



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) Игнаков К. М.
(фамилия, и.,о.)

19-ВМ
(шифр группы)

Работа защищена « ____ » _____
С оценкой _____

Цель работы

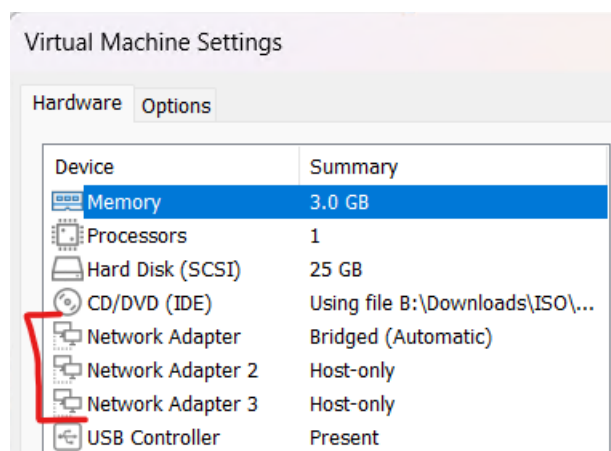
Изучение установки и конфигурирования сетевого интерфейса, маршрутизации и контроля сетевых связей в ОС Linux

Ход работы

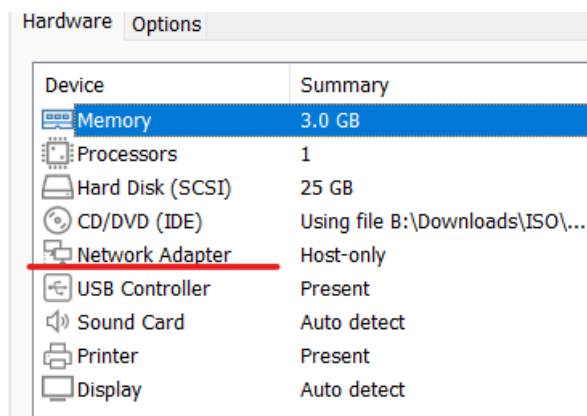
Конфигурирование сетевого интерфейса

В работе используются три виртуальные машины на основе ОС Astra Linux Орел с именами: astra, astra2-poly и astra3-poly.

Для начала работы необходимо одному узлы добавить два новых сетевых интерфейса, в итоге получим: Bridged и 2 Host-only.



На двух других установим по одному сетевому интерфейсу Host-only:



С помощью команды `ifconfig` зададим статический IP-адрес и маску для виртуальной машины astra на сетевом интерфейсе eth0, который подключен как сетевой мост и служит для связи с основной машиной, а также имеет доступ в интернет. Ip = 192.168.0.108, Netmask = 255.255.255.0, Broadcast = 192.168.0.255.

```
root@astra:/home/poly# ifconfig eth0 192.168.0.108 netmask 255.255.255.0 broadcast 192.168.0.255
```

Настроим интерфейс eth1, который будет служить для связи с виртуальной машиной astra2-poly. Ip = 192.168.1.2, Netmask = 255.255.255.128, Broadcast = 192.168.1.127.

```
root@astra:/home/poly# ifconfig eth1 192.168.1.2 netmask 255.255.255.128 broadcast 192.168.1.127
```

Выведем настройки сети для astra, увидим, что изменения успешно применились для всех сетевых интерфейсов:

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:3f:5b:ef brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.108/24 brd 192.168.0.255 scope global noprefixroute dynamic eth0
        valid_lft 85301sec preferred_lft 85301sec
    inet6 fe80::e606:f3c:9ae7:4b33/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:24:58:b4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.2/25 brd 192.168.1.127 scope global noprefixroute eth1
        valid_lft forever preferred_lft forever
    inet 192.168.246.129/24 brd 192.168.246.255 scope global noprefixroute dynamic eth1
        valid_lft 1444sec preferred_lft 1444sec
    inet6 fe80::868e:3418:ff8b:9475/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:3b:11:e7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.129/25 brd 192.168.1.255 scope global noprefixroute eth2
        valid_lft forever preferred_lft forever
    inet 192.168.246.128/24 brd 192.168.246.255 scope global noprefixroute dynamic eth2
        valid_lft 1551sec preferred_lft 1551sec
    inet6 fe80::e76:7490:c618:65f5/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
root@astra:/home/poly#
```

Для сетевого интерфейса eth2 настроим доступ для виртуальной машины astra3-poly. Ip = 192.168.1.129, Netmask = 255.255.255.128, Broadcast = 192.168.1.255.

```
root@astra:/home/poly# ifconfig eth2 192.168.1.129 netmask 255.255.255.128 broadcast 192.168.1.255
```

На второй машине astra2-poly настроим сетевой интерфейс eth0, которая будет входить в сеть 192.168.1.0/25, brd = 192.168.1.127 и Ip = 192.168.1.3.

