vbox-swap none        swap   defaults      >
server.nvsakhno.net:/srv/nfs /mnt/nfs nfs _netdev 0 0
```

### Редактирование файла

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

```
[root@vbox ~]# systemctl status remote-fs.target
● remote-fs.target - Remote File Systems
  Loaded: loaded (/usr/lib/systemd/system/remote-fs.target; enabled; pre>
  Active: active since Thu 2026-02-12 13:02:27 MSK; 2h 28min ago
    Until: Thu 2026-02-12 13:02:27 MSK; 2h 28min ago
      Docs: man:systemd.special(7)
Lines 1-5/5 (END)
```

*Проверка наличия автоматического монтирования удалённых ресурсов*

Перезапустим клиент и убедимся, что удалённый ресурс подключается автоматически.

### 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`.

```
[root@vbox ~]# mkdir -p /srv/nfs/www
[root@vbox ~]# mount -o bind /var/www/ /srv/nfs/www/
mount: (hint) your fstab has been modified, but systemd still uses
      the old version; use 'systemctl daemon-reload' to reload.
[root@vbox ~]#
```

*Содержимое каталога*

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

```
[root@vbox ~]# cd /srv/nfs
[root@vbox nfs]# ls
www
[root@vbox nfs]#
```

*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)`

```
root@vbox:/mnt/nfs
GNU nano 5.6.1          /etc/fstab   Изменение
#
# /etc/fstab
# Created by anaconda on Thu Jan 29 08:39:33 2026
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more
#
# After editing this file, run 'systemctl daemon-reload' to update system
# units generated from this file.

/dev/mapper/r1_vbox-root /           xfs     defaults
UUID=c56db9f5-7c89-4468-310b-043149818586 /boot   xfs
/dev/mapper/r1_vbox-swap none   swap   defaults
server.nvsakhno.net:/srv/nfs /mnt/nfs nfs _netdev 0 0
/var/www /srv/nfs/www none bind 0 0
```

*Редактирование файла*

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

Проверим на клиенте каталог /mnt/nfs.

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

### 3.4 Подключение каталогов для работы пользователей

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

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

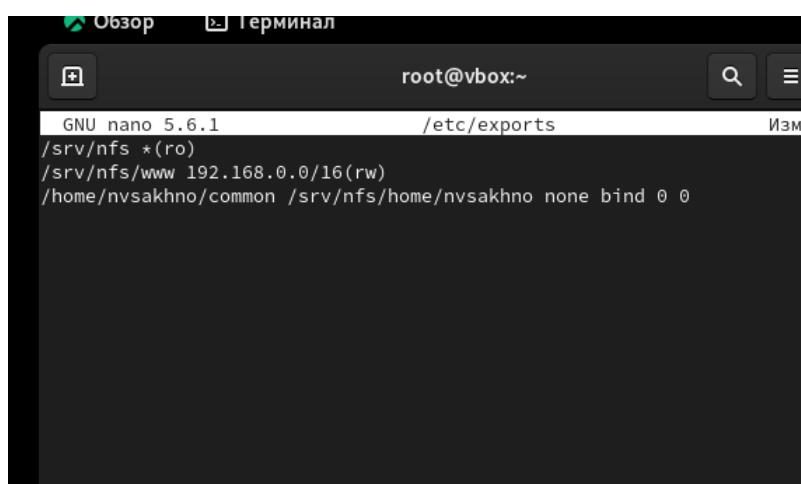
На сервере создадим общий каталог для работы пользователя nvsakhno по сети: `mkdir -p /srv/nfs/home/user`

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

```
[root@vbox ~]# mkdir -p -m 700 ~/common  
[root@vbox ~]# cd ~/common  
[root@vbox common]# touch nvsakhno@server.txt  
[root@vbox common]#
```

#### 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)`



The screenshot shows a terminal window titled "Терминал" (Terminal) with the command "root@vbox:~". The window displays the contents of the "/etc/exports" file in a nano editor. The file contains the following lines:

```
GNU nano 5.6.1          /etc/exports          Изм.  
/srv/nfs *(ro)  
/srv/nfs/www 192.168.0.0/16(rw)  
/home/nvsakhno/common /srv/nfs/home/nvsakhno none bind 0 0
```

#### 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
```

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:

```
root@server:/vagrant/provision/server
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.