

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования



НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

## ОТЧЕТ

по лабораторной работе №2

«Настройка сетевой системы ОС Linux»

по дисциплине

«Администрирование систем и сетей»

РУКОВОДИТЕЛЬ:

\_\_\_\_\_

(подпись)

Кочешков А. А.

(фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_

(подпись)

Сухоруков В.А.

(фамилия, и.,о.)

19-ВМ

(шифр группы)

Работа защищена «\_\_» \_\_\_\_\_

С оценкой \_\_\_\_\_

Нижний Новгород 2023

# Цель работы

Изучение установки и конфигурирования сетевого интерфейса, маршрутизации и контроля сетевых связей в ОС Linux

## Ход работы

### 1. Конфигурирование сетевого интерфейса.

Спланируем работу трёх виртуальных машин Linux. Виртуальная машина ValeriiAstra будет иметь 3 сетевых интерфейса: Bridged и 2 Host-Only.

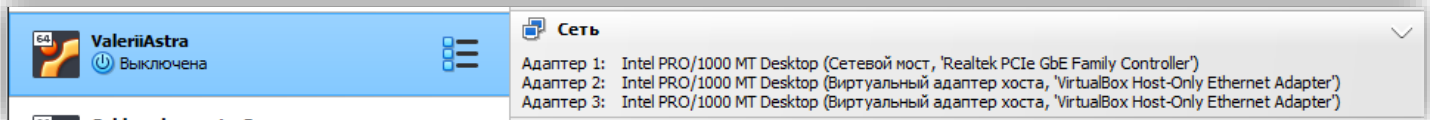


Рис 1.

Виртуальная машина SukhorukovAstra2 будет иметь один сетевой интерфейс: Host-Only.

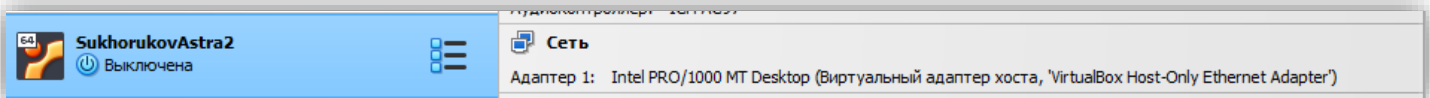


Рис 2.

Виртуальная машина Sukhorukov-astra3 будет иметь один сетевой интерфейс: Host-Only.

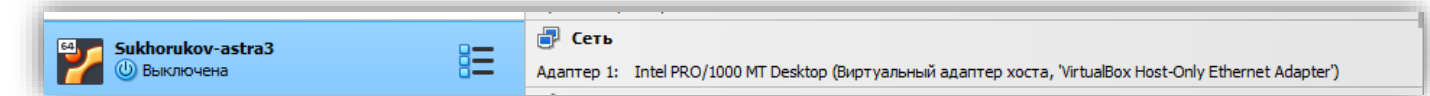


Рис 3.

С помощью команды `ifconfig` зададим параметры IP-адреса и сетевой маски. Начнем с машины ValeriiAstra.

Интерфейс `eth0`, подключенный как сетевой мост, служит для связи с основной машиной Windows и доступа к Internet. Зададим этому интерфейсу статический IP адрес 192.168.0.108, маску сети 255.255.255.0, broadcast адрес 192.168.0.255.

```
valerii@ValeriiAstra:~$ sudo ifconfig eth0 192.168.0.108 netmask 255.255.255.0 broadcast 192.168.0.255
```

Рис 4.

Проверим соединение из основной машины, умеющую адрес 192.168.0.104/24. Соединение доступно.

```
C:\Users\suxor>ping 192.168.0.108

Обмен пакетами с 192.168.0.108 по с 32 байтами данных:
Ответ от 192.168.0.108: число байт=32 время<1мс TTL=64
Ответ от 192.168.0.108: число байт=32 время<1мс TTL=64
Ответ от 192.168.0.108: число байт=32 время<1мс TTL=64
Ответ от 192.168.0.108: число байт=32 время<1мс TTL=64
```

Рис 5.

Настроим интерфейс eth1, который служит для связи с машиной SukhorukovAstra2. Зададим этому интерфейсу IP адрес 192.168.11.2, маску сети 255.255.255.128, broadcast адрес 192.168.11.127.

```
valerii@ValeriiAstra:~$ sudo ifconfig eth1 192.168.11.2 netmask 255.255.255.128 broadcast 192.168.11.127
```

Рис 6.

Настроим интерфейс eth2, который служит для связи с машиной Sukhorukov-astra3. Зададим этому интерфейсу IP адрес 192.168.11.130, маску сети 255.255.255.128, broadcast адрес 192.168.11.255.

```
valerii@ValeriiAstra:~$ sudo ifconfig eth2 192.168.11.130 netmask 255.255.255.128 broadcast 192.168.11.255
```

Рис 7.