```
root@astra2-poly:~# ifconfig eth0 192.168.1.3 netmask 255.255.255.128 broadcast 192.168.1.127
root@astra2-poly:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:23:d2:40 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.3/25 brd 192.168.1.127 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::1c68:370a:168d:34cc/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

На третьей машине установим интерфейс eth0 со следующими параметрами: Ip = 192.168.1.130/25, brd = 192.168.1.255.

```
root@astra3-poly:~# ifconfig eth0 192.168.1.130 netmask 255.255.255.128 broadcast 192.168.1.255
root@astra3-poly:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:2d:70:2b brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.130/25 brd 192.168.1.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::2934:68cd:b1b1:68c2/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Попробуем с третьей машины проверить подключение ко второй.

```
root@astra3-poly:~# ping 192.168.1.3
connect: Сеть недоступна
root@astra3-poly:~#
```

Получили сообщение, что сеть не доступна, проверим подключение между третьей и первой, которая настроена с тремя сетевыми интерфейсами.

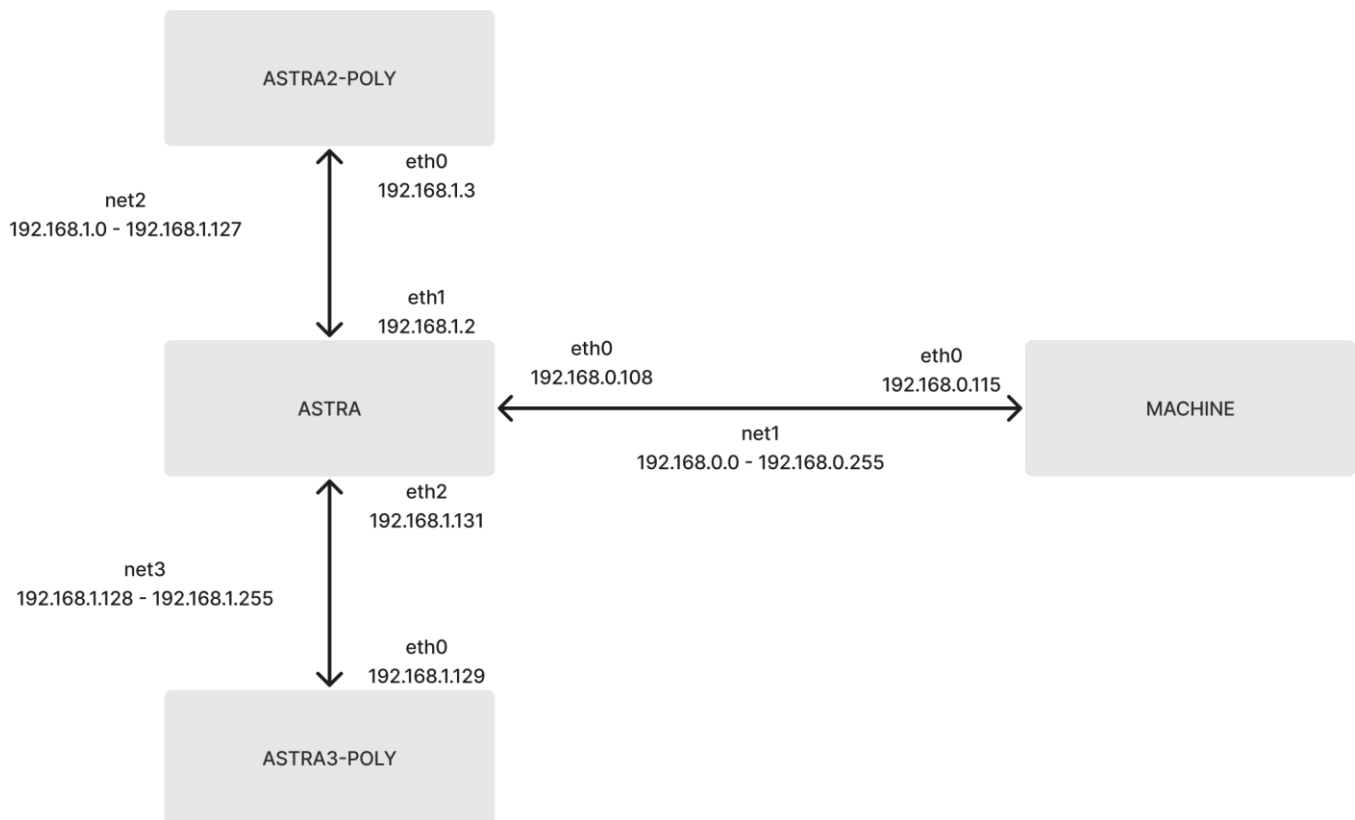
```
root@astra3-poly:~# ping 192.168.1.131
PING 192.168.1.131 (192.168.1.131) 56(84) bytes of data.
64 bytes from 192.168.1.131: icmp_seq=1 ttl=64 time=0.965 ms
64 bytes from 192.168.1.131: icmp_seq=2 ttl=64 time=0.490 ms
64 bytes from 192.168.1.131: icmp_seq=3 ttl=64 time=0.473 ms
^C
--- 192.168.1.131 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2025ms
rtt min/avg/max/mdev = 0.473/0.642/0.965/0.229 ms
```

Попробуем со второй машины установить соединение по адресу, которого не существует в сети.

```
root@astra2-poly:~# ping 192.168.1.55
PING 192.168.1.55 (192.168.1.55) 56(84) bytes of data.
From 192.168.1.3 icmp_seq=1 Destination Host Unreachable
From 192.168.1.3 icmp_seq=2 Destination Host Unreachable
From 192.168.1.3 icmp_seq=3 Destination Host Unreachable
^C
--- 192.168.1.55 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time 4088ms
pipe 4
```

Получили сообщение, что хост не доступен в этой сети.

Составим схему сети после проделанных настроек.



Управление пространством сетевых имен

В сетях UNIX используется система именования узлов DNS, которая предоставляет полное доменное имя FQDN для каждого компьютера, состоящее из локального имени хоста и имени домена. В локальной сети можно использовать сокращенное локальное имя. Для преобразования сетевого имени хоста в IP-адрес должен быть настроен механизм разрешения (трансляции).

В локальной сети для разрешения сетевого имени хоста в IP-адрес могут быть использованы следующие решения:

- **Файл /etc/hosts:** это локальный текстовый файл, в котором можно определить связь между именем хоста и его IP-адресом. Этот файл расположен на каждом компьютере в локальной сети и позволяет решать имена хостов без использования DNS.
- **DNS-сервер:** это сервер, который обрабатывает DNS-запросы и возвращает соответствующие IP-адреса. Он используется для разрешения сетевых имён хостов в IP-адреса в локальной и глобальной сети. В локальной сети обычно устанавливается DNS-сервер, который хранит информацию о всех компьютерах в локальной сети.
- **NetBIOS-имена:** это метод именования компьютеров в сети, который используется в Microsoft Windows. Это позволяет обращаться к компьютеру по его имени NetBIOS вместо IP-адреса. Для разрешения

NetBIOS-имён в IP-адрес используется служба WINS (Windows Internet Name Service), которая работает на сервере.

- mDNS (Multicast DNS): это протокол, позволяющий компьютерам в локальной сети автоматически обнаруживать друг друга и разрешать имена без использования централизованного DNS-сервера. Компьютеры в локальной сети отправляют широковещательные запросы для разрешения имен хостов.

Для нашей задачи воспользуемся первым вариантом и рассмотрим файлы `hosts` и `hostname`, которые расположены в директории `/etc/`.

В Linux для конфигурации компьютера в сети используется несколько файлов. Некоторые из них:

`/etc/hosts` - содержит таблицу соответствия между IP-адресами и именами хостов в локальной сети. Этот файл используется для быстрого разрешения имен хостов без обращения к DNS-серверу.

`/etc/hostname` - содержит имя хоста (hostname) системы, которое используется при загрузке системы и для локальной сети.

`/etc/resolv.conf` - определяет параметры настройки DNS-сервера, который используется для разрешения имен хостов в IP-адреса.

`/etc/hosts.allow` и `/etc/hosts.deny` - содержат список правил доступа для различных сетевых служб, таких как SSH, FTP, SMTP и т.д. Они позволяют настраивать доступ к сетевым сервисам на основе адреса IP или имени хоста.

`/etc/host.conf` - определяет порядок разрешения имен хостов в системе. Он содержит параметры, такие как "order", "multi", "spoof" и другие, которые контролируют использование локальных таблиц и DNS-серверов при разрешении имен хостов.

`/etc/resolv.conf` - содержит информацию о DNS-серверах, которые используются для разрешения имен в IP-адреса в локальной сети. Первый указанный DNS-сервер будет использован для разрешения имен в IP-адреса, если он не отвечает, то будет использован второй указанный DNS-сервер.

`/etc/networks` в Linux используется для определения сетей и связанных с ними атрибутов, таких как имя сети и IP-диапазон. Этот файл используется в различных сетевых утилитах, включая команду `route`, для определения маршрутов и сопоставления имен сетей с IP-адресами. Он может быть использован для определения различных сетевых параметров в многопользовательской среде.

Пропишем настройки во всех узлах.

```
default      0.0.0.0
loopback     127.0.0.0
link-local   169.254.0.0
net1         192.168.0.0
net2         192.168.1.0
net3         192.168.1.128
```

В файл /etc/hosts добавим имя и IP адреса “соседних” узлов.

ASTRA

```
GNU nano 2.7.4
127.0.0.1      localhost
127.0.1.1      astra
# --- new ---
192.168.1.3    astra2-poly
192.168.1.130 astra3-poly
```

ASTRA2-POLY

```
127.0.0.1      localhost
127.0.1.1      astra2-poly

192.168.1.2     astra
192.168.1.129  astra
192.168.1.130  astra3-poly
```

ASTRA3-POLY

```
127.0.0.1      localhost
127.0.1.1      astra3-poly

192.168.1.2     astra
192.168.1.129  astra
192.168.1.3     astra2-poly
```

Проверим доступ к хостам по их имени.

```
root@astra3-poly:~# ping astra
PING astra (192.168.1.129) 56(84) bytes of data:
64 bytes from astra (192.168.1.129): icmp_seq=1 ttl=64 time=0.836 ms
64 bytes from astra (192.168.1.129): icmp_seq=2 ttl=64 time=0.378 ms
64 bytes from astra (192.168.1.129): icmp_seq=3 ttl=64 time=0.327 ms
64 bytes from astra (192.168.1.129): icmp_seq=4 ttl=64 time=0.450 ms
64 bytes from astra (192.168.1.129): icmp_seq=5 ttl=64 time=0.492 ms
64 bytes from astra (192.168.1.129): icmp_seq=6 ttl=64 time=0.419 ms
64 bytes from astra (192.168.1.129): icmp_seq=7 ttl=64 time=0.353 ms
64 bytes from astra (192.168.1.129): icmp_seq=8 ttl=64 time=0.426 ms
^C
```

```
root@astra2-poly:~# ping astra
PING astra (192.168.1.2) 56(84) bytes of data:
64 bytes from astra (192.168.1.2): icmp_seq=1 ttl=64 time=0.837 ms
64 bytes from astra (192.168.1.2): icmp_seq=2 ttl=64 time=0.444 ms
64 bytes from astra (192.168.1.2): icmp_seq=3 ttl=64 time=0.629 ms
64 bytes from astra (192.168.1.2): icmp_seq=4 ttl=64 time=0.491 ms
64 bytes from astra (192.168.1.2): icmp_seq=5 ttl=64 time=0.462 ms
```

Доступ между узлами, находящимися в одной сети, успешно установлено.

Формирование подсетей и маршрутизация.

Команда `route` в Linux используется для просмотра и управления таблицами маршрутизации IP-пакетов в системе. Таблица маршрутизации содержит информацию о маршрутах, которые используются для пересылки IP-пакетов между различными сетевыми интерфейсами и хостами в сети.

Команда `route` без аргументов выводит текущую таблицу маршрутизации. С помощью различных опций, таких как `add`, `del` и `change`, можно добавлять, удалять и изменять маршруты в таблице маршрутизации. Например, с помощью команды `route add` можно добавить новый маршрут в таблицу маршрутизации, указав IP-адрес целевой сети, маску подсети и шлюз по умолчанию для доставки пакетов до этой сети.

Кроме того, команда `route` позволяет настраивать множество других параметров, связанных с маршрутизацией, таких как максимальное количество переходов до целевого хоста (таймауты), предпочтительный интерфейс для исходящих пакетов и т.д.

Посмотрим таблицу маршрутизации на узле ASTRA.


```

root@astra:~# route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
default        192.168.0.1     0.0.0.0         UG    100    0      0 eth0
net1            0.0.0.0         255.255.255.0   U     100    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     102    0      0 eth2

```

Таблица маршрутизации машины ASTRA2-POLY.

```

root@astra2-poly:~# route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
net2            0.0.0.0         255.255.255.128 U     100    0      0 eth0

```

Добавим маршруты к сетям net1 и net3 и посмотрим результат.

```

root@astra2-poly:~# route add -net 192.168.0.0 netmask 255.255.255.0 gw 192.168.1.2 eth0
root@astra2-poly:~# route add -net 192.168.1.128 netmask 255.255.255.128 gw 192.168.1.2 eth0
root@astra2-poly:~# route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
net1            astra           255.255.255.0   UG    0      0      0 eth0
net2            0.0.0.0         255.255.255.128 U     100    0      0 eth0
net3            astra           255.255.255.128 UG    0      0      0 eth0

```

В столбце Flags значение U обозначает, что он включен, а G — используется шлюз.

Аналогично настроим таблицу маршрутизации для машины ASTRA3-POLY.

```

root@astra3-poly:~# route add -net 192.168.0.0 netmask 255.255.255.0 gw 192.168.1.129 eth0
root@astra3-poly:~# route add -net 192.168.1.128 netmask 255.255.255.128 gw 192.168.1.129 eth0
root@astra3-poly:~# route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
net1            astra           255.255.255.0   UG    0      0      0 eth0
net3            astra           255.255.255.128 UG    0      0      0 eth0
net3            0.0.0.0         255.255.255.128 U     100    0      0 eth0

```

При попытке установить соединение между net2 и net3 возникнет “зависание” команды ping, это происходит из-за того, что пакеты не доходят до узла. Пример попытки установки соединения между третьей и второй машиной.

```

root@astra2-poly:~# ping astra3-poly
PING astra3-poly (192.168.1.130) 56(84) bytes of data.
^C
--- astra3-poly ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4092ms

```

Это происходит потому, что на машине ASTRA отключена пересылка пакетов между сетевыми интерфейсами. Для того чтобы включить его в файл /proc/sys/net/ipv4/ip_forward необходимо прописать значение 1.

```
GNU nano 2.7.4                               Файл: /proc/sys/net/ipv4/ip_forward
1
```

Попробуем еще раз установить соединение между ASTRA2-POLY и ASTRA3-POLY.

```
root@astra2-poly:~# ping astra3-poly
PING astra3-poly (192.168.1.130) 56(84) bytes of data.
64 bytes from astra3-poly (192.168.1.130): icmp_seq=1 ttl=63 time=0.885 ms
64 bytes from astra3-poly (192.168.1.130): icmp_seq=2 ttl=63 time=0.847 ms
64 bytes from astra3-poly (192.168.1.130): icmp_seq=3 ttl=63 time=0.923 ms
^C
--- astra3-poly ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/mdev = 0.847/0.885/0.923/0.031 ms
```

```
root@astra3-poly:~# ping astra2-poly
PING astra2-poly (192.168.1.3) 56(84) bytes of data.
64 bytes from astra2-poly (192.168.1.3): icmp_seq=1 ttl=63 time=1.14 ms
64 bytes from astra2-poly (192.168.1.3): icmp_seq=2 ttl=63 time=0.893 ms
64 bytes from astra2-poly (192.168.1.3): icmp_seq=3 ttl=63 time=0.806 ms
64 bytes from astra2-poly (192.168.1.3): icmp_seq=4 ttl=63 time=0.853 ms
64 bytes from astra2-poly (192.168.1.3): icmp_seq=5 ttl=63 time=0.886 ms
64 bytes from astra2-poly (192.168.1.3): icmp_seq=6 ttl=63 time=0.822 ms
^C
--- astra2-poly ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5104ms
rtt min/avg/max/mdev = 0.806/0.901/1.146/0.113 ms
```

Соединение прошло успешно, узлы “видят” друг друга. Как это работает? Сначала пакет проходит через шлюз ASTRA, затем пакет идет в другую сеть, то есть ASTRA3-POLY -> ASTRA -> ASTRA2-POLY и обратно.

Рассмотрим таблицу маршрутизации на домашней системе Windows с помощью команды `route print`. В которой отображается ID, MAC адрес и название интерфейса.

```
PowerShell 7.3.3
PS C:\Users\ignak> route print
=====
Список интерфейсов
10... ..Intel(R) Ethernet Controller (3) I225-V
9... ..TAP-Windows Adapter V9 #2
20... ..TAP-Windows Adapter V9
6... ..Intel(R) Wi-Fi 6 AX200 160MHz
3... ..Microsoft Wi-Fi Direct Virtual Adapter
5... ..Microsoft Wi-Fi Direct Virtual Adapter #2
13... ..VMware Virtual Ethernet Adapter for VMnet1
24... ..VMware Virtual Ethernet Adapter for VMnet8
1.....Software Loopback Interface 1
=====
```

Далее в вывод этой команды отображается таблица маршрута IPv4 сети, в “Метрике” хранится 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	192.168.0.115	25	
10.3.0.20	255.255.255.255	On-link	127.0.0.1	76	
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	192.168.0.115	281	
192.168.0.115	255.255.255.255	On-link	192.168.0.115	281	
192.168.0.255	255.255.255.255	On-link	192.168.0.115	281	
192.168.240.0	255.255.255.0	On-link	192.168.240.1	291	
192.168.240.1	255.255.255.255	On-link	192.168.240.1	291	
192.168.240.255	255.255.255.255	On-link	192.168.240.1	291	
192.168.246.0	255.255.255.0	On-link	192.168.246.1	291	
192.168.246.1	255.255.255.255	On-link	192.168.246.1	291	
192.168.246.255	255.255.255.255	On-link	192.168.246.1	291	
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	192.168.246.1	291	
224.0.0.0	240.0.0.0	On-link	192.168.240.1	291	
224.0.0.0	240.0.0.0	On-link	192.168.0.115	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	192.168.246.1	291	
255.255.255.255	255.255.255.255	On-link	192.168.240.1	291	
255.255.255.255	255.255.255.255	On-link	192.168.0.115	281	
=====					

При настройке IPv6 маршрутизации будет так же отображено в выводе этой команды.

```
IPv6 таблица маршрута
=====
Активные маршруты:
Метрика    Сетевой адрес                Шлюз
  1        331 ::1/128                  On-link
 13        291 fe80::/64             On-link
 24        291 fe80::/64             On-link
 10        281 fe80::/64             On-link
 24        291 fe80::3156:6789:b4c9:e4be/128
                                     On-link
 13        291 fe80::a5fa:f8cb:fe9:b76/128
                                     On-link
 10        281 fe80::e76f:5420:f4c5:75e7/128
                                     On-link
  1        331 ff00::/8              On-link
 13        291 ff00::/8              On-link
 24        291 ff00::/8              On-link
 10        281 ff00::/8              On-link
=====
```

Реализовать настройку IP-сетей по внешним правилам для использования шлюза во внешней сети и Internet

Выполним следующую команду для трансляции адресов NAT на машине ASTRA.

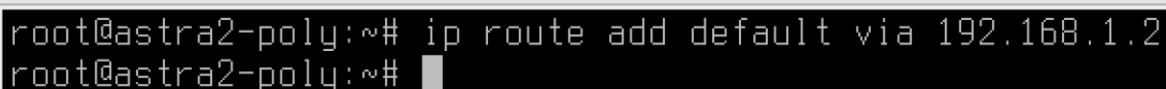
```
root@astra:~# iptables -t nat -A POSTROUTING -s 192.168.1.0/25 -o eth1 -j MASQUERADE
root@astra:~# iptables -t nat -A POSTROUTING -s 192.168.1.128/25 -o eth2 -j MASQUERADE
root@astra:~#
```

Эта команда добавляет правило в таблицу NAT (-t nat), цепочку POSTROUTING (-A POSTROUTING), которая будет применяться к пакетам, исходящим из подсети 192.168.1.0/25 (-s 192.168.1.0/25), и направляемым на интерфейс eth1 (-o eth1).

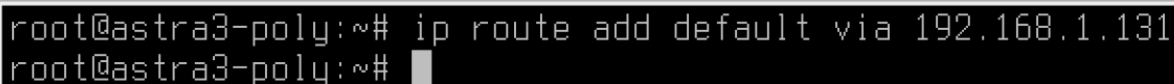
Действие, которое применяется к таким пакетам - MASQUERADE (-j MASQUERADE), которое заменяет исходный адрес и порт на адрес и порт, соответствующий выходному интерфейсу eth1. Это необходимо, чтобы пакеты могли правильно проходить через NAT-шлюз и выходить в интернет или другую сеть с адресом, соответствующим интерфейсу eth1.

Таким образом, данная команда настраивает правило NAT для маршрутизатора, который используется для переадресации пакетов, проходящих через сеть, и позволяет подключенным устройствам в подсети 192.168.1.0/25 получать доступ к другим сетям через интерфейс eth1.

Далее нужно настроить шлюз по умолчанию на других машинах.

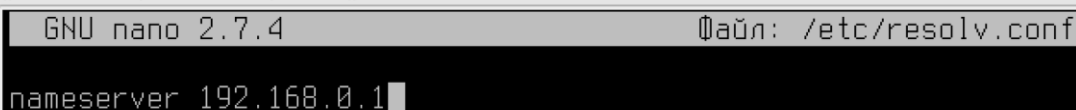


```
root@astra2-poly:~# ip route add default via 192.168.1.2
root@astra2-poly:~#
```



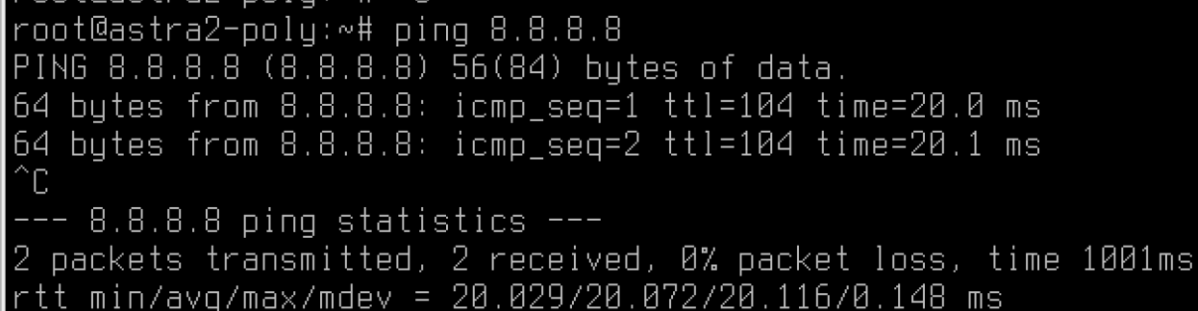
```
root@astra3-poly:~# ip route add default via 192.168.1.131
root@astra3-poly:~#
```

Теперь необходимо настроить DNS сервер в конфигурационном файле всех машин. В моем случае это 192.168.0.1 – IP адрес роутера.



```
GNU nano 2.7.4                               Файл: /etc/resolv.conf
nameserver 192.168.0.1
```

Попробуем установить соединение с DNS сервером Google. Как видим доступ в интернет есть.



```
root@astra2-poly:~# ping 8.8.8.8
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=104 time=20.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=104 time=20.1 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 20.029/20.072/20.116/0.148 ms
```

Попробуем установить соединение по имени домена.

```
root@astra3-poly:~# ping google.com
PING google.com (209.85.233.100) 56(84) bytes of data:
64 bytes from lr-in-f100.1e100.net (209.85.233.100): icmp_seq=1 ttl=104 time=24.8 ms
64 bytes from lr-in-f100.1e100.net (209.85.233.100): icmp_seq=2 ttl=104 time=22.6 ms
64 bytes from lr-in-f100.1e100.net (209.85.233.100): icmp_seq=3 ttl=104 time=22.6 ms
64 bytes from lr-in-f100.1e100.net (209.85.233.100): icmp_seq=4 ttl=104 time=22.5 ms
64 bytes from lr-in-f100.1e100.net (209.85.233.100): icmp_seq=5 ttl=104 time=22.4 ms
64 bytes from lr-in-f100.1e100.net (209.85.233.100): icmp_seq=6 ttl=104 time=22.5 ms
^C
--- google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 500ms
rtt min/avg/max/mdev = 22.451/22.951/24.831/0.866 ms
```

Для того чтобы узнать куда идет пакет во время установки соединения выполним команду traceroute, который по умолчанию пытается установить TCP-соединение.

```
root@astra3-poly:~# traceroute nntu.ru
traceroute to nntu.ru (213.177.120.42), 30 hops max, 60 byte packets
 1 astra (192.168.1.131) 0.373 ms 0.267 ms 0.284 ms
 2 192.168.0.1 (192.168.0.1) 0.668 ms 1.236 ms 1.221 ms
 3 100.79.0.1 (100.79.0.1) 2.205 ms 2.205 ms 2.328 ms
 4 10.1.139.5 (10.1.139.5) 1.650 ms 1.908 ms 2.027 ms
 5 10.1.139.17 (10.1.139.17) 2.210 ms 2.301 ms 2.219 ms
 6 10.0.22.163 (10.0.22.163) 2.202 ms 1.804 ms 1.705 ms
 7 10.17.0.154 (10.17.0.154) 1.554 ms 1.535 ms 1.767 ms
 8 136.169.213.6.dynamic.ufanet.ru (136.169.213.6) 1.669 ms 1.651 ms 1.732 ms
 9 Elita.local.elita-mineral.ru (37.29.111.142) 2.189 ms 2.104 ms 2.378 ms
10 10.222.152.113 (10.222.152.113) 15.892 ms 15.898 ms 16.140 ms
11 83.169.204.176 (83.169.204.176) 10.007 ms 9.924 ms 9.097 ms
12 188.128.126.149 (188.128.126.149) 9.830 ms 8.703 ms 9.646 ms
13 109.172.24.193 (109.172.24.193) 8.685 ms 9.384 ms 8.798 ms
14 asbr2-nnov.vt.ru (79.126.123.2) 9.830 ms 9.883 ms 9.908 ms
15 * * *
16 213.177.120.42 (213.177.120.42) 9.101 ms 8.922 ms 10.277 ms
```

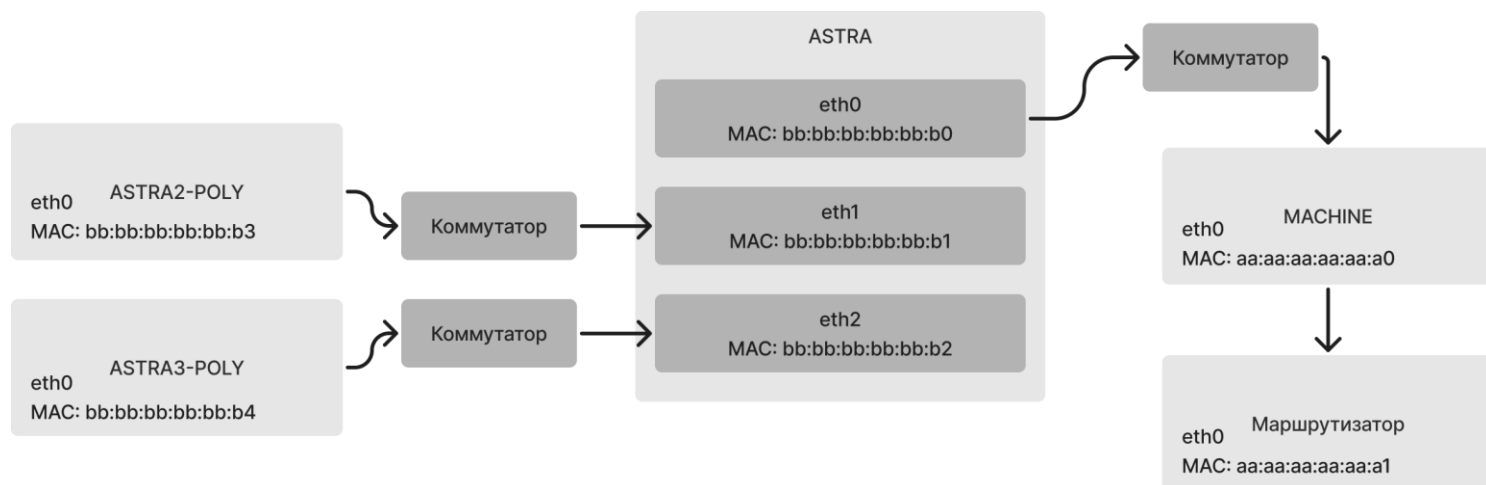
Видим, что пакет идет сначала на узел ASTRA, затем на роутер, а затем в интернет. Для достижения ресурса потребовалось 16 шагов.

Попробуем выполнить соединение по протоколу iCMP, для этого вызовем команду traceroute с флагом -I (заглавная i).

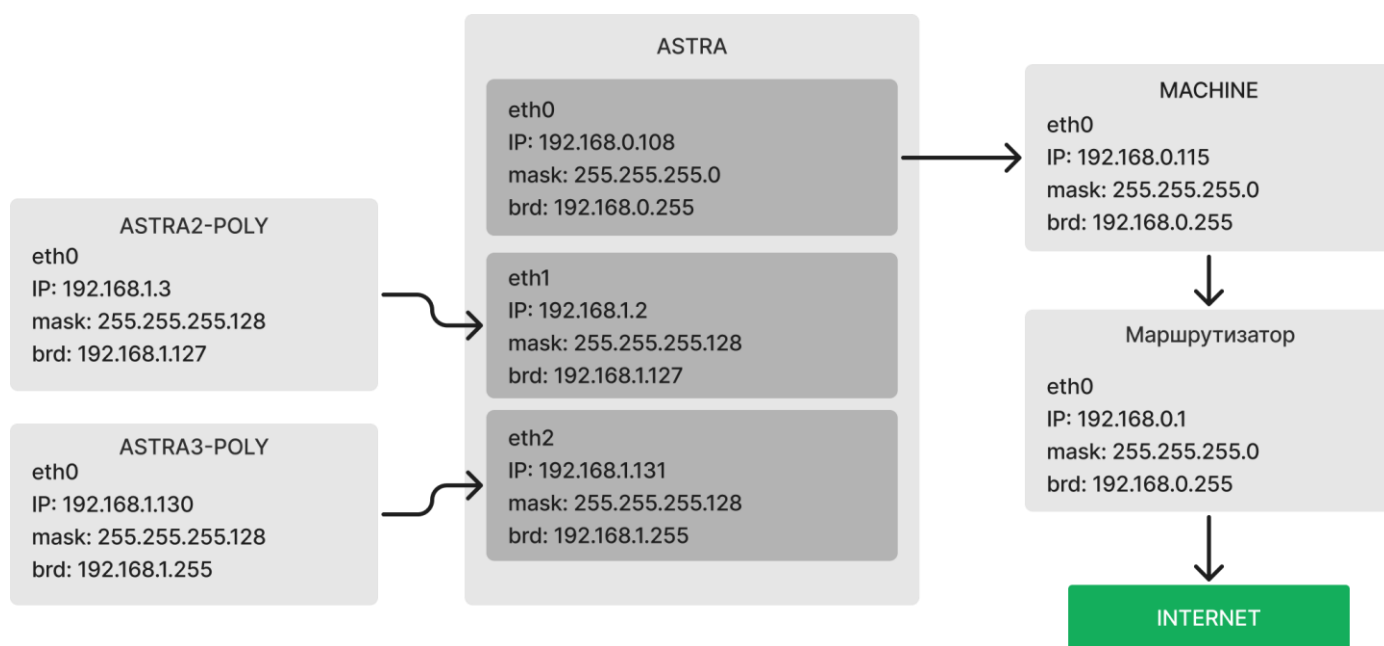
```
root@astra3-poly:~# traceroute -I nntu.ru
traceroute to nntu.ru (213.177.120.42), 30 hops max, 60 byte packets
 1 astra (192.168.1.131) 0.290 ms 0.202 ms 0.217 ms
 2 192.168.0.1 (192.168.0.1) 1.696 ms * 1.546 ms
 3 100.79.0.1 (100.79.0.1) 17.634 ms 17.560 ms 17.486 ms
 4 10.1.139.5 (10.1.139.5) 2.087 ms 2.009 ms 2.001 ms
 5 10.1.139.17 (10.1.139.17) 17.308 ms 17.398 ms 17.397 ms
 6 10.0.22.163 (10.0.22.163) 2.026 ms 1.908 ms 2.001 ms
 7 10.17.0.146 (10.17.0.146) 1.925 ms 1.246 ms 1.431 ms
 8 136.169.213.6.dynamic.ufanet.ru (136.169.213.6) 1.747 ms 1.921 ms 1.846 ms
 9 Elita.local.elita-mineral.ru (37.29.111.142) 1.772 ms 1.805 ms 1.869 ms
10 10.222.152.113 (10.222.152.113) 9.609 ms 9.448 ms 9.463 ms
11 83.169.204.166 (83.169.204.166) 9.059 ms 8.778 ms 9.111 ms
12 188.128.126.149 (188.128.126.149) 9.906 ms 9.827 ms 9.744 ms
13 109.172.24.193 (109.172.24.193) 9.996 ms 9.714 ms 9.634 ms
14 asbr2-nnov.vt.ru (79.126.123.2) 17.615 ms 17.533 ms 17.525 ms
15 * * *
16 213.177.120.42 (213.177.120.42) 10.294 ms 10.414 ms 10.323 ms
```

Нарисуем схему сети на канальном и сетевом уровне, для безопасности фактические MAC-адреса не были отображены.

Канальный.



Сетевой.



Контроль за сетью, получение статистики и другой сетевой информации.

Файл `/proc/net/route` является интерфейсом для доступа к таблице маршрутизации ядра Linux. Он содержит информацию о сетевых интерфейсах, маршрутах и шлюзах по умолчанию. Каждая строка в файле представляет собой запись таблицы маршрутизации для определенного сетевого интерфейса.

Структура записи, следующая:

- Имя сетевого интерфейса;

- Значение маркера;
- IP-адрес шлюза;
- IP-адрес сети;
- Флаги;
- Счетчики.

```
root@astra:~# cat /proc/net/route
```

Iface	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	0001A8C0	00000000	0001	0	0	101	80FFFFFF	0	0	0
eth2	0001A8C0	00000000	0001	0	0	102	00FFFFFF	0	0	0
eth1	00F6A8C0	00000000	0001	0	0	101	00FFFFFF	0	0	0
eth2	00F6A8C0	00000000	0001	0	0	102	00FFFFFF	0	0	0
eth2	80F6A8C0	00000000	0001	0	0	102	80FFFFFF	0	0	0

Файл `/proc/net/dev` в Linux предоставляет информацию о сетевых интерфейсах и статистику сетевого трафика.

В этом файле каждый сетевой интерфейс на сервере отображается в отдельной строