

Лабораторная работа №13

Настройка NFS

Лисовская Арина Валерьевна

Содержание

1 Цель работы	4
2 Задание	5
3 Выполнение лабораторной работы	6
4 Выводы	17
5 Ответы на контрольные вопросы	18

Список иллюстраций

3.1 Установка nfs-utils на клиенте	6
3.2 Создание корневого каталога NFS на сервере	6
3.3 Редактирование /etc(exports	7
3.4 Настройка SELinux	7
3.5 Запуск служб и настройка Firewall	7
3.6 Повторная проверка пакетов на клиенте	8
3.7 Ошибка при выполнении showmount	8
3.8 Остановка Firewall на сервере	8
3.9 Успешный вывод showmount	9
3.10 Проверка TCP сокетов	9
3.11 Проверка UDP сокетов	10
3.12 Дополнительная настройка Firewall	10
3.13 Проверка доступа после настройки Firewall	11
3.14 Монтирование ресурса на клиенте	11
3.15 Проверка смонтированных файловых систем	11
3.16 Редактирование /etc/fstab на клиенте	12
3.17 Проверка статуса remote-fs.target	12
3.18 Монтирование каталога www на сервере	12
3.19 Проверка видимости каталога www на клиенте	12
3.20 Экспорт каталога www	13
3.21 Фиксация bind-монтирования в fstab	13
3.22 Создание каталога пользователя	13
3.23 Ошибка монтирования пользовательского каталога	13
3.24 Экспорт домашнего каталога	14
3.25 Проверка ресурсов на клиенте	14
3.26 Проверка прав доступа пользователя	14
3.27 Подготовка каталогов для скриптов	15
3.28 Скрипт настройки сервера nfs.sh	15
3.29 Подготовка скрипта для клиента	15
3.30 Скрипт настройки клиента nfs.sh	16
3.31 Настройка provision для сервера в Vagrantfile	16
3.32 Настройка provision для клиента в Vagrantfile	16

1 Цель работы

Приобретение навыков 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 пользователя к дереву NFS.
5. Написать скрипты для Vagrant, фиксирующие действия по установке и настройке.

3 Выполнение лабораторной работы

Начинаю работу с установки необходимого программного обеспечения на клиентской машине. Использую команду `dnf -y install nfs-utils` для установки утилит NFS, чтобы иметь возможность монтировать удалённые ресурсы (рис. 3.1).

```
[root@client.avlisovskaya.net ~]# dnf -y install nfs-utils
Extra Packages for Enterprise Linux 9 - x86_64 11 kB/s | 37 kB    00:03
Extra Packages for Enterprise Linux 9 - x86_64 2.8 MB/s | 20 MB    00:07
Rocky Linux 9 - BaseOS                  4.1 kB/s | 4.1 kB    00:00
Rocky Linux 9 - AppStream                3.6 kB/s | 4.5 kB    00:01
Rocky Linux 9 - Extras                  2.3 kB/s | 2.9 kB    00:01
Dependencies resolved.

=====
Package          Arch      Version       Repository  Size
=====
Installing:
nfs-utils        x86_64    1:2.5.4-34.el9   baseos     430 k
Upgrading:
libipa_hbac     x86_64    2.9.6-4.el9_6.2  baseos     35 k
libldb           x86_64    4.21.3-14.el9_6  baseos    177 k
libsmbclient     x86_64    4.21.3-14.el9_6  baseos     73 k
libssss_certmap  x86_64    2.9.6-4.el9_6.2  baseos     89 k
libssss_idmap    x86_64    2.9.6-4.el9_6.2  baseos     40 k
libssss_nss_idmap x86_64    2.9.6-4.el9_6.2  baseos     45 k
libssss_sudo     x86_64    2.9.6-4.el9_6.2  baseos     34 k
libtalloc         x86_64    2.4.2-1.el9    baseos     30 k
libtdb            x86_64    1.4.12-1.el9    baseos     50 k
libtevent          x86_64    0.16.1-1.el9    baseos     47 k
libwbclient       x86_64    4.21.3-14.el9_6  baseos     42 k
samba-client-libs x86_64    4.21.3-14.el9_6  baseos     5.2 M
samba-common      noarch    4.21.3-14.el9_6  baseos    173 k
samba-common-libs x86_64    4.21.3-14.el9_6  baseos     100 k
```

Рис. 3.1: Установка nfs-utils на клиенте

Перехожу к настройке сервера. Создаю основной каталог, который будет служить корнем дерева NFS, с помощью команды `mkdir -p /srv/nfs` (рис. 3.2).

```
omplete!
root@server.avlisovskaya.net ~]# mkdir -p /srv/nfs
root@server.avlisovskaya.net ~]# nano /etcexports
```

Рис. 3.2: Создание корневого каталога NFS на сервере

Для экспорта созданного каталога редактирую конфигурационный файл

/etc/exports. Добавляю строку /srv/nfs *(ro), разрешающую всем клиентам монтировать этот ресурс в режиме «только чтение» (рис. 3.3).

Рис. 3.3: Редактирование /etc/exports

Настраиваю контекст безопасности SELinux, чтобы сервер мог корректно предоставлять доступ к файлам. Выполняю команды semanage fcontext и restorecon для каталога /srv/nfs (рис. 3.4).

```
root@server.avlisovskaya.net ~]# nano /etc/exports
root@server.avlisovskaya.net ~]# semanage fcontext -a -t nfs_t "/srv/nfs(/. )?"
root@server.avlisovskaya.net ~]# |
```

Рис. 3.4: Настройка SELinux

Запускаю службу NFS-сервера и добавляю её в автозагрузку. Также настраивая межсетевой экран (firewall), разрешая работу службы nfs (рис. 3.5).

```
file context for /srv/nfs(/.*)? already defined, modifying instead
root@server.avlisovskaya.net ~]# restorecon -vR /srv/nfs
root@server.avlisovskaya.net ~]# systemctl start nfs-server.service
root@server.avlisovskaya.net ~]# systemctl enable nfs-server.service
reated symlink /etc/systemd/system/multi-user.target.wants/nfs-server.servi
e → /usr/lib/systemd/system/nfs-server.service.
root@server.avlisovskaya.net ~]# firewall-cmd --add-service=nfs
uccess
root@server.avlisovskaya.net ~]# firewall-cmd --add-service=nfs --permanent
uccess
root@server.avlisovskaya.net ~]# firewall-cmd --reload
uccess
root@server.avlisovskaya.net ~]# |
```

Рис. 3.5: Запуск служб и настройка Firewall

Убеждаюсь, что на клиенте установлены утилиты NFS (повторная проверка/установка для корректности выполнения шагов) (рис. 3.6).

```

eth0 metric 100
[root@client.avlisovskaya.net ~]# dnf -y install nfs-utils
Extra Packages for Enterprise Linux 9 - x86_64 11 kB/s | 37 kB  00:03
Extra Packages for Enterprise Linux 9 - x86_64 2.8 MB/s | 20 MB  00:07
Rocky Linux 9 - BaseOS 4.1 kB/s | 4.1 kB  00:00
Rocky Linux 9 - AppStream 3.6 kB/s | 4.5 kB  00:01
Rocky Linux 9 - Extras 2.3 kB/s | 2.9 kB  00:01
Dependencies resolved.
=====
  Package          Arch    Version        Repository   Size
=====
Installing:
  nfs-utils       x86_64  1:2.5.4-34.el9  baseos      430 k
Upgrading:
  libipa_hbac    x86_64  2.9.6-4.el9_6.2  baseos      35 k
  libldb         x86_64  4.21.3-14.el9_6  baseos     177 k
  libssmclient   x86_64  4.21.3-14.el9_6  baseos      73 k
  libsss_certmap x86_64  2.9.6-4.el9_6.2  baseos      89 k
  libsss_idmap   x86_64  2.9.6-4.el9_6.2  baseos      40 k
  libsss_nss_idmap x86_64  2.9.6-4.el9_6.2  baseos      45 k
  libsss_sudo    x86_64  2.9.6-4.el9_6.2  baseos      34 k
  libtalloc       x86_64  2.4.2-1.el9  baseos      30 k
  libtdb          x86_64  1.4.12-1.el9  baseos      50 k
  libtevent       x86_64  0.16.1-1.el9  baseos      47 k
  libwbclient    x86_64  4.21.3-14.el9_6  baseos      42 k
  samba-client-libs x86_64  4.21.3-14.el9_6  baseos      5.2 M
  samba-common   noarch  4.21.3-14.el9_6  baseos     173 k
  samba-common-libs x86_64  4.21.3-14.el9_6  baseos     100 k
  sssd           x86_64  2.9.6-4.el9_6.2  baseos      27 k
  sssd-ad        x86_64  2.9.6-4.el9_6.2  baseos     217 k
  sssd-client    x86_64  2.9.6-4.el9_6.2  baseos      159 k

```

Рис. 3.6: Повторная проверка пакетов на клиенте

Пробую просмотреть список экспортируемых ресурсов с клиента командой `showmount -e`. Получаю ошибку `clnt_create: RPC: Unable to receive`, что свидетельствует о блокировке соединений межсетевым экраном сервера (рис. 3.7).

```

dev eth1 metric 100
[root@client.avlisovskaya.net ~]# showmount -e server.avlisovskaya.net
clnt_create: RPC: Unable to receive\n
[root@client.avlisovskaya.net ~]#

```

Рис. 3.7: Ошибка при выполнении `showmount`

Для диагностики проблемы временно останавливаю службу `firewalld` на сервере (рис. 3.8).

```

success
[root@server.avlisovskaya.net ~]# systemctl stop firewalld.service
[root@server.avlisovskaya.net ~]#

```

Рис. 3.8: Остановка Firewall на сервере

Повторяю команду `showmount -e` на клиенте. Теперь список экспортируемых

ресурсов отображается корректно, что подтверждает проблему в настройках фаервола (рис. 3.9).

```
[root@client.avlisovskaya.net ~]# showmount -e server.avlisovskaya.net
Export list for server.avlisovskaya.net:
/srv/nfs
```

Рис. 3.9: Успешный вывод showmount

Включаю фаервол обратно и анализирую используемые порты. С помощью команды lsof | grep TCP просматриваю запущенные TCP-службы (рис. 3.10).

```
[root@server.avlisovskaya.net ~]# systemctl start firewalld
[root@server.avlisovskaya.net ~]# lsof | grep TCP
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/990/gvfs
      Output information may be incomplete.
systemd      1          root    53u    IPv4
4000      0t0      TCP *:sunrpc (LISTEN)
systemd      1          root    55u    IPv6
4054      0t0      TCP *:sunrpc (LISTEN)
cupsd     896          root    6u    IPv6
1569      0t0      TCP localhost:ipp (LISTEN)
cupsd     896          root    7u    IPv4
1570      0t0      TCP localhost:ipp (LISTEN)
sshd     908          root    3u    IPv4
1589      0t0      TCP *:down (LISTEN)
sshd     908          root    4u    IPv6
1598      0t0      TCP *:down (LISTEN)
sshd     908          root    5u    IPv4
1600      0t0      TCP *:ssh (LISTEN)
sshd     908          root    6u    IPv6
1602      0t0      TCP *:ssh (LISTEN)
named    942          named   3lu    IPv4
1192      0t0      TCP localhost:rndc (LISTEN)
named    942          named   34u    IPv4
```

Рис. 3.10: Проверка TCP сокетов

Аналогично проверяю UDP-сокеты командой lsof | grep UDP, чтобы понять, какие еще службы требуют разрешения в межсетевом экране (рис. 3.11).

780	0t0	TCP	*:mountd (LISTEN)			
root@server.avlisovskaya.net ~]# lsof grep UDP						
sof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/990/gvfs						
Output information may be incomplete.						
systemd	1	UDP	*:sunrpc	root	54u	IPv4
047	0t0	UDP	*:sunrpc	root	56u	IPv6
systemd	1	UDP	*:sunrpc	avahi	12u	IPv4
061	0t0	UDP	*:mdns	avahi	13u	IPv6
vahi-dae	578	UDP	*:mdns	avahi	14u	IPv4
217	0t0	UDP	*:mdns	avahi	15u	IPv6
vahi-dae	578	UDP	*:mdns	avahi	16u	IPv4
218	0t0	UDP	*:mdns	avahi	17u	IPv6
vahi-dae	578	UDP	*:46617	avahi	18u	IPv4
219	0t0	UDP	*:42546	chrony	5u	IPv4
vahi-dae	578	UDP	*:42546	chrony	6u	IPv6
220	0t0	UDP	localhost:323	chrony	7u	IPv4
chronyd	601	UDP	localhost:323	chrony	8u	IPv6
106	0t0	UDP	localhost:323	chrony	9u	IPv4
chronyd	601	UDP	localhost:323	chrony	10u	IPv6
chronyd	601	UDP	*:ntp	named	6u	IPv4
108	0t0	UDP	*:ntp	named	7u	IPv6
amed	942	UDP	*:ntp	named	8u	IPv4

Рис. 3.11: Проверка UDP сокетов

Добавляю разрешающие правила в `firewalld` для служб `mountd` и `rpc-bind`, которые необходимы для корректной работы NFS и RPC-запросов (рис. 3.12).

```
root@server.avlisovskaya.net ~]# firewall-cmd --get-services
H-Satellite-6 RH-Satellite-6-capsule afp amanda-client amanda-k5-client amq
amqps apcupsd audit ausweisapp2 bacula bacula-client bareos-director bareo
-filedaemon bareos-storage bb bgp bitcoin bitcoin-rpc bitcoin-testnet bitco
n-testnet-rpc bittorrent-lsd ceph ceph-exporter ceph-mon cfengine checkmk-a
ent cockpit collectd condor-collector cratedb ctdb dds dds-multicast dds-un
cast dhcp dhcipv6 dhcipv6-client distcc dns dns-over-tls docker-registry dock
r-swarm dropbox-lansync elasticsearch etcd-client etcd-server finger forema
foreman-proxy freeipa-4 freeipa-ldap freeipa-ldaps freeipa-replication fre
ipa-trust ftp galeria ganglia-client git gpgsql grafana gre hig
-availability http http3 https ident imap imaps ipfs ipp ipp-client ipsec i
c ircs iscsi-target isns jenkins kadmin kdeconnect kerberos kibana klogin k
asswd kprop kshell kube-api kube-apiserver kube-control-plane kube-control-
lane-secure kube-controller-manager kube-controller-manager-secure kube-nod
port-services kube-scheduler kube-scheduler-secure kube-worker kubelet kube
et-readonly kubelet-worker ldap ldaps libvirt libvirt-tls lightning-network
llmnr llmnr-client llmnr-tcp llmnr-udp managesieve matrix mdns memcache min
dlna mongod mosh mountd mqtt mqtt-tls ms-wbt mssql murmur mysql nbd nebula
netbios-ns netdata-dashboard nfs nfs3 nmea-0183 nrpe ntp openvpn ovirt-
imageio ovirt-storageconsole ovirt-vmconsole plex pmcd pmproxy pmwebapi pmwe
apis pop3 pop3s postgresql privoxy prometheus prometheus-node-exporter prox
-dhcp ps2link ps3netsrv ptp pulseaudio puppetmaster quassel radius rdp redi
redis-sentinel rpc-bind rquotad rsh rsyncd rtsp salt-master samba samba-cl
ient samba-de sane sip sips slp smtp smtp-submission smtsp snmp snmpvls snmp
ls-trap snmptrap spideroak-lansync spotify-sync squid ssdp ssh steam-stream
ng svdrp svn syncthing syncthing-gui syncthing-relay synergy syslog syslog-
ls telnet tentacle tftp tile38 tinc tor-socks transmission-client upnp-clie
t vdsm vnc-server warpinator wbem-http wbem-https wireguard ws-discovery ws
discovery-client ws-discovery-tcp ws-discovery-udp wsman wsmans xdmcp xmpp-
osh xmpp-client xmpp-local xmpp-server zabbix-agent zabbix-server zerotier
root@server.avlisovskaya.net ~]# firewall-cmd --add-service=mountd --add-se
vice=rpc-bind
uaccess
root@server.avlisovskaya.net ~]# firewall-cmd --add-service=mountd --add-se
vice=rpc-bind --permanent
uaccess
root@server.avlisovskaya.net ~]#
```

Рис. 3.12: Дополнительная настройка Firewall

После корректной настройки фаервола снова проверяю доступность ресурсов

с клиента. Команда `showmount` отрабатывает успешно (рис. 3.13).

```
[root@client.avlisovskaya.net ~]# showmount -e server.avlisovskaya.net
Export list for server.avlisovskaya.net:
[srv/nfs *
[root@client.avlisovskaya.net ~]#
```

Рис. 3.13: Проверка доступа после настройки Firewall

На клиенте создаю точку монтирования `/mnt/nfs` и монтирую удаленный ресурс сервера командой `mount` (рис. 3.14).

```
[root@client.avlisovskaya.net ~]# mkdir -p /mnt/nfs
[root@client.avlisovskaya.net ~]# mount server.avlisovskaya.net:/srv/nfs /mnt/nfs
[root@client.avlisovskaya.net ~]#
```

Рис. 3.14: Монтирование ресурса на клиенте

Проверяю успешность операции с помощью команды `mount`. Вижу, что ресурс сервера успешно смонтирован как тип `nfs4` (рис. 3.15).

```
root@client.avlisovskaya.net ~]# 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,seclabel,size=4096k,nr_inodes=219909,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=00)
tmpfs on /run type tmpfs (rw,nosuid,nodev,seclabel,size=363616k,nr_inodes=819200,mode=755,inode64)
group2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,seclabel,nsdelegate,memory_recursiveprot)
store on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime,seclabel)
pf 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,pipe_ino=19001)
ugeqlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel,pagesize=2M)
queue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime,seclabel)
ebugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime,seclabel)
racefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime,seclabel)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
usectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
one on /run/credentials/systemd-sysctl.service type ramfs (ro,nosuid,nodev,noexec,relatime)
```

Рис. 3.15: Проверка смонтированных файловых систем

Для автоматического монтиrovания ресурса при загрузке системы добавляю соответствующую запись в файл `/etc/fstab` на клиенте, используя опцию `_netdev` (рис. 3.16).

```

#UUID=c84cce45-9088-48d9-9617-2f1d1bd45fdd /          xfs      defaults
/swappfile 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
server.avlisovskaya.net:/srv/nfs /mnt/nfs nfs _netdev 0 0

```

Рис. 3.16: Редактирование /etc/fstab на клиенте

Проверяю статус службы `remote-fs.target`, отвечающей за монтирование удаленных файловых систем (рис. 3.17).

```

root@client.avlisovskaya.net ~]# 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 Mon 2025-11-24 16:52:02 UTC; 2min 4s ago
    Until: Mon 2025-11-24 16:52:02 UTC; 2min 4s ago
      Docs: man:systemd.special(7)

Nov 24 16:52:02 client.avlisovskaya.net systemd[1]: Reached target Remote F
[lines 1-7/7 (END)]
[1]+ Stopped                  systemctl status remote-fs.target
root@client.avlisovskaya.net ~]#

```

Рис. 3.17: Проверка статуса `remote-fs.target`

Перехожу к настройке экспорта каталога веб-сервера. На сервере создаю каталог `/srv/nfs/www` и выполняю bind-монтирование реального каталога `/var/www` в дерево NFS (рис. 3.18).

```

root@server.avlisovskaya.net ~]# mkdir -p /srv/nfs/www
root@server.avlisovskaya.net ~]# mount -o bind /var/www/ /srv/nfs/www/
root@server.avlisovskaya.net ~]# ls -la /srv/nfs/
total 0
drwxr-xr-x. 3 root root 17 Nov 24 16:54 .
drwxr-xr-x. 3 root root 17 Nov 24 16:21 ..
drwxr-xr-x. 6 root root 92 Oct 27 16:58 www
root@server.avlisovskaya.net ~]# ls -la /var/www/

```

Рис. 3.18: Монтируем каталог `www` на сервере

Проверяю на клиенте содержимое смонтированного ресурса `/mnt/nfs`. Появился каталог `www`, что означает успешное отображение изменений (рис. 3.19).

```

[2]+ Stopped                  [200~ls -la /mnt/nfs/
[root@client.avlisovskaya.net ~]# ls -la /mnt/nfs/
total 0
drwxr-xr-x. 3 root root 17 Nov 24 16:54 .
drwxr-xr-x. 3 root root 17 Nov 24 16:45 ..
drwxr-xr-x. 2 root root 6 Nov 24 16:54 www
root@client.avlisovskaya.net ~]# ls -la /var/www/

```

Рис. 3.19: Проверка видимости каталога `www` на клиенте

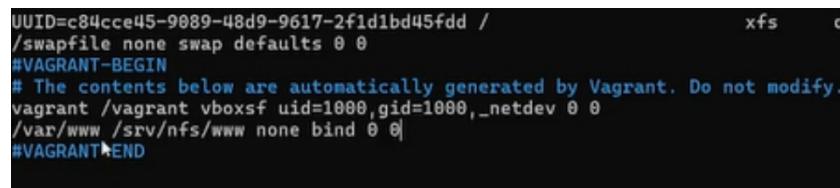
Редактирую `/etc(exports` на сервере, добавляя экспорт каталога `/srv/nfs/www` для локальной сети с правами на чтение и запись (рис. 3.20).



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

Рис. 3.20: Экспорт каталога www

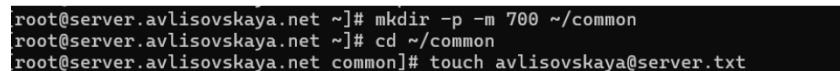
Чтобы bind-монтирование сохранялось после перезагрузки, добавляю соответствующую запись в файл `/etc/fstab` на сервере (рис. 3.21).



```
UUID=c84cce45-9089-48d9-9617-2f1d1bd45fdd /          xfs      d
/swappfile 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
/var/www /srv/nfs/www none bind 0 0
#VAGRANT-END
```

Рис. 3.21: Фиксация bind-монтирования в fstab

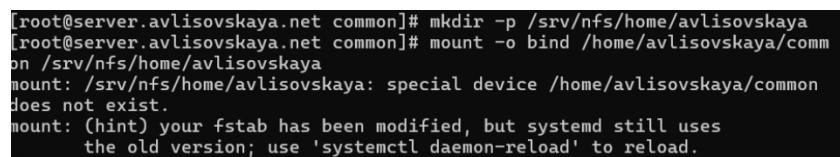
Перехожу к настройке пользовательских каталогов. Создаю директорию пользователя и тестовый файл в ней (рис. 3.22).



```
root@server.avlisovskaya.net ~]# mkdir -p -m 700 ~/common
root@server.avlisovskaya.net ~]# cd ~/common
root@server.avlisovskaya.net common]# touch avlisovskaya@server.txt
```

Рис. 3.22: Создание каталога пользователя

При попытке выполнить действия с пользовательскими каталогами возникла ошибка, связанная с отсутствием точки монтирования или применением настроек (рис. 3.23).



```
[root@server.avlisovskaya.net common]# mkdir -p /srv/nfs/home/avlisovskaya
[root@server.avlisovskaya.net common]# mount -o bind /home/avlisovskaya/common
on /srv/nfs/home/avlisovskaya
mount: /srv/nfs/home/avlisovskaya: special device /home/avlisovskaya/common
does not exist.
mount: (hint) your fstab has been modified, but systemd still uses
the old version; use 'systemctl daemon-reload' to reload.
```

Рис. 3.23: Ошибка монтирования пользовательского каталога

Добавляю экспорт домашнего каталога пользователя в /etc(exports, предоставляя права доступа для локальной сети (рис. 3.24).

```
GNU nano 5.6.1          /etc/exports          Modified [ ]  
/srv/nfs *(ro)  
/srv/nfs/www 192.168.0.0/16(rw)  
/srv/nfs/home/avlisovskaya| 192.168.0.0/16(rw)
```

Рис. 3.24: Экспорт домашнего каталога

Проверяю на клиенте содержимое /mnt/nfs. Теперь там отображаются и www, и home (рис. 3.25).

```
[root@client.avlisovskaya.net ~]# ls -la /mnt/nfs  
total 0  
drwxr-xr-x. 4 root root 29 Nov 24 17:07 .  
drwxr-xr-x. 3 root root 17 Nov 24 16:45 ..  
drwxr-xr-x. 3 root root 26 Nov 24 17:07 home  
drwxr-xr-x. 6 root root 92 Oct 27 16:58 www
```

Рис. 3.25: Проверка ресурсов на клиенте

Захожу в смонтированный каталог пользователя на клиенте, создаю файл и проверяю права доступа. Файлы успешно создаются и читаются (рис. 3.26).

```
[avlisovskaya@client.avlisovskaya.net ~]$ cd /mnt/nfs/home/avlisovskaya  
[avlisovskaya@client.avlisovskaya.net avlisovskaya]$ ls -la  
total 0  
drwxr-xr-x. 2 avlisovskaya avlisovskaya 37 Nov 24 17:09 .  
drwxr-xr-x. 3 root      root      26 Nov 24 17:07 ..  
-rw-r--r--. 1 avlisovskaya avlisovskaya  0 Nov 24 17:09 avlisovskaya@server.txt  
[avlisovskaya@client.avlisovskaya.net avlisovskaya]$ touch avlisovskaya@client.txt  
[avlisovskaya@client.avlisovskaya.net avlisovskaya]$ echo "Файл создан пользователем avlisovskaya с клиентской машины" > avlisovskaya@client.txt  
[avlisovskaya@client.avlisovskaya.net avlisovskaya]$ ls -la  
total 4  
drwxr-xr-x. 2 avlisovskaya avlisovskaya 68 Nov 24 17:27 .  
drwxr-xr-x. 3 root      root      26 Nov 24 17:07 ..  
-rw-r--r--. 1 avlisovskaya avlisovskaya 99 Nov 24 17:28 avlisovskaya@client.txt  
-rw-r--r--. 1 avlisovskaya avlisovskaya  0 Nov 24 17:09 avlisovskaya@server.txt
```

Рис. 3.26: Проверка прав доступа пользователя

Приступаю к автоматизации. На сервере подготавливаю структуру каталогов в /vagrant для создания скрипта провиженинга (рис. 3.27).

```
[avlisovskaya@server.avlisovskaya.net ~]$ cd /vagrant/provision/server
[avlisovskaya@server.avlisovskaya.net server]$ mkdir -p /vagrant/provision/server/nfs/etc
[avlisovskaya@server.avlisovskaya.net server]$ cp -R /etc/exports /vagrant/provision/server/nfs/etc/
[avlisovskaya@server.avlisovskaya.net server]$ cd /vagrant/provision/server
[avlisovskaya@server.avlisovskaya.net server]$ touch nfs.sh
[avlisovskaya@server.avlisovskaya.net server]$ chmod +x nfs.sh
[avlisovskaya@server.avlisovskaya.net server]$ nano nfs.sh
```

Рис. 3.27: Подготовка каталогов для скриптов

Создаю скрипт `nfs.sh` для сервера, который устанавливает пакеты, копирует конфиги, настраивает фаервол, SELinux и монтирует каталоги (рис. 3.28).

```
GNU nano 5.6.1                                     nfs.sh                                         Modified
#!/bin/bash
echo "Provisioning script $@"
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
```

Рис. 3.28: Скрипт настройки сервера `nfs.sh`

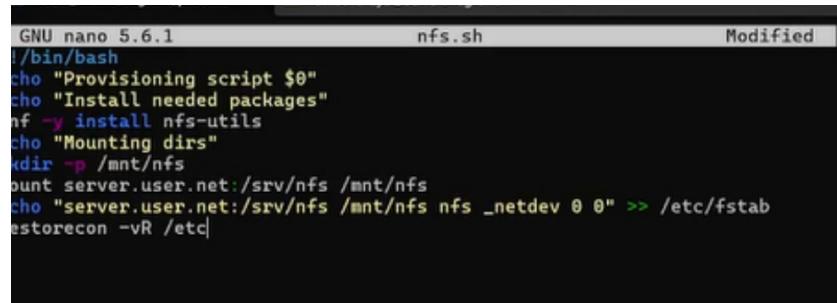
Аналогично создаю структуру и файл скрипта для клиентской машины (рис. 3.29).

```
[root@client.avlisovskaya.net avlisovskaya]# cd /vagrant/provision/client
[root@client.avlisovskaya.net client]# cd /vagrant/provision/client
[root@client.avlisovskaya.net client]# touch nfs.sh
[root@client.avlisovskaya.net client]# chmod +x nfs.sh
[root@client.avlisovskaya.net client]# nano nfs.sh
```

Рис. 3.29: Подготовка скрипта для клиента

Наполняю скрипт `nfs.sh` для клиента командами установки пакетов и авто-

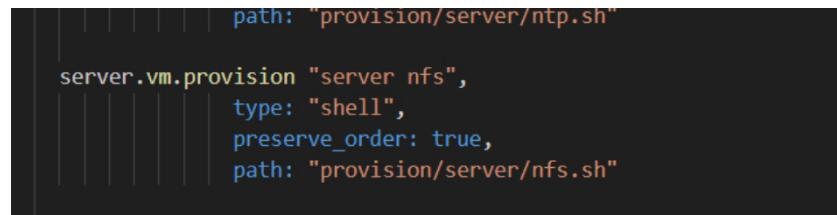
матического мониторинга ресурса при развертывании (рис. 3.30).



```
GNU nano 5.6.1          nfs.sh          Modified
#!/bin/bash
cho "Provisioning script $0"
cho "Install needed packages"
nf -y install nfs-utils
cho "Mounting dirs"
kdir -p /mnt/nfs
ount server.user.net:/srv/nfs /mnt/nfs
cho "server.user.net:/srv/nfs /mnt/nfs nfs _netdev 0 0" >> /etc/fstab
estorecon -vR /etc|
```

Рис. 3.30: Скрипт настройки клиента nfs.sh

Вношу изменения в Vagrantfile, добавляя секцию provision для запуска скрипта настройки на сервере (рис. 3.31).



```
path: "provision/server/ntp.sh"

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

Рис. 3.31: Настройка provision для сервера в Vagrantfile

Добавляю аналогичную секцию provision для клиента в Vagrantfile (рис. 3.32).



```
path: "provision/client/ntp.sh"

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

end
```

Рис. 3.32: Настройка provision для клиента в Vagrantfile

4 Выводы

В ходе выполнения лабораторной работы я приобрела навыки настройки NFS-сервера в Linux. Я научилась экспортировать файловые системы, управлять правами доступа через `/etc(exports` и настройки `firewall`, а также монтировать удаленные ресурсы на клиенте. Кроме того, была реализована автоматизация настройки стенда с использованием `Vagrant` и `bash`-скриптов.

5 Ответы на контрольные вопросы

1. **Как называется файл конфигурации, содержащий общие ресурсы NFS?** Файл конфигурации называется `/etc(exports`. В нем описываются экспортируемые каталоги, разрешенные клиенты и параметры доступа.
2. **Какие порты должны быть открыты в брандмауэре, чтобы обеспечить полный доступ к серверу NFS?** Для полноценной работы NFSv4 и сопутствующих служб необходимо открыть порты для служб:
 - `nfs` (TCP/2049)
 - `mountd` (обычно TCP/UDP 20048)
 - `rpc-bind` (TCP/UDP 111)
3. **Какую опцию следует использовать в `/etc/fstab`, чтобы убедиться, что общие ресурсы NFS могут быть установлены автоматически при перезагрузке?** Следует использовать опцию `_netdev`. Она указывает системе, что данное устройство требует сети, и предотвращает попытки монтирования до того, как сетевые интерфейсы будут подняты, что позволяет избежать зависания системы при загрузке.