Федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики им. Тихонова

Департамент электронной инженерии

ОТЧЕТ
О ПРАКТИЧЕСКОЙ РАБОТЕ №2

по дисциплине «Программные и аппаратные средства защиты информации»

«Основы безопасности ОС на базе Linux»

Студент гр. БИБ201

Шадрунов Алексей

Дата выполнения: 24 февраля 2023 г.

Преподаватель:

Перов А. А.

«___» _____ 2023 г.

Москва, 2023

Содержание

1 Цель работы **3**

2 Ход работы **3**

2.1 Подготовка виртуальной машины 3

2.2 Настройка сетевых интерфейсов 5

2.3 `adduser` 5

2.4 `passwd` 6

2.5 mc 7

2.6 `history` 12

2.7 `grep, cat, ls` 12

2.8 `chmod` 14

2.9 `arp` 15

2.10 ip 16

2.11 `ping` 17

2.12 `traceroute` 18

2.13 `netstat` 19

2.14 `nslookup` 20

2.15 Работа с `python` 21

2.16 Механизм безопасности 22

3 Выводы о проделанной работе **23**

1 Цель работы

Целью данной лабораторной работы является обучение студентов базовым навыками обращения с ОС класса Linux и основам обеспечения безопасности данных систем.

2 Ход работы

2.1 Подготовка виртуальной машины

Создадим виртуальную машину в гипервизоре `gnome-boxes` (`kvm`). Процесс изображён на рисунках 1-3.

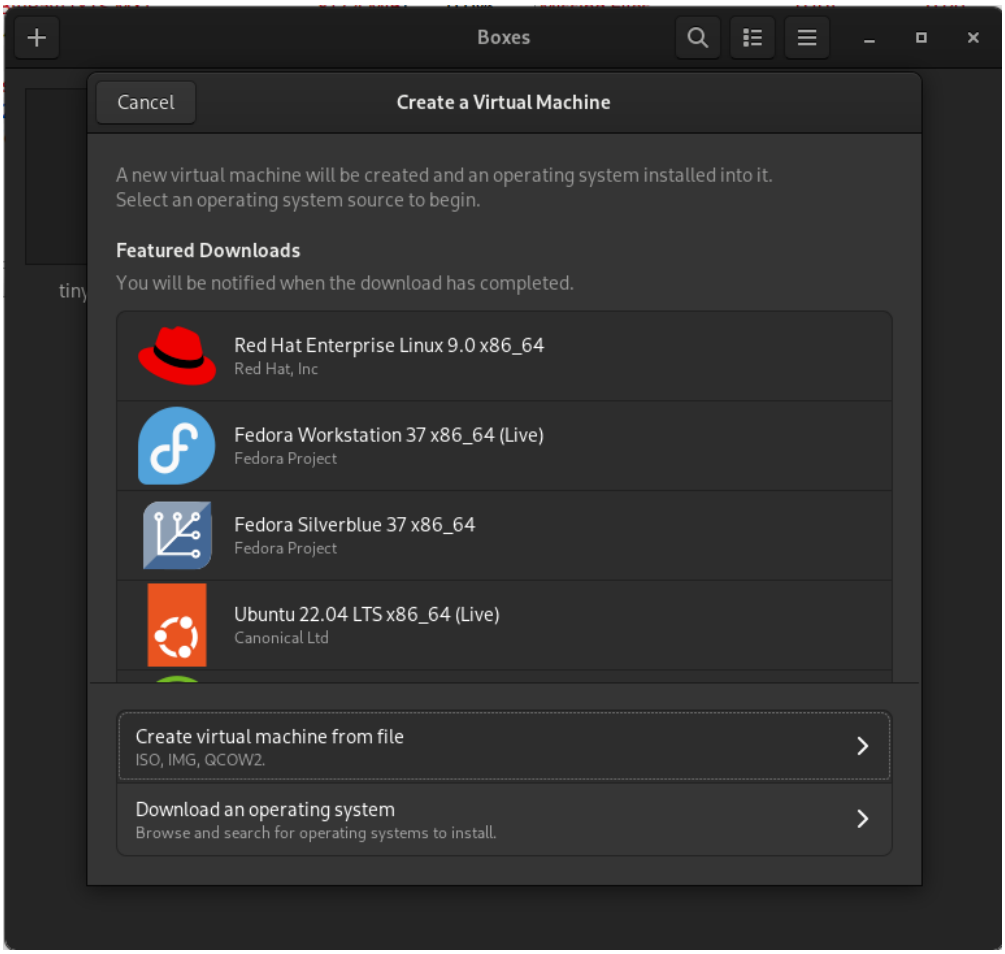

Рисунок 1 – Установка ВМ

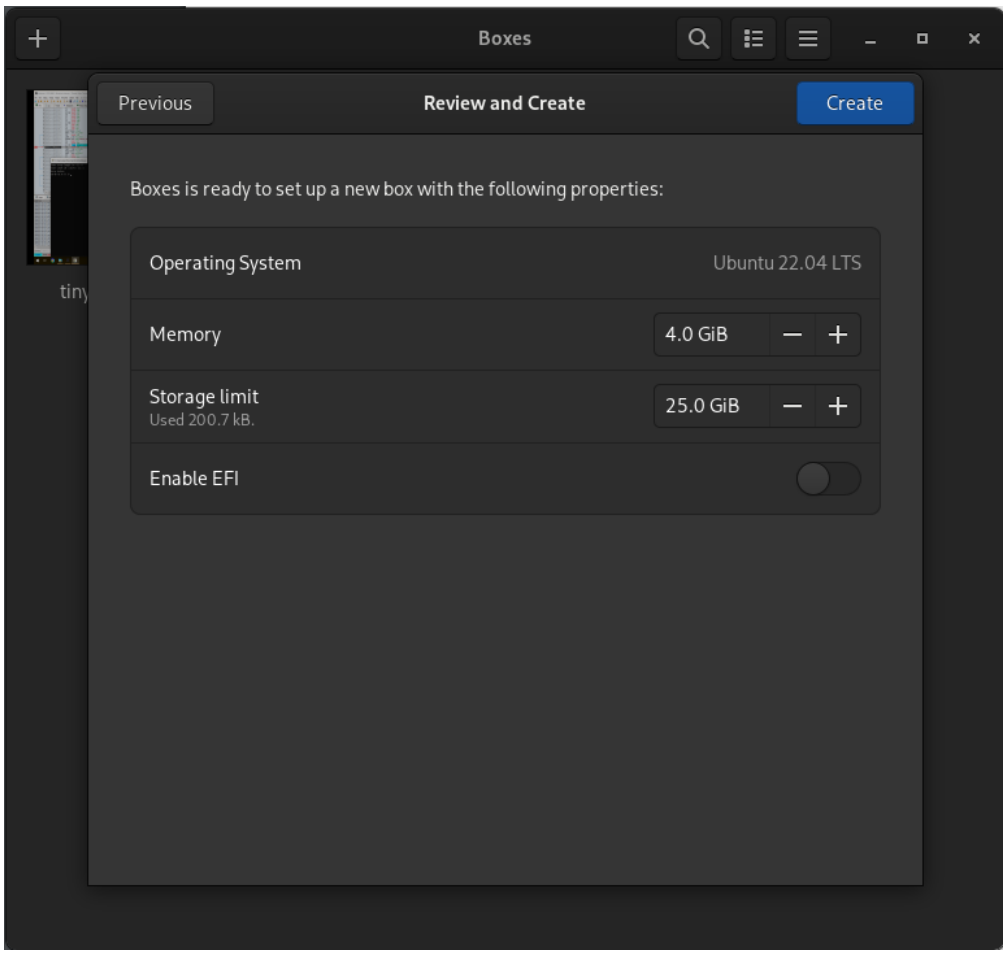

Рисунок 2 – Установка ВМ

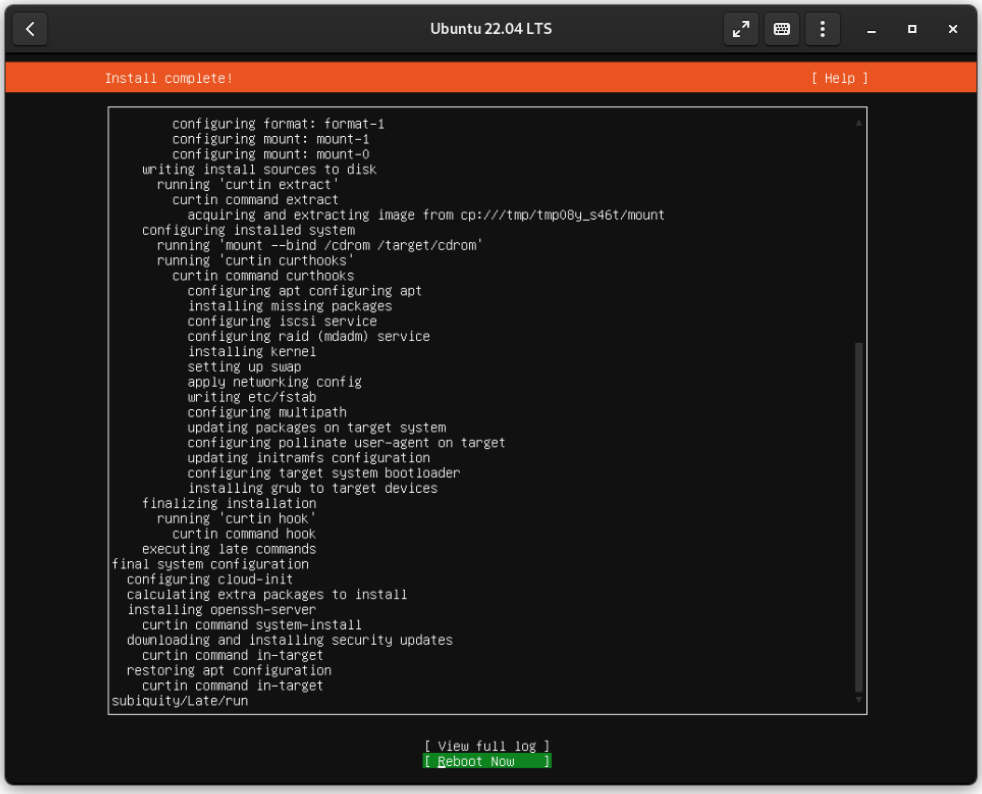

Рисунок 3 – Установка ВМ

2.2 Настройка сетевых интерфейсов

Для настройки сетевых интерфейсов используется утилита `netplan`. По умолчанию настроено получение адреса по `DHCP` (рисунок 4).

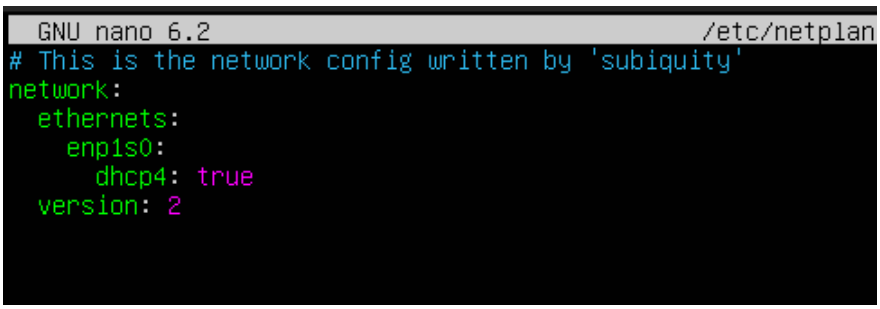

Рисунок 4 – `DHCP`

Настроим статический адрес. Для этого в файл `/etc/netplan/00-installer-config.yaml` пропишем следующую конфигурацию (рисунок 5). Применим конфигурацию и проверим результат (рисунок 6).

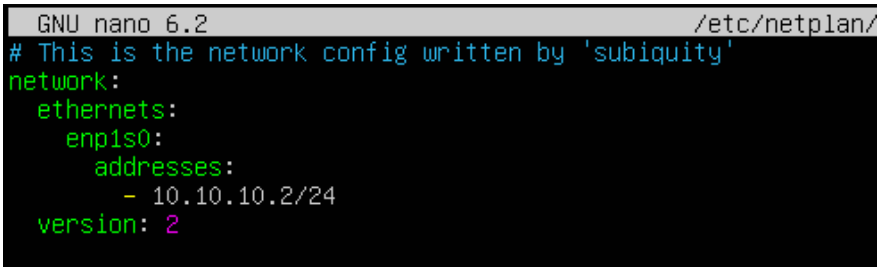

Рисунок 5 – Static

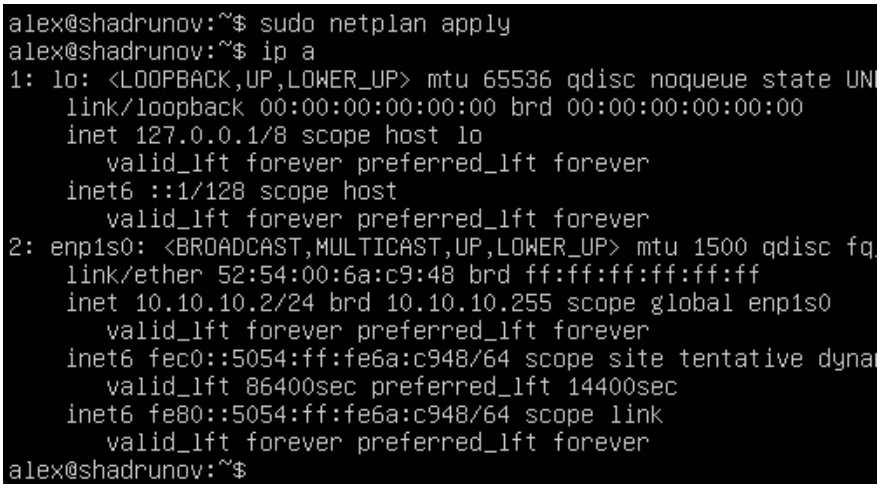

Рисунок 6 – Применение конфигурации

Вернёмся к автоматической конфигурации.

2.3 `adduser`

Для создания пользователей в Linux существует утилита `adduser`. Параметры команды позволяют задать `uid/gid`, домашний каталог, имя группы, оболочку. Создадим двух пользователей (рисунки 7-8).

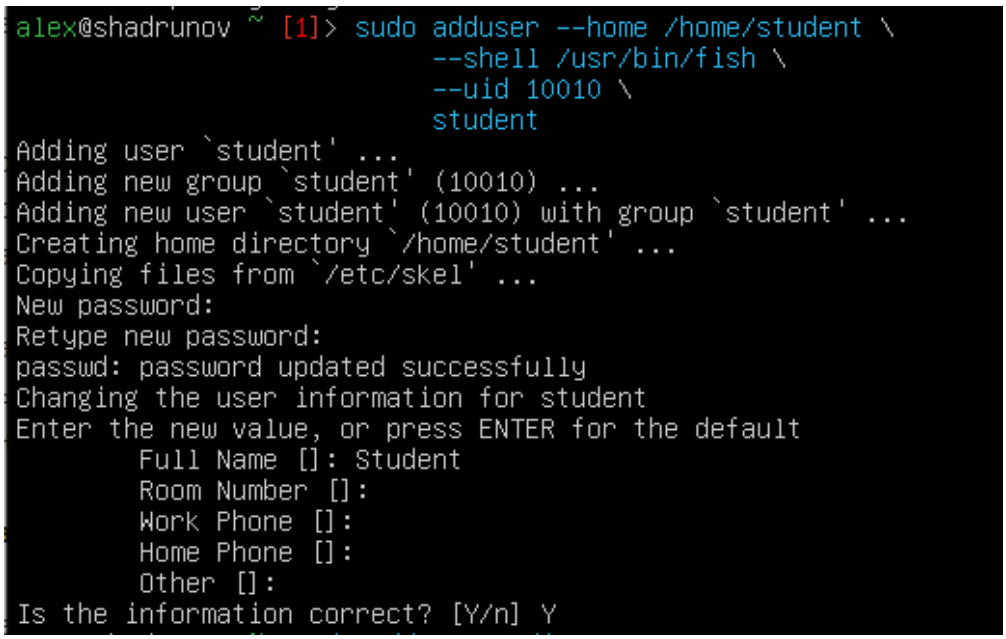

Рисунок 7 – `student`

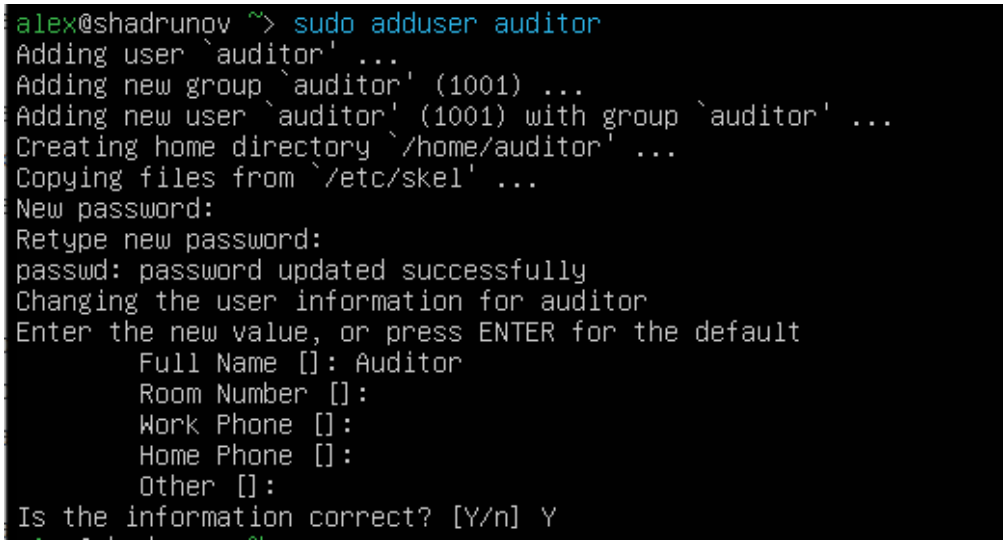

Рисунок 8 – `auditor`

2.4 `passwd`

Для смены пароля учётной записи используется утилита `passwd`. Для смены своего пароля достаточно выполнить команду `passwd`, для смены пароля чужой учётной записи нужно выполнить `sudo passwd <user>` (рисунок 9).

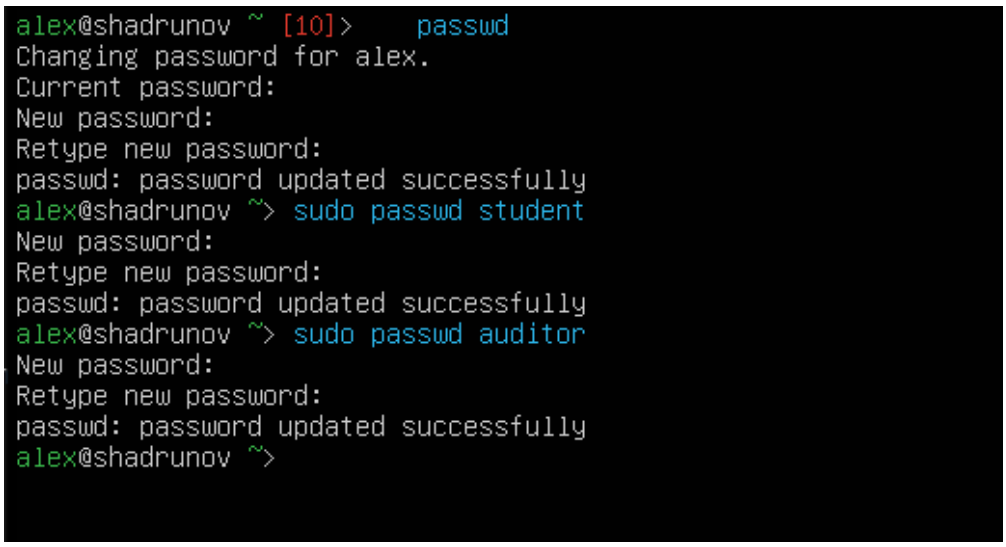

Рисунок 9 – Смена пароля

Также утилита позволяет отключить пароль или заставить пользователя сменить пароль при следующем входе или заблокировать аккаунт.

2.5 mc

В Linux есть файловый менеджер с псевдографическим интерфейсом `Midnight Commander`. Для установки используется команда `sudo apt install mc`. Рабочее окно `mc` представлено на рисунке 10.

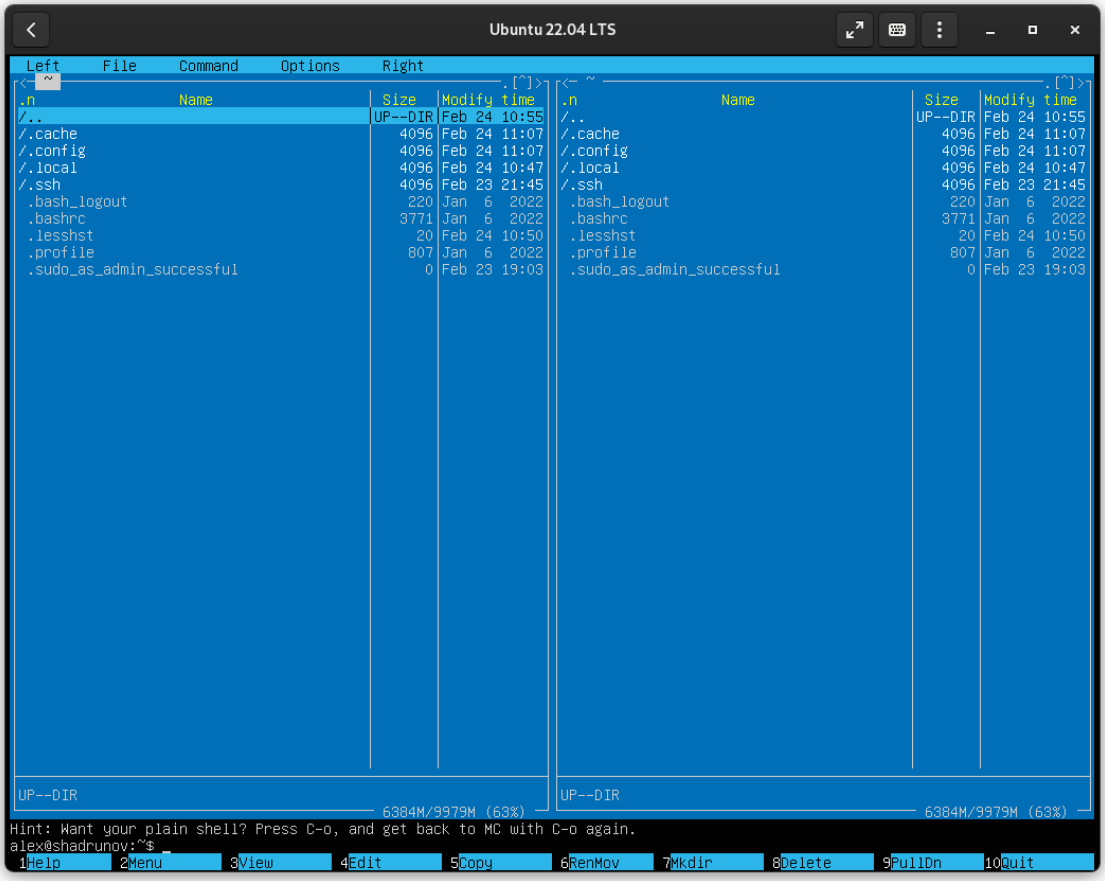

Рисунок 10 – mc

Для поиска файлов нужно перейти в меню `Command > Find file` (рисунок 11). Далее можно ввести имя файла и файл будет найден (рисунки 12-13).

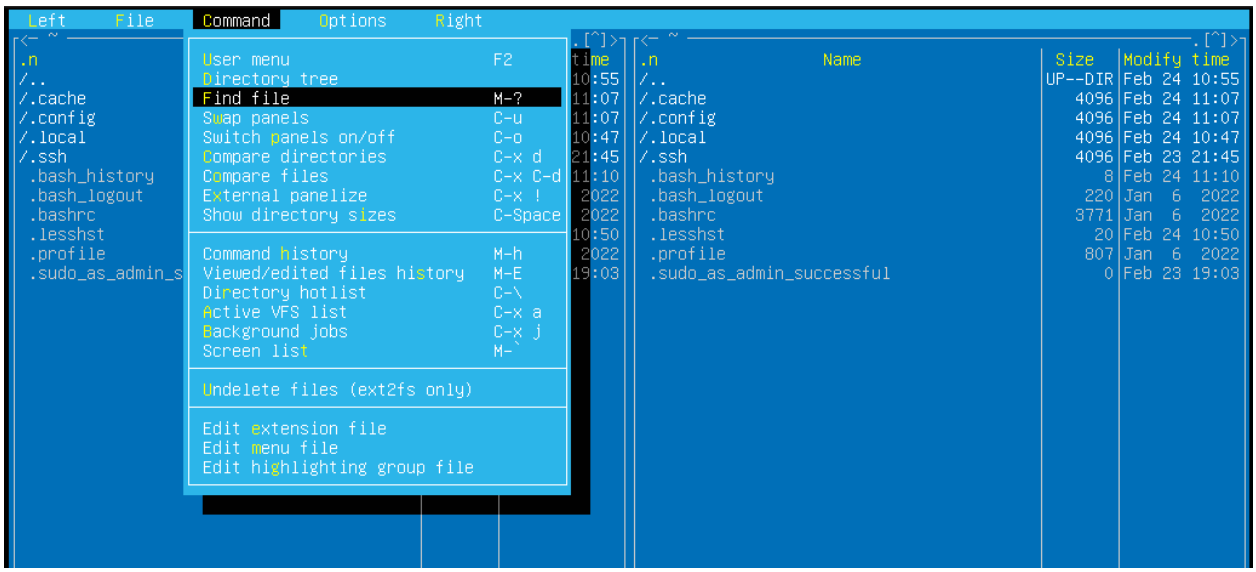

Рисунок 11 – `Command > Find file`

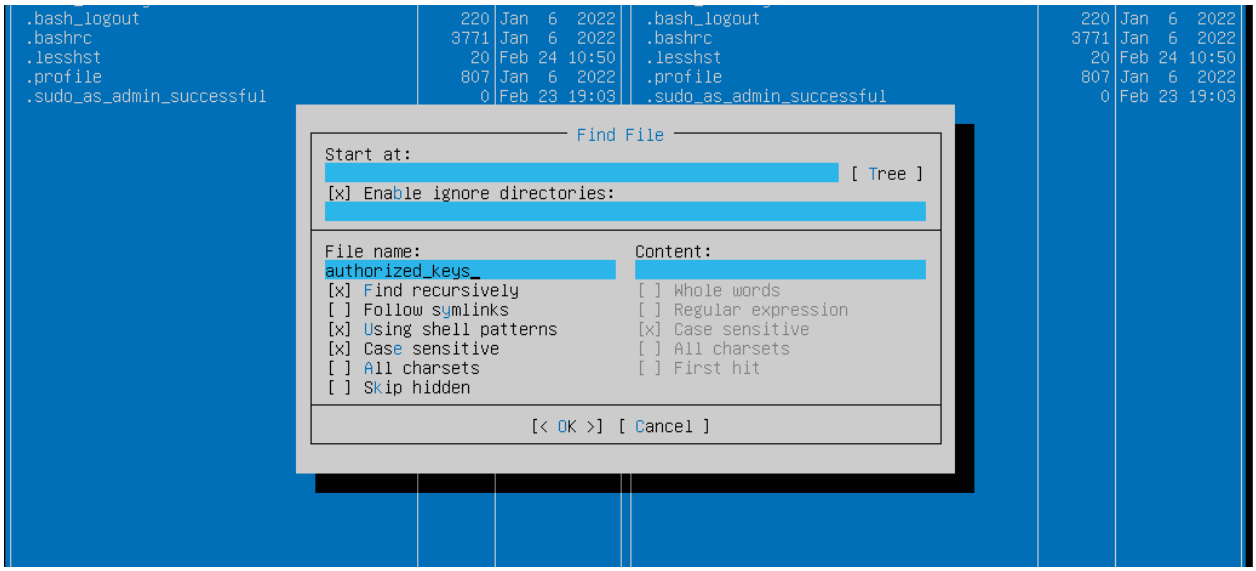

Рисунок 12 – `File name`

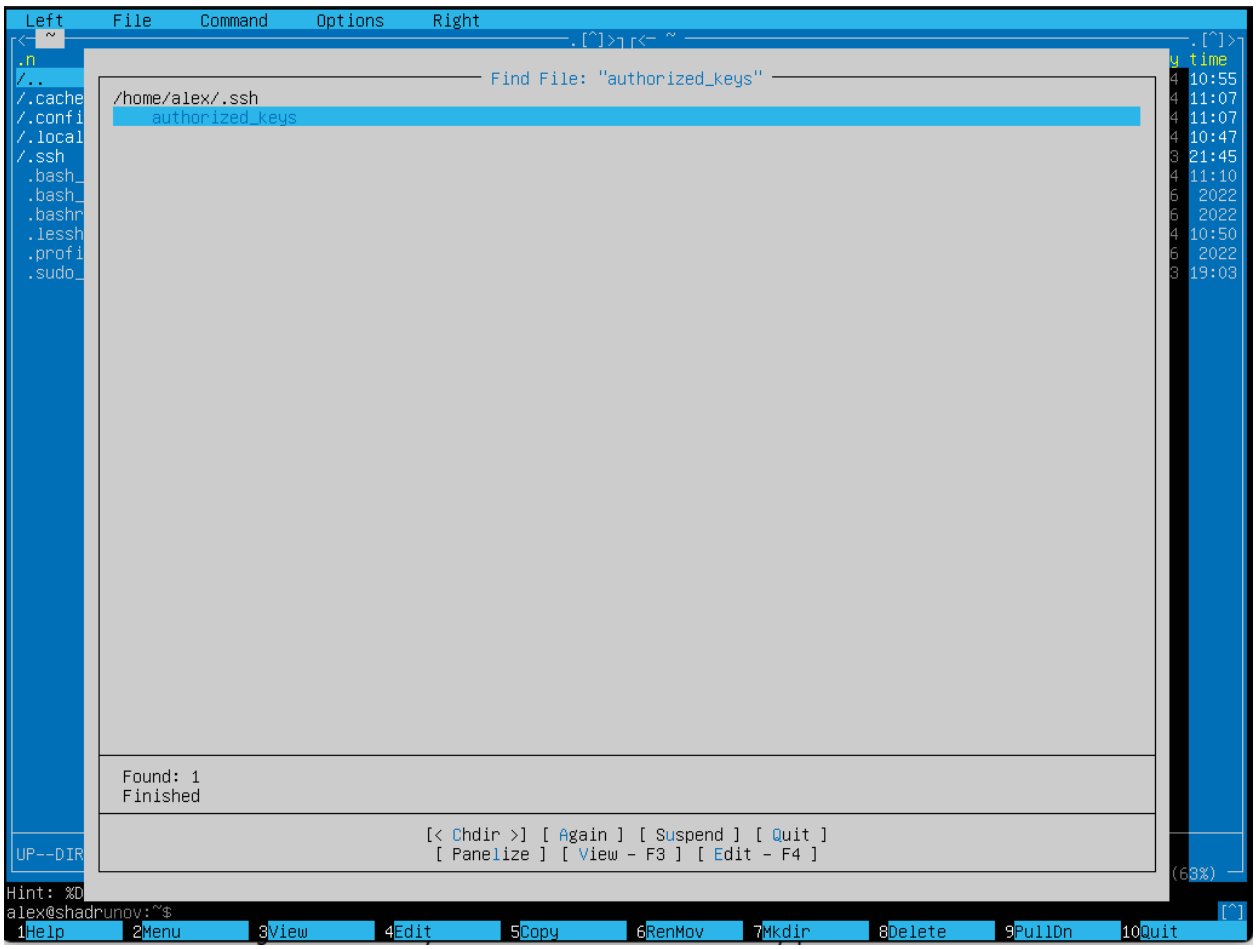

Рисунок 13 – `Results`

Далее создадим файл с помощью команды `touch` (рисунок 14). Новый файл появляется в списке (рисунок 15).

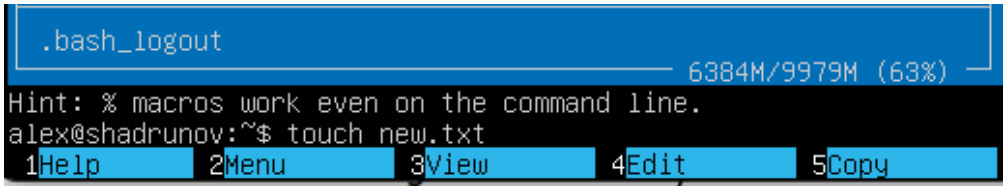

Рисунок 14 – Команда `touch`

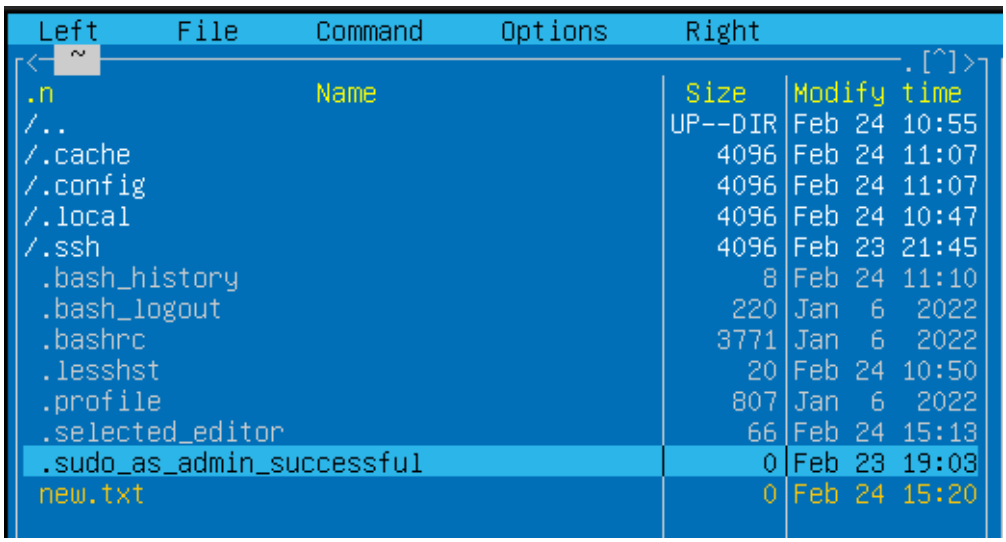

Рисунок 15 – Новый файл

Для редактирования нужно выбрать кнопку F4 (`Edit`). Открывается текстовый редактор (рисунок 16).

Рисунок 16 – Редактирование

Далее для просмотра разрешений перейдём в меню `File > Chmod` (рисунок 17). Появится окно с просмотром разрешений (рисунок 18).

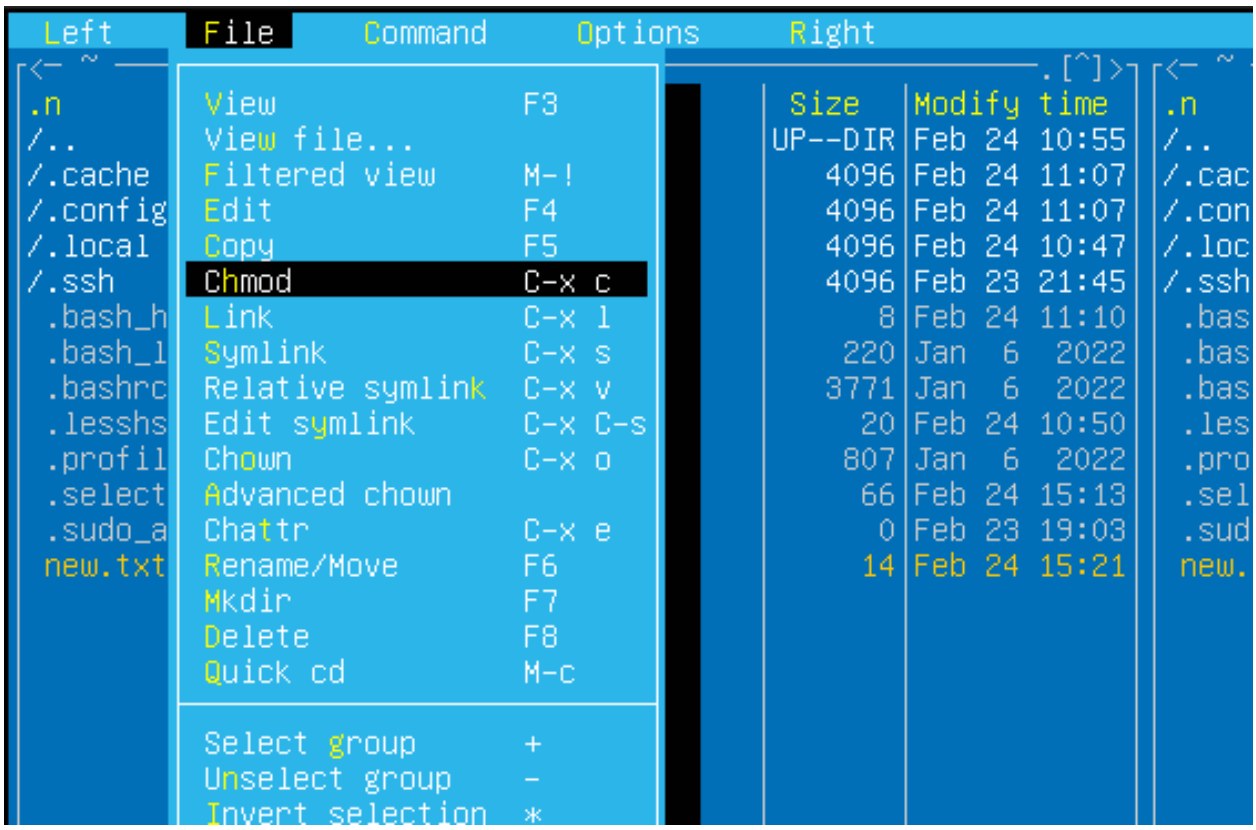

Рисунок 17 – `File > Chmod`

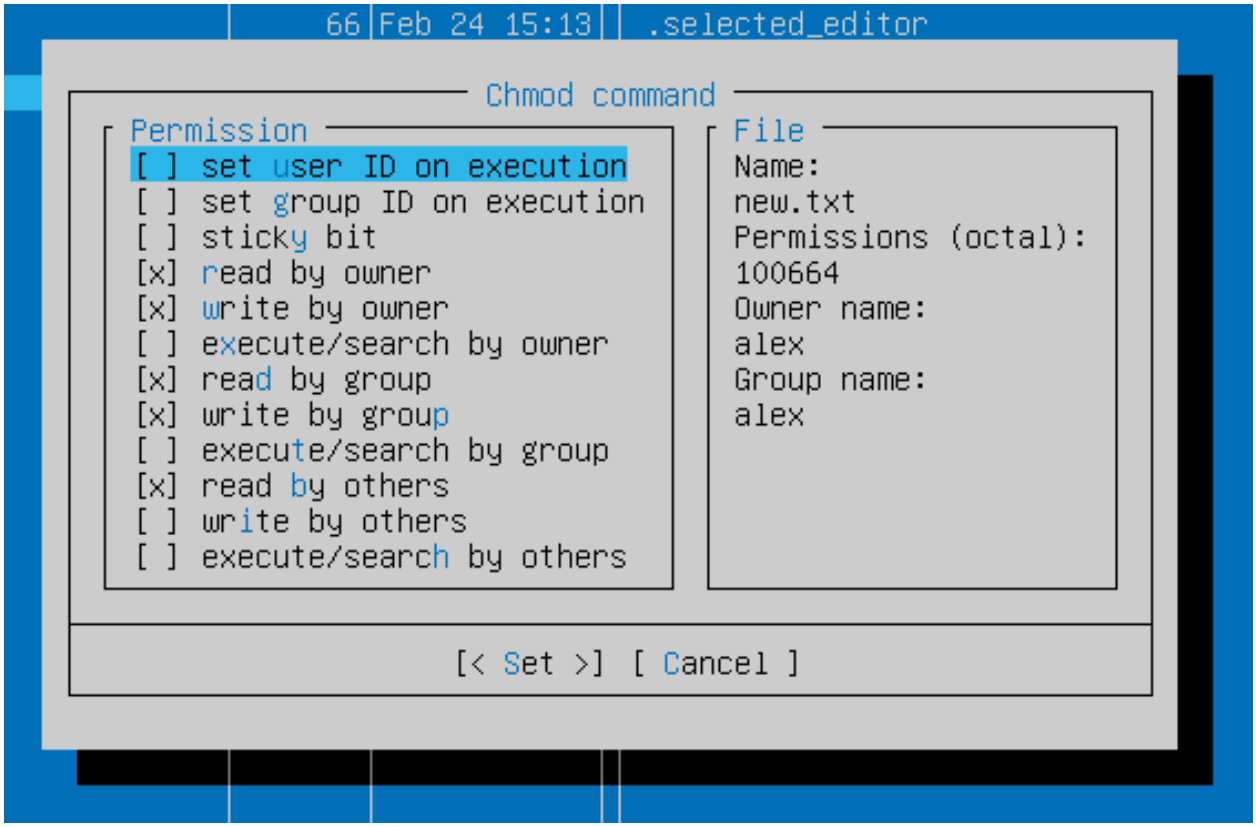

Рисунок 18 – `Chmod`

2.6 history

Для просмотра истории команд можно использовать утилиту `history`. Она отображает содержание файла `.bash_history` (для оболочки `bash`). В истории команд можно искать с помощью команды `grep` (рисунки 19-20). Можно настроить различные параметры, например, глубину хранения, с помощью переменных окружения `HISTSIZE` (сколько команд хранится в текущей истории) и `HISTFILESIZE` (сколько команд из текущей истории записывается в файл `.bash_history`).

```
alex@shadrunov:~$ ls -a
.   .bash_history  .bashrc  .config   .local   .profile          .ssh
..  .bash_logout   .cache   .lesshst  new.txt  .selected_editor  .sudo_as_admin_successful
alex@shadrunov:~$ history
    1  File
    2  ls
    3  cccccCC
    4  [D
    5  touch new.txt
    6  ls -la
    7  history
    8  ls -a
    9  history
alex@shadrunov:~$ history | grep ls
    2  ls
    6  ls -la
    8  ls -a
   10  history | grep ls
alex@shadrunov:~$ _
```

Рисунок 19 – Просмотр истории

```
alex@shadrunov:~$ export HISTSIZE=2
alex@shadrunov:~$ ls
new.txt
alex@shadrunov:~$ ls -a
.   .bash_history  .bashrc  .config   .local   .profile          .ssh
..  .bash_logout   .cache   .lesshst  new.txt  .selected_editor  .sudo_as_admin_successful
alex@shadrunov:~$ pwd
/home/alex
alex@shadrunov:~$ history
   13  pwd
   14  history
alex@shadrunov:~$ _
```

Рисунок 20 – Ограничение истории до 2 команд

2.7 grep, cat, ls

Для манипуляций с файлами есть команды `grep, cat, ls`. Первая позволяет выбрать строки из файлов по определённому параметру. Можно использовать вместе с оператором `pipe` (рисунки 21-22). Также можно использовать регулярные выражения.

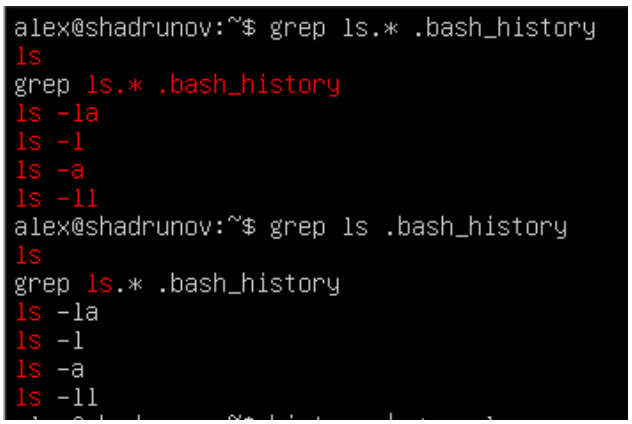

Рисунок 21 – Команды grep

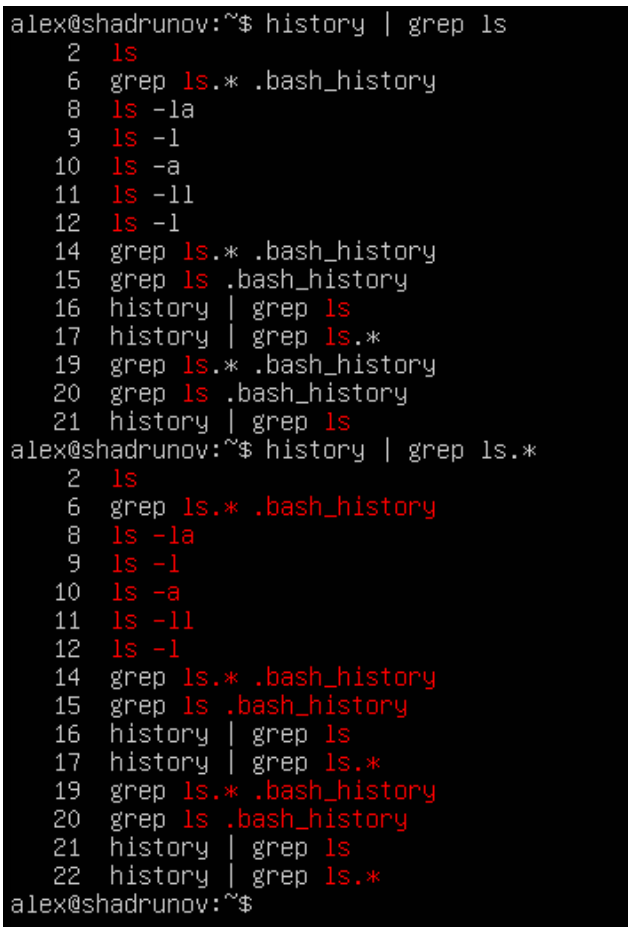

Рисунок 22 – Регулярные выражения

Команда cat читает последовательно файлы и выдает их содержимое в стандартный выходной поток. Одна из полезных опций — `-n` — позволяет выдавать порядковый номер строки перед каждой строкой (рисунок 23).

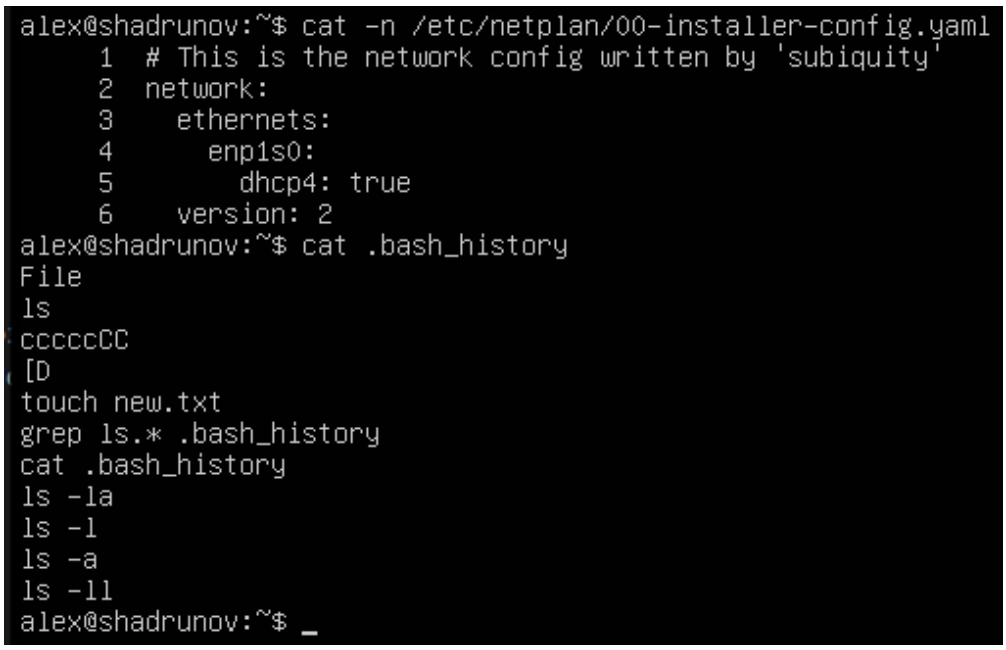

Рисунок 23 – Команда `cat`

Команда `ls` выводит список файлов. Полезные опции: `-l` — вывод списком, `-a` — вывод скрытых файлов (рисунок 24).

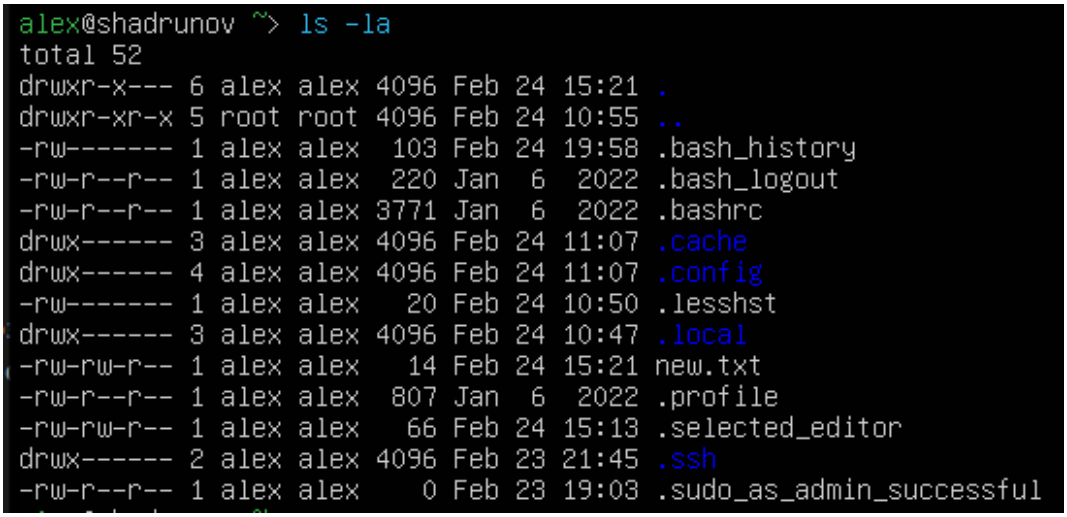

Рисунок 24 – Команда `ls`

2.8 chmod

Для манипуляций с разрешениями используется команда `chmod`. Для выставления разрешений используются буквенный или цифровой синтаксисы (рисунок 25).

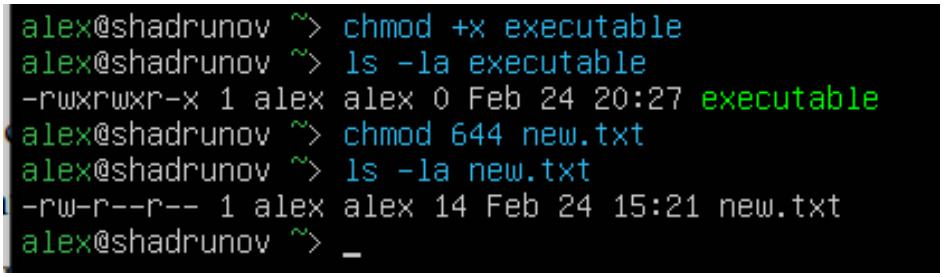

Рисунок 25 – Команда `chmod`

2.9 arp

Команда `arp` работает с arp-таблицей, то есть соответствием `ip-` и `mac-`адресов. Без ключей выводит кэш таблицы (рисунок 25). Также можно добавлять или удалять строки (ключи `-d` и `-s`) или выбирать форматирование (рисунок 26).

```
alex@shadrunov ~ [255]> arp
Address                  HWtype  HWaddress           Flags Mask            Iface
_gateway                 ether   52:55:0a:00:02:02   C                     enp1s0
10.0.2.3                 ether   52:55:0a:00:02:03   C                     enp1s0
alex@shadrunov ~> sudo arp -d 10.0.2.4 -v
arp: SIOCDARP(dontpub)
arp: SIOCDARP(pub)
No ARP entry for 10.0.2.4
```

Рисунок 26 – Команда arp

```
alex@shadrunov ~ [255]> arp -e -v -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.2                 ether   52:55:0a:00:02:02   C                     enp1s0
10.0.2.3                 ether   52:55:0a:00:02:03   C                     enp1s0
Entries: 2      Skipped: 0      Found: 2
alex@shadrunov ~> _
```

Рисунок 27 – Команда arp с ключами

2.10 ip

Команда `ip` позволяет настроить сеть и сетевые интерфейсы. У неё очень много параметров и подкоманд. `ip link` используется для отображения и изменения сетевых интерфейсов (рисунок 27). `ip monitor` позволяет просматривать сетевые события (рисунок 28). `ip route` позволяет работать с таблицей маршрутизации (рисунок 29). Кроме того, можно управлять адресами (ip addr) и arp-таблицей (ip neigh).

```
alex@shadrunov ~> ip monitor
10.0.2.3 dev enp1s0 lladdr 52:55:0a:00:02:03 PROBE
10.0.2.3 dev enp1s0 lladdr 52:55:0a:00:02:03 REACHABLE
10.0.2.2 dev enp1s0 lladdr 52:55:0a:00:02:02 REACHABLE
2: enp1s0: <BROADCAST,MULTICAST> mtu 1500 qdisc fq_codel state DOWN group default
    link/ether 52:54:00:6a:c9:48 brd ff:ff:ff:ff:ff:ff
Deleted 10.0.2.2 dev enp1s0 lladdr 52:55:0a:00:02:02 REACHABLE
Deleted 10.0.2.3 dev enp1s0 lladdr 52:55:0a:00:02:03 REACHABLE
Deleted fe80::/64 dev enp1s0 proto kernel metric 256 pref medium
Deleted fec0::/64 dev enp1s0 proto ra metric 100 expires 85860sec pref medium
Deleted default via fe80::2 dev enp1s0 proto ra metric 100 expires 1260sec pref medium
Deleted local fe80::5054:ff:fe6a:c948 dev enp1s0 table local proto kernel metric 0 pref medium
Deleted local fec0::5054:ff:fe6a:c948 dev enp1s0 table local proto kernel metric 0 pref medium
Deleted multicast ff00::/8 dev enp1s0 table local proto kernel metric 256 pref medium
Deleted fe80::2 dev enp1s0 lladdr 52:56:00:00:00:02 router STALE
Deleted ff02::2 dev enp1s0 lladdr 33:33:00:00:00:02 NOARP
Deleted ff02::1:ff00:3 dev enp1s0 lladdr 33:33:ff:00:00:03 NOARP
Deleted ff02::16 dev enp1s0 lladdr 33:33:00:00:00:16 NOARP
Deleted ff02::1:ff6a:c948 dev enp1s0 lladdr 33:33:ff:6a:c9:48 NOARP
Deleted fec0::3 dev enp1s0 lladdr 52:56:00:00:00:03 router STALE
Deleted 2: enp1s0    inet6 fec0::5054:ff:fe6a:c948/64 scope site dynamic mngtmpaddr noprefixroute
       valid_lft 85861sec preferred_lft 13861sec
Deleted 2: enp1s0    inet6 fe80::5054:ff:fe6a:c948/64 scope link
       valid_lft forever preferred_lft forever
Deleted 2: enp1s0    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp1s0
       valid_lft 49351sec preferred_lft 49351sec
Deleted local 10.0.2.15 dev enp1s0 table local proto kernel scope host src 10.0.2.15
```

Рисунок 28 – Команда `ip link`

```
alex@shadrunov ~> ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:6a:c9:48 brd ff:ff:ff:ff:ff:ff
alex@shadrunov ~> ip link show dev enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:6a:c9:48 brd ff:ff:ff:ff:ff:ff
alex@shadrunov ~> ip link ls up
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:6a:c9:48 brd ff:ff:ff:ff:ff:ff
alex@shadrunov ~> sudo ip link set enp1s0 up
alex@shadrunov ~> sudo ip link set enp1s0 down
alex@shadrunov ~> ip link ls up
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

Рисунок 29 – Команда `ip monitor`

```
alex@shadrunov ~ [2]> ip route
default via 10.0.2.2 dev enp1s0 proto dhcp src 10.0.2.15 metric 100
10.0.2.0/24 dev enp1s0 proto kernel scope link src 10.0.2.15 metric 100
10.0.2.2 dev enp1s0 proto dhcp scope link src 10.0.2.15 metric 100
10.0.2.3 dev enp1s0 proto dhcp scope link src 10.0.2.15 metric 100
alex@shadrunov ~> sudo ip route del default
alex@shadrunov ~> ip route
10.0.2.0/24 dev enp1s0 proto kernel scope link src 10.0.2.15 metric 100
10.0.2.2 dev enp1s0 proto dhcp scope link src 10.0.2.15 metric 100
10.0.2.3 dev enp1s0 proto dhcp scope link src 10.0.2.15 metric 100
```

Рисунок 30 – Команда `ip route`

2.11 `ping`

Команда `ping` позволяет проверять доступность хостов. Для работы нужно выполнить команду `ping` и адрес хоста (рисунок 31). Аргументы команды отображены на рисунке 32.

```
alex@shadrunov ~> ping 8.8.8.8 -A
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=32.8 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=33.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=34.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=255 time=33.0 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=255 time=32.2 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=255 time=32.9 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=255 time=31.7 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=255 time=33.1 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=255 time=32.5 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=255 time=32.9 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=255 time=31.4 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=255 time=33.2 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=255 time=32.8 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=255 time=34.1 ms
64 bytes from 8.8.8.8: icmp_seq=15 ttl=255 time=42.5 ms
^C
--- 8.8.8.8 ping statistics ---
16 packets transmitted, 15 received, 6.25% packet loss, time 505ms
rtt min/avg/max/mdev = 31.437/33.509/42.470/2.512 ms, pipe 2, ipg/ewma 33
alex@shadrunov ~> ping 8.8.8.8 -c 3
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=213 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=49.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=68.2 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 49.035/109.974/212.692/73.052 ms
alex@shadrunov ~> ping 8.8.8.8 -i 10
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=66.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=54.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=43.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=255 time=78.9 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 30032ms
rtt min/avg/max/mdev = 43.617/60.923/78.856/13.145 ms
alex@shadrunov ~> ping 8.8.8.8 -q
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 45.655/51.785/63.646/6.350 ms
```

Рисунок 31 – Команда `ping`

```
alex@shadrunov ~> ping 8.8.8.8 -s 100000
PING 8.8.8.8 (8.8.8.8) 100000(100028) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
17 packets transmitted, 0 received, 100% packet loss, time 16392ms

alex@shadrunov ~ [1]> ping 8.8.8.8 -s 100
PING 8.8.8.8 (8.8.8.8) 100(128) bytes of data.
108 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=49.8 ms
108 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=99.7 ms
108 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=34.9 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 34.903/61.443/99.657/27.694 ms
alex@shadrunov ~> ping 8.8.8.8 -t 64
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=195 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=63.2 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=62.8 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 62.808/107.146/195.387/62.395 ms
```

Рисунок 32 – Команда `ping`

2.12 `traceroute`

Команда `traceroute` позволяет отобразить маршрут до удалённых хостов. Механизм работы основан на поле ip-пакета `ttl`. Примеры работы команды приведены на рисунке 33.

```
alex@shadrunov ~> traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  _gateway (10.0.2.2)  0.324 ms  0.260 ms  0.246 ms
 2  192.168.43.1 (192.168.43.1)  3.457 ms  3.445 ms  3.555 ms
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  178.176.133.15 (178.176.133.15)  41.135 ms 209.85.168.98 (209.85.168.98)  39.198 ms  21.4
12  108.170.250.113 (108.170.250.113)  37.779 ms 108.170.250.130 (108.170.250.130)  25.804 ms
0.270 ms
13  142.250.238.138 (142.250.238.138)  36.562 ms 216.239.51.32 (216.239.51.32)  38.916 ms 172
8 ms
14  142.251.238.70 (142.251.238.70)  37.715 ms 172.253.66.110 (172.253.66.110)  42.350 ms 142
3 ms
15  142.250.209.161 (142.250.209.161)  36.545 ms 216.239.47.203 (216.239.47.203)  36.928 ms 7
1 ms
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * dns.google (8.8.8.8)  35.889 ms
```

Рисунок 33 – Команда `traceroute`

2.13 `netstat`

Команда `netstat` позволяет отобразить сетевую статистику, используемые порты, интерфейсы и процессы, их использующие. Примеры работы команды приведены на рисунках 34-35.

```
alex@shadrunov ~> netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
udp        0      0 localhost:domain        0.0.0.0:*
udp        0      0 shadrunov:bootpc        0.0.0.0:*
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     28632    /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     28638    /run/user/1000/bus
unix  2      [ ACC ]     STREAM     LISTENING     24619    @/org/kernel/linux/storage/multipathd
unix  2      [ ACC ]     STREAM     LISTENING     28640    /run/user/1000/gnupg/S.dirmngr
unix  2      [ ACC ]     STREAM     LISTENING     28642    /run/user/1000/gnupg/S.gpg-agent.browser
unix  2      [ ACC ]     STREAM     LISTENING     28644    /run/user/1000/gnupg/S.gpg-agent.extra
unix  2      [ ACC ]     STREAM     LISTENING     28646    /run/user/1000/gnupg/S.gpg-agent.ssh
unix  2      [ ACC ]     STREAM     LISTENING     28648    /run/user/1000/gnupg/S.gpg-agent
unix  2      [ ACC ]     STREAM     LISTENING     28650    /run/user/1000/pk-debconf-socket
unix  2      [ ACC ]     STREAM     LISTENING     28652    /run/user/1000/snapd-session-agent.socket
unix  2      [ ACC ]     STREAM     LISTENING     36975    /var/snap/lxd/common/lxd-user/unix.socket
unix  2      [ ACC ]     STREAM     LISTENING     28942    /run/systemd/resolve/io.systemd.Resolve
unix  2      [ ACC ]     STREAM     LISTENING     24888    /run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     24891    /run/snapd.socket
unix  2      [ ACC ]     STREAM     LISTENING     24893    /run/snapd-snap.socket
unix  2      [ ACC ]     STREAM     LISTENING     24895    /run/uuidd/request
unix  2      [ ACC ]     STREAM     LISTENING     16744    /run/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     16746    /run/systemd/userdb/io.systemd.DynamicUser
unix  2      [ ACC ]     STREAM     LISTENING     16747    /run/systemd/io.system.ManagedOOM
unix  2      [ ACC ]     STREAM     LISTENING     25778    /run/irqbalance/irqbalance939.sock
unix  2      [ ACC ]     STREAM     LISTENING     24617    /run/lvm/lvmpolld.socket
unix  2      [ ACC ]     STREAM     LISTENING     36980    /var/snap/lxd/common/lxd/unix.socket
unix  2      [ ACC ]     STREAM     LISTENING     24622    /run/systemd/fsck.progress
unix  2      [ ACC ]     STREAM     LISTENING     24890    @ISCSIADM_ABSTRACT_NAMESPACE
unix  2      [ ACC ]     STREAM     LISTENING     24633    /run/systemd/journal/stdout
unix  2      [ ACC ]     SEQPACKET  LISTENING     24636    /run/udev/control
unix  2      [ ACC ]     STREAM     LISTENING     134      /run/systemd/journal/io.systemd.journal
```

Рисунок 34 – Команда `netstat`

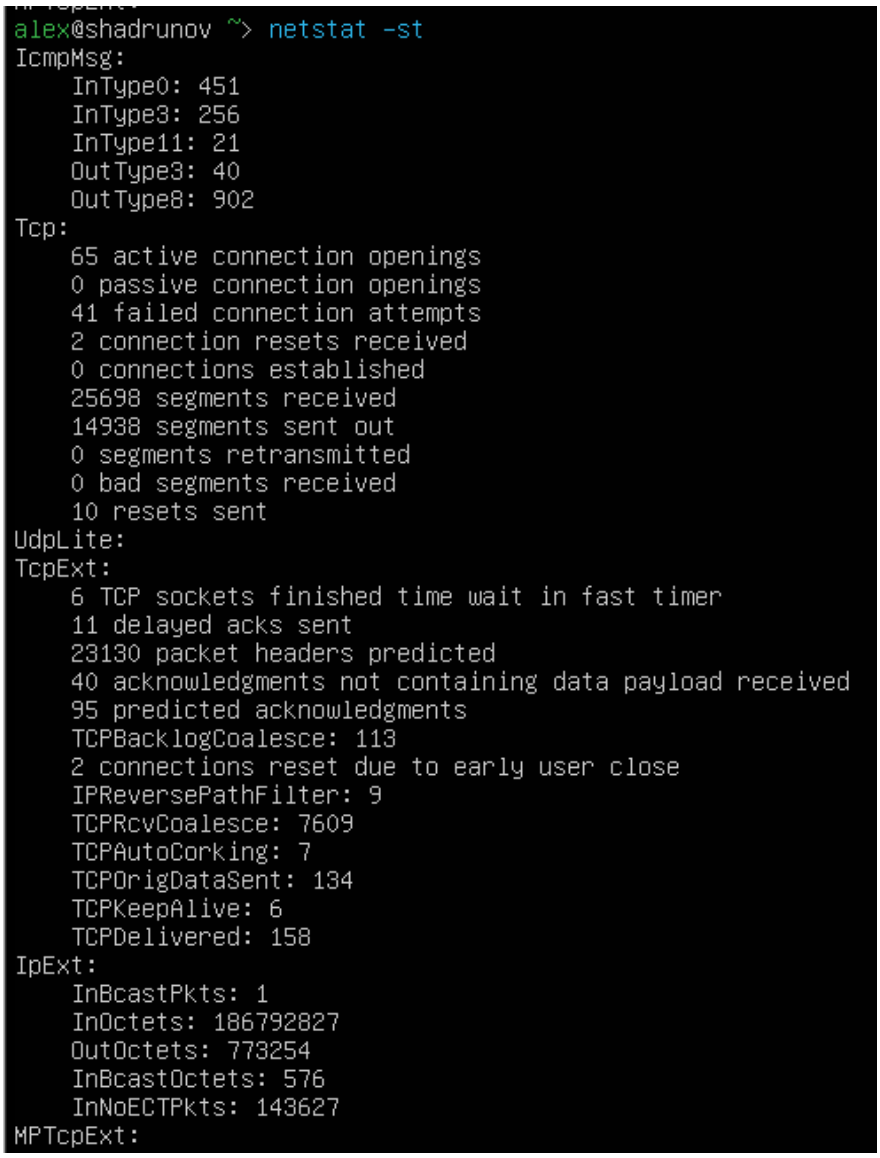

Рисунок 35 – Команда `netstat`

2.14 `nslookup`

Команда `nslookup` позволяет работать с dns-записями, запрашивать ip-адрес по имени хоста и наоборот. Примеры работы команды приведены на рисунке 36.

```
alex@shadrunov ~ [1]> nslookup 8.8.8.8
8.8.8.8.in-addr.arpa    name = dns.google.

Authoritative answers can be found from:

alex@shadrunov ~> nslookup yandex.ru
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   yandex.ru
Address: 5.255.255.55
Name:   yandex.ru
Address: 77.88.55.66
Name:   yandex.ru
Address: 77.88.55.70
Name:   yandex.ru
Address: 5.255.255.50
Name:   yandex.ru
Address: 2a02:6b8:a::a

alex@shadrunov ~> nslookup google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.150.139
Name:   google.com
Address: 142.250.150.101
Name:   google.com
Address: 142.250.150.100
Name:   google.com
Address: 142.250.150.102
Name:   google.com
Address: 142.250.150.113
Name:   google.com
Address: 142.250.150.138
Name:   google.com
Address: 2a00:1450:4010:c1c::8a
Name:   google.com
Address: 2a00:1450:4010:c1c::65
Name:   google.com
Address: 2a00:1450:4010:c1c::66
Name:   google.com
Address: 2a00:1450:4010:c1c::71
```

Рисунок 36 – Команда `nslookup`

2.15 Работа с `python`

Для работы с `python` требуется установить интерпретатор языка и менеджер пакетов `pip` (`sudo apt install python3`). После этого можно запустить скрипт на языке `python` (рисунок 37). Работа с менеджером пакетов показана на рисунке 38.

```
alex@shadrunov ~> echo "print('that is python skript!!')" > main.py
alex@shadrunov ~> python3 main.py
that is python skript!!
alex@shadrunov ~> _
```

Рисунок 37 – Команда `python`

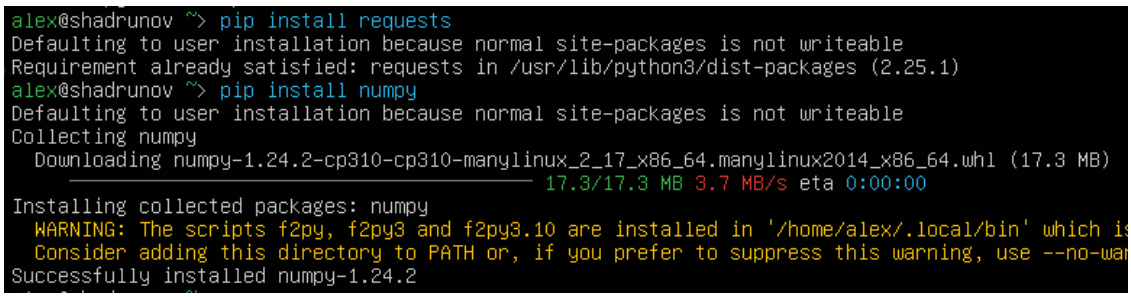

Рисунок 38 – Команда `pip`

2.16 Механизм безопасности

Один из механизмов безопасности, предусмотренный в системах `Linux`, это удалённый доступ без пароля по `ssh`. Позволяет отключить доступ к аккаунту по паролю и оставить доступ только по ключу. Это предпочтительнее, так как пароль может быть подобран злоумышленником, либо забыт владельцем, ключ имеет заданную сложность и лишён этих недостатков. На рисунке 39 показано, как передать ключ на машину, а также как получить удалённую консоль по `ssh`.

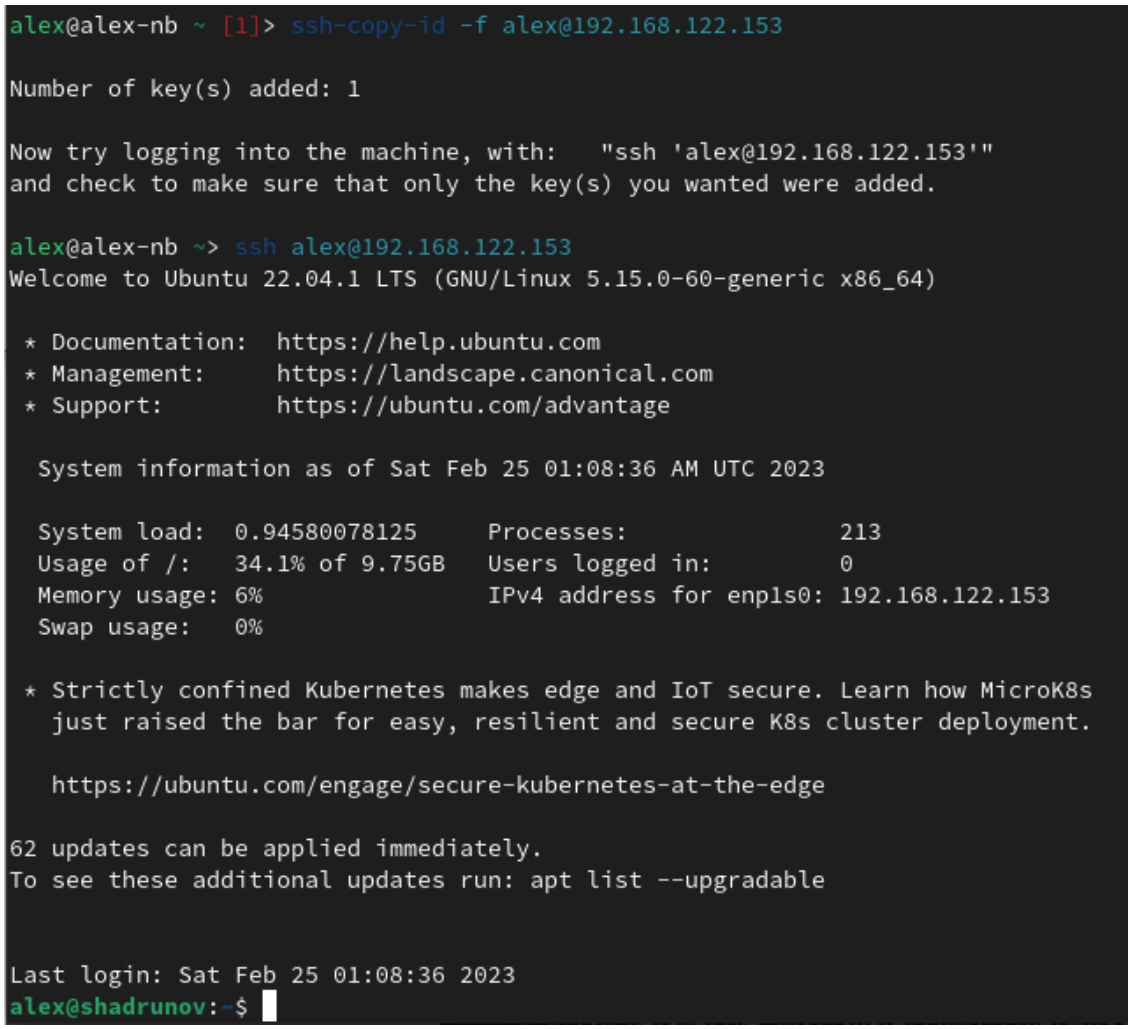

Рисунок 39 – Команда `ssh`

3 Выводы о проделанной работе

В рамках данной работы я освоил базовые навыки работы с `Linux` и основам обеспечения безопасности этих систем.