Настроим интерфейс на машине SukhorukovAstra2. Эта машина входит в сеть 192.168.11.0/25, broadcast -192.168.11.127 с адресом 192.168.11.1.

```
valeriiastra2@SukhorukovAstra2:~$ sudo ifconfig eth0 192.168.11.1 netmask 255.255.255.128 broadcast 192.168.11.127
```

Рис 8.

Проверим соединение с машиной ValeriiAstra по адресу 192.168.11.2. Сеть доступна.

```
valeriiastra2@SukhorukovAstra2:~$ ping 192.168.11.2
PING 192.168.11.2 (192.168.11.2) 56(84) bytes of data.
64 bytes from 192.168.11.2: icmp_seq=1 ttl=64 time=0.409 ms
64 bytes from 192.168.11.2: icmp_seq=2 ttl=64 time=0.202 ms
```

Рис 9.

Настроим интерфейс на машине Sukhorukov-astra3. Эта машина входит в сеть 192.168.11.0/25, broadcast -192.168.11.255 с адресом 192.168.11.129.

```
valeriiastra3@Sukhorukov-astra3:~$ sudo ifconfig eth0 192.168.11.129 netmask 255.255.255.128 broadcast 192.168.11.255
```

Рис 10.

Проверим соединение с машиной ValeriiAstra по адресу 192.168.11.130. Сеть доступна.

```
valeriiastra3@Sukhorukov-astra3:~$ ping 192.168.11.130
PING 192.168.11.130 (192.168.11.130) 56(84) bytes of data.
64 bytes from 192.168.11.130: icmp_seq=1 ttl=64 time=0.334 ms
64 bytes from 192.168.11.130: icmp_seq=2 ttl=64 time=0.226 ms
```

Рис 11.

Проверим соединение с машиной SukhorukovAstra2 по адресу 192.168.11.1. Сеть не доступна. Эта проблема возникла из-за того, что нет настроенного пути в другую сеть.

```
valeriiastra3@Sukhorukov-astra3:~$ ping 192.168.11.1  
connect: Сеть недоступна
```

Рис 12.

Попробуем установить соединение по IP адресу, который отсутствует в сети. Было получено сообщение о недоступности узла.

```
valeriiastra3@Sukhorukov-astra3:~$ ping 192.168.11.131  
PING 192.168.11.131 (192.168.11.131) 56(84) bytes of data.  
From 192.168.11.129 icmp_seq=1 Destination Host Unreachable  
From 192.168.11.129 icmp_seq=2 Destination Host Unreachable
```

Рис 13.

В результате настроек была реализована следующая модель сетей.

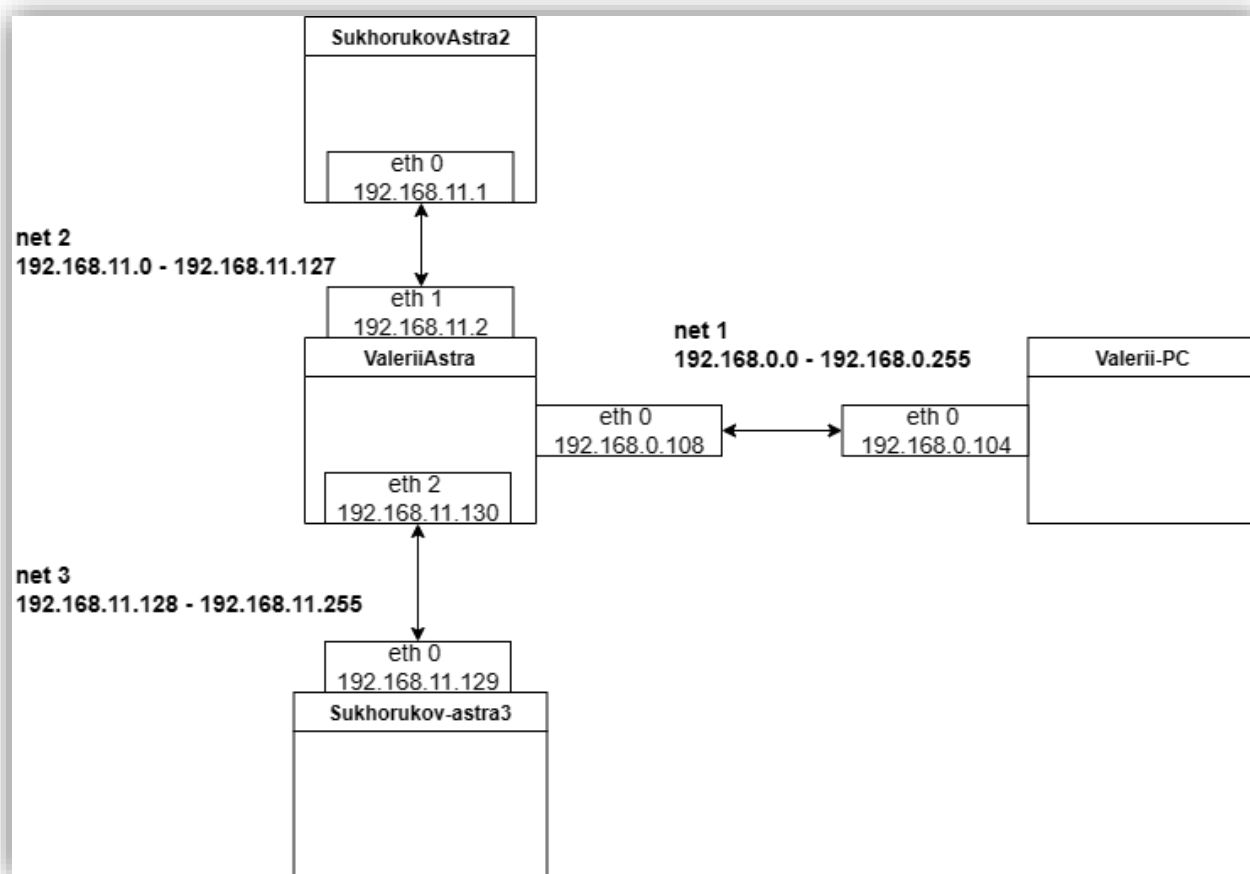


Рис 14.

## 2. Управление пространством сетевых имен.

В UNIX-сетях используется доменная система именования узлов сети DNS. В соответствии с DNS каждый компьютер имеет полное доменное имя FQDN, которое состоит из локального имени хоста и имени домена. При работе в локальной сети можно использовать сокращенное локальное имя. В системе должен быть реализован механизм разрешения (трансляции) сетевого имени хоста в IP-адрес. В Linux могут быть задействованы разные способы решения:

- ❖ Локальные файлы hosts
- ❖ Запросы к серверу DNS

- ❖ Службы NIS
- ❖ Современные системы также могут применять методы авто конфигурирования в локальных сетях.

Рассмотрим файлы конфигурации имен.

- ❖ hostname- содержит символическое имя локального узла.
- ❖ hosts- связывает хост-имена с IP-адресами. Каждому IP-адресу ставится в соответствие доменное имя.

Заполним файлы сопоставления IP- адреса и имени хоста трёх машин, причем для машины ValeriiAstra необходимо внести два разных IP-адреса, но одно и то же имя

ValeriiAstra

```
/etc/hosts
127.0.0.1    localhost
127.0.1.1    ValeriiAstra
192.168.11.1  SukhorukovAstra2
192.168.11.129 Sukhorukov-astra3
```

Рис 15.

SukhorukovAstra2

```
/etc/hosts
127.0.0.1    localhost
127.0.1.1    SukhorukovAstra2
192.168.11.2  ValeriiAstra
192.168.11.130 ValeriiAstra
192.168.11.129 Sukhorukov-astra3
```

Рис 16.

Sukhorukov-astra3

```
/etc/hosts
127.0.0.1    localhost
127.0.1.1    Sukhorukov-astra3
192.168.11.2  ValeriiAstra
192.168.11.130 ValeriiAstra
192.168.11.1  SukhorukovAstra2
```

Рис 17.

- ❖ networks- связывает имена сетей с IP адресами.

Зададим имена сетям и укажем их IP-адреса на трёх машинах.

```
/etc/networks
default      0.0.0.0
loopback     127.0.0.0
link-local   169.254.0.0
net2         192.168.11.0
net3         192.168.11.128
net1         192.168.0.0
```

Рис 18.

- ❖ hosts.allow и hosts.deny- содержат разрешение входов и отрицание доступа к услугам и хостам. Просматриваются сначала hosts.allow, затем hosts.deny. Если запись была найдена в hosts.allow, доступ предоставляется независимо от hosts.deny. Если соответствие найдено в hosts.deny, то запрос будет отклонен и связь разорвана. Если никакого соответствие не найдено вообще, запрос будет принят

❖ `host.conf`- содержит настройки для библиотеки `resolver`. `resolver`- механизм преобразования имен узлов сети в IP адреса и обратно (прямое и обратное преобразование). Данный файл должен содержать в каждой строке одно ключевое слово, за которым следует информация о соответствующей этому ключевому слову настройке.

❖ `resolv.conf`- определяет, как механизм преобразования имен узлов получает доступ к системе доменных имён

Проверим доступ к хостам по их заданным нами именам с помощью команды `ping`.

```
valerii@ValeriiAstra:~$ ping SukhorukovAstra2
PING SukhorukovAstra2 (192.168.11.1) 56(84) bytes of data.
64 bytes from SukhorukovAstra2 (192.168.11.1): icmp_seq=1 ttl=64 time=0.190 ms
64 bytes from SukhorukovAstra2 (192.168.11.1): icmp_seq=2 ttl=64 time=0.241 ms
^C
[1]+  Остановлен    ping SukhorukovAstra2
valerii@ValeriiAstra:~$ ping Sukhorukov-astra3
PING Sukhorukov-astra3 (192.168.11.129) 56(84) bytes of data.
64 bytes from Sukhorukov-astra3 (192.168.11.129): icmp_seq=1 ttl=64 time=0.195 ms
64 bytes from Sukhorukov-astra3 (192.168.11.129): icmp_seq=2 ttl=64 time=0.218 ms
^C
[2]+  Остановлен    ping Sukhorukov-astra3
```

Рис 19.

```
valeriiastra2@SukhorukovAstra2:~$ ping ValeriiAstra
PING ValeriiAstra (192.168.11.2) 56(84) bytes of data.
64 bytes from ValeriiAstra (192.168.11.2): icmp_seq=1 ttl=64 time=0.246 ms
64 bytes from ValeriiAstra (192.168.11.2): icmp_seq=2 ttl=64 time=0.238 ms
```

Рис 20.

```
valeriiastra3@Sukhorukov-astra3:~$ ping ValeriiAstra
PING ValeriiAstra (192.168.11.130) 56(84) bytes of data.
64 bytes from ValeriiAstra (192.168.11.130): icmp_seq=1 ttl=64 time=0.225
```

Рис 21.

Доступ был получен ко всем узлам, которые находятся в одной подсети.

### 3. Формирование подсетей и маршрутизация.

Рассмотрим таблицы маршрутизации, с помощью команды `route` на узле `ValeriiAstra`.

```
valerii@ValeriiAstra:~$ sudo route
Kernel IP routing table
Destination        Gateway            Genmask           Flags Metric Ref    Use Iface
default            192.168.0.1       0.0.0.0           UG    105   0      0 eth0
net1                0.0.0.0           255.255.255.0     U     105   0      0 eth0
net2                0.0.0.0           255.255.255.128   U     101   0      0 eth1
net3                0.0.0.0           255.255.255.128   U     104   0      0 eth2
```

Рис 22.

В таблице можно увидеть имя сети, шлюз, маску, флаги и интерфейс, использующийся для этой сети. На данной машине есть пути во всех трёх сетях.

Рассмотрим таблице узле с одним сетевым интерфейсом типа `host only` - `SukhorukovAstra2`.

```
valeriiastra2@SukhorukovAstra2:~$ sudo route
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
net2             0.0.0.0         255.255.255.128 U        0      0      0 eth0
```

Рис 23.

На данном узле настроен путь только в одной сети. Добавим маршруты к сетям net1 и net3.

```
valeriiastra2@SukhorukovAstra2:~$ sudo route add -net 192.168.0.0 netmask 255.255.255.0 gw 192.168.11.2 eth0
```

Рис 24.

```
valeriiastra2@SukhorukovAstra2:~$ sudo route add -net 192.168.11.128 netmask 255.255.255.128 gw 192.168.11.2 eth0
```

Рис 25.

Получим новую таблицу с помощью route.

```
valeriiastra2@SukhorukovAstra2:~$ sudo route
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
net1             ValeriiAstra    255.255.255.0    UG      0      0      0 eth0
net2             0.0.0.0         255.255.255.128 U        0      0      0 eth0
net3             ValeriiAstra    255.255.255.128 UG      0      0      0 eth0
```

Рис 26.

В столбце флагов у добавленных сетей присутствует флаг G, который означает, что используется шлюз.

Аналогично настроим пути на узле Sukhorukov-astra3. Получим таблицу сетей.

```
valeriiastra3@Sukhorukov-astra3:~$ sudo route
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
net1             ValeriiAstra    255.255.255.0    UG      0      0      0 eth0
net2             ValeriiAstra    255.255.255.0    UG      0      0      0 eth0
net3             0.0.0.0         255.255.255.128 U        0      0      0 eth0
```

Рис 27.

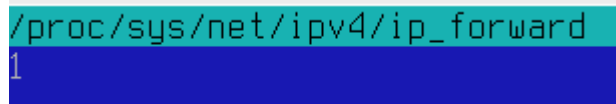
Проверим связь между узлом SukhorukovAstra2 и Sukhorukov-astra3.

```
valeriiastra2@SukhorukovAstra2:~$ ping Sukhorukov-astra3
PING Sukhorukov-astra3 (192.168.11.129) 56(84) bytes of data:
^C
--- Sukhorukov-astra3 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4075ms
```

Рис 28.

При сравнении результата, который мы получили в п.1, где узлы из разных сетей были недоступны, получаем уже другую реакцию. В данном случае узлы доступны, но

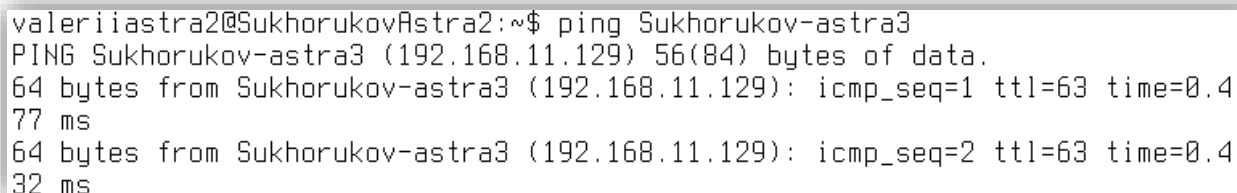
пакеты не доходят. Это происходит, потому что функция маршрутизации в ядре Linux на узле-маршрутизаторе ValeriiAstra не включена. Включим данную функцию, для этого запишем «1» в содержимое файла: /proc/sys/net/ipv4/ip\_forward.



```
/proc/sys/net/ipv4/ip_forward
1
```

Рис 29.

Пропингуем еще раз и посмотрим результат.

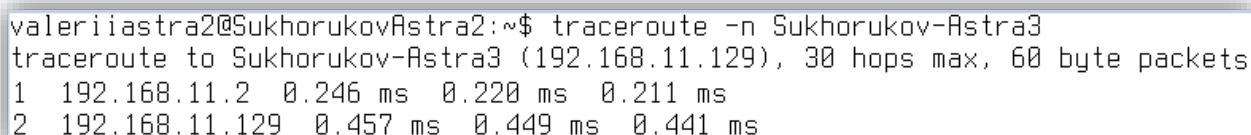


```
valeriiastra2@SukhorukovAstra2:~$ ping Sukhorukov-astra3
PING Sukhorukov-astra3 (192.168.11.129) 56(84) bytes of data.
64 bytes from Sukhorukov-astra3 (192.168.11.129): icmp_seq=1 ttl=63 time=0.477 ms
64 bytes from Sukhorukov-astra3 (192.168.11.129): icmp_seq=2 ttl=63 time=0.432 ms
```

Рис 30.

Теперь соединение доступно, и пакеты проходят.

Выполним трассировку прохождения пакетов от узла SukhorukovAstra2 до узла Sukhorukov-astra3 с помощью команды traceroute.

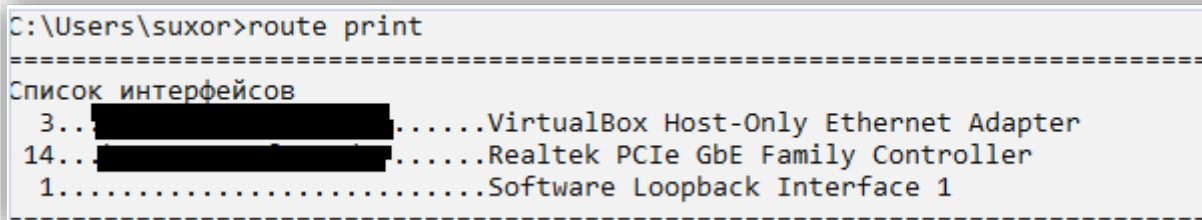


```
valeriiastra2@SukhorukovAstra2:~$ traceroute -n Sukhorukov-Astra3
traceroute to Sukhorukov-Astra3 (192.168.11.129), 30 hops max, 60 byte packets
 1  192.168.11.2  0.246 ms  0.220 ms  0.211 ms
 2  192.168.11.129  0.457 ms  0.449 ms  0.441 ms
```

Рис 31.

Сначала пакет проходит через шлюз, затем доходят до указанного узла в другой подсети. Таким образом, статическая маршрутизация между подсетями была настроена верно.

Рассмотрим таблицу маршрутизации на Windows с помощью команды route print. Сначала данная команды выводит список сетевых интерфейсов, отображая их ID, MAC-адреса и названия.



```
C:\Users\suxor>route print

=====
Список интерфейсов
 3...[REDACTED].....VirtualBox Host-Only Ethernet Adapter
14...[REDACTED].....Realtek PCIe GbE Family Controller
1.....Software Loopback Interface 1
=====
```

Рис 32.

Далее следует таблица маршрута сетей IPv4, которая схожа с таблицей в Linux. Вместо флагов в маршруте используется столбец «Метрика», который хранит числовое значение флагов.



IPv4 таблица маршрута

---

Активные маршруты:

Сетевой адрес	Маска сети	Адрес шлюза	Интерфейс	Метрика
0.0.0.0	0.0.0.0	192.168.0.1	192.168.0.104	25
127.0.0.0	255.0.0.0	On-link	127.0.0.1	331
127.0.0.1	255.255.255.255	On-link	127.0.0.1	331
127.255.255.255	255.255.255.255	On-link	127.0.0.1	331
192.168.0.0	255.255.255.0	On-link	192.168.0.104	281
192.168.0.104	255.255.255.255	On-link	192.168.0.104	281
192.168.0.255	255.255.255.255	On-link	192.168.0.104	281
192.168.56.0	255.255.255.0	On-link	192.168.56.1	281
192.168.56.1	255.255.255.255	On-link	192.168.56.1	281
192.168.56.255	255.255.255.255	On-link	192.168.56.1	281
224.0.0.0	240.0.0.0	On-link	127.0.0.1	331
224.0.0.0	240.0.0.0	On-link	192.168.56.1	281
224.0.0.0	240.0.0.0	On-link	192.168.0.104	281
255.255.255.255	255.255.255.255	On-link	127.0.0.1	331
255.255.255.255	255.255.255.255	On-link	192.168.56.1	281
255.255.255.255	255.255.255.255	On-link	192.168.0.104	281

Рис 33.

При наличии и настройке IPv6 адресации также выводится таблица маршрутизации по данному протоколу.

IPv6 таблица маршрута

---

Активные маршруты:

Метрика	Сетевой адрес	Шлюз
1	331 ::1/128	On-link
3	281 fe80::/64	On-link
14	281 fe80::/64	On-link
14	281 fe80::4787:d864:3c4b:1b4a/128	On-link
3	281 fe80::9b19:adc8:74dc:9c5f/128	On-link
1	331 ff00::/8	On-link
3	281 ff00::/8	On-link
14	281 ff00::/8	On-link

Рис 34.

#### 4. Реализовать настройку IP-сетей по внешним правилам для использования шлюза во внешние сети и Internet

Реализуем функцию трансляции адресов NAT в ядре узла ValeriiAstra с помощью команды `iptables`, которая указывает, что нужно добавить правило цепочки, относящейся к таблице NAT.

```
valerii@ValeriiAstra:~$ sudo iptables -t nat -A POSTROUTING -s 192.168.11.0/25 -o eth1 -j MASQUERADE
valerii@ValeriiAstra:~$ sudo iptables -t nat -A POSTROUTING -s 192.168.11.128/25 -o eth2 -j MASQUERADE
```

Рис 35.

Теперь настроим шлюзы по умолчанию на двух других машинах.

```
valeriiastra2@SukhorukovAstra2:~$ sudo ip route add default via 192.168.11.2
```

Рис 36.

```
valeriiastra3@Sukhorukov-astra3:~$ sudo ip route add default via 192.168.11.130
```

Рис 37.

Теперь настроим IP адреса сервера DNS в конфигурационном файле resolv.conf на хостах. Добавим туда IP адрес моего роутера 192.168.0.1

```
/etc/resolv.conf
nameserver 192.168.0.1
```

Рис 38.

Попробуем получить доступ к сайту google.com с машины SukhorukovAstra2. Доступ к Интернет ресурсам возможен.

```
valeriiastra2@SukhorukovAstra2:~$ ping google.com
PING google.com (64.233.161.139) 56(84) bytes of data:
64 bytes from lh-in-f139.1e100.net (64.233.161.139): icmp_seq=1 ttl=105 time
=28.7 ms
64 bytes from lh-in-f139.1e100.net (64.233.161.139): icmp_seq=2 ttl=105 time
=28.6 ms
```

Рис 39.

Выполним трассировку к серверу yandex.ru с хоста Sukhorukov-astra3 по протоколу TCP.

```
valeriiastra3@Sukhorukov-astra3:~$ traceroute yandex.ru
traceroute to yandex.ru (5.255.255.77), 30 hops max, 60 byte packets
 1 ValeriiAstra (192.168.11.130) 0.233 ms 0.212 ms 0.203 ms
 2 192.168.0.1 (192.168.0.1) 0.512 ms 0.654 ms 0.637 ms
 3 * * *
 4 109x194x232x238.static-business.nn.ertelecom.ru (109.194.232.238) 1.74
6 ms 109x194x232x234.static-business.nn.ertelecom.ru (109.194.232.234) 1.9
22 ms 109x194x232x238.static-business.nn.ertelecom.ru (109.194.232.238) 1.
913 ms
 5 188.234.131.242 (188.234.131.242) 7.158 ms 7.271 ms 7.234 ms
 6 yandex.w-ix.ru (193.106.112.112) 7.929 ms net131.234.188-243.ertelecom
.ru (188.234.131.243) 7.790 ms 8.577 ms
 7 * * *
 8 yandex.ru (5.255.255.77) 10.856 ms 11.158 ms *
```

Рис 40.

Сначала происходит запрос к узлу ValeriiAstra, далее к роутеру и в сеть Internet. Для достижения ресурса потребовалось 8 шагов.

Теперь выполним трассировку по протоколу ICMP. Можно увидеть, что при использовании разных протоколов маршрутизация происходит по разному.

```
valeriiastra3@Sukhorukov-astra3:~$ sudo traceroute -I yandex.ru
traceroute to yandex.ru (5.255.255.77), 30 hops max, 60 byte packets
 1 ValeriiAstra (192.168.11.130) 0.089 ms 0.129 ms 0.124 ms
 2 192.168.0.1 (192.168.0.1) 0.754 ms * *
 3 * 5x3x215x252.dynamic.nn.ertelecom.ru (5.3.215.252) 1.254 ms 1.247 ms
 4 109x194x232x234.static-business.nn.ertelecom.ru (109.194.232.234) 1.39
1 ms 1.591 ms 1.585 ms
 5 188.234.131.242 (188.234.131.242) 7.136 ms 7.335 ms 7.278 ms
 6 net131.234.188-243.ertelecom.ru (188.234.131.243) 7.240 ms 7.084 ms
7.071 ms
 7 10.7.2.1 (10.7.2.1) 13.610 ms 12.140 ms 12.302 ms
 8 yandex.ru (5.255.255.77) 9.966 ms 9.961 ms 9.936 ms
```

Рис 41.

Схема сети на канальном уровне:

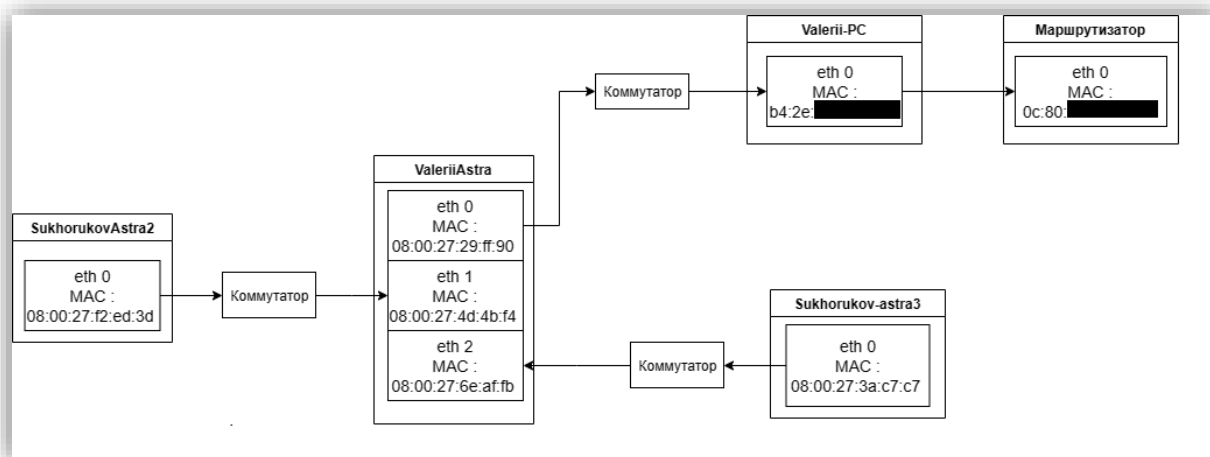


Рис 42.

Схема сети на сетевом уровне с выходом в Интернет:

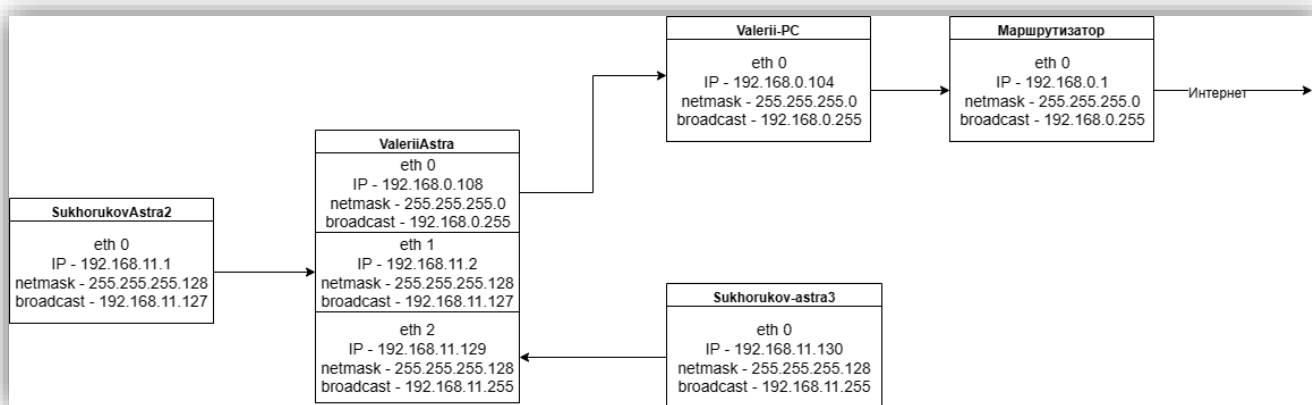


Рис 43.

## 5. Контроль за сетью, получение статистики и другой сетевой информации.

В псевдофайловой системе ядра /proc/net/ и /proc/sys/net найдем отдельные примеры текущей информации о сетевой подсистеме во внутреннем представлении:

Таблица маршрутизации (файл proc/net/route):

/proc/net/route										896/896
Interface	Destination	Gateway	Flags	RefCnt	Use	Metric	Mask	MTU	Window	IRTT
eth0	00000000	0100A8C0	0003	0	0	100	00000000	0	0	0
eth0	0000A8C0	00000000	0001	0	0	100	00FFFFFF	0	0	0
eth1	0000A8C0	00000000	0001	0	0	106	00FFFFFF	0	0	0
eth2	0000A8C0	00000000	0001	0	0	107	00FFFFFF	0	0	0
eth1	0038A8C0	00000000	0001	0	0	106	00FFFFFF	0	0	0
eth2	0038A8C0	00000000	0001	0	0	107	00FFFFFF	0	0	0

Рис 44.

Статистика сетевых интерфейсов (файл /proc/net/dev)

/proc/net/dev													693/693	1000
Interface	Receive									Transmit				
face	bytes	packets	errs	drop	fifo	frame	compressed	multicast	bytes	packets	errs	drop	fifo	colls
lo:	6327	47	0	0	0	0	0	0	6327	47	0	0	0	0
eth0:	34337349	39044	0	9	0	0	0	0	4114	901600	7559	0	0	0
eth1:	652344	4905	0	0	0	0	0	0	273	26228960	13086	0	0	0
eth2:	590652	4753	0	0	0	0	0	0	259	26955272	12824	0	0	0

Рис 45.

Используем команду `arp`, чтобы получить кэш-таблицу протокола `arp`

```
valerii@ValeriiAstra:~$ sudo arp
Address          HWtype HWaddress      Flaos Mask      Iface
Sukhorukov-astra3 ether 08:00:27:3a:c7:c7 C               eth2
SukhorukovAstra2 ether 08:00:27:f2:ed:3d C               eth1
SukhorukovAstra2 ether 08:00:27:f2:ed:3d C               eth2
192.168.0.1      ether 0c:80: [REDACTED] C               eth0
```

Рис 46.

Здесь мы можем увидеть MAC адреса всех хостов, с которыми устанавливали соединения.

Команда **netstat** предоставляет различную информацию о сетевых соединениях, портах, маршрутизации и т.д. С помощью различных опций можно получить разную статистическую информацию. Например:

- ❖ **netstat -i** выводит статистику по интерфейсам, такую как количество отправленных и принятых пакетов, ошибки и т.д.
- ❖ **netstat -r** выводит таблицу маршрутизации.
- ❖ **netstat -n** отображает IP-адреса и номера портов без их преобразования в имена хостов и служб.
- ❖ **netstat -a** отображает все активные сетевые соединения.

Команда **nstat** также предоставляет статистику сетевых подключений, но использует протокол SNMP для сбора этой информации. **nstat** позволяет получить более подробную информацию, например, статистику для каждого протокола, типа трафика, IP-адресов и т.д.

Команда **ss** также позволяет получать информацию о сетевых соединениях и маршрутизации, но в отличие от **netstat** и **nstat** предоставляет информацию о сокетах, а не о процессах, которые используют эти сокеты. **ss** также имеет более удобный и понятный формат вывода.

Как правило, использование команд **netstat**, **nstat** и **ss** зависит от конкретной задачи, которую необходимо выполнить. Команда **netstat** является стандартной утилитой и обычно доступна на большинстве операционных систем. Команда **nstat** может быть установлена дополнительно и предоставляет более подробную информацию с помощью протокола SNMP. Команда **ss** является более современной и удобной утилитой для работы с сокетами, которая может быть использована вместо **netstat**.

## Вывод

По выполнению данной лабораторной работы были созданы три виртуальные машины, которые составляли сеть из двух клиентских узлов и одного узла-маршрутизатора.

На каждой из виртуальных машин были настроены сетевые интерфейсы и таблицы маршрутизации.

С помощью таблиц маршрутизации удалось связать два узла из разных подсетей и проверить связь между ними и реализовать с узла-клиента доступ в интернет.

Также рассмотрены важные конфигурационные файлы: `hosts`, который связывает хост-имена с IP-адресами и `networks`, который связывает доменные имена с адресами сетей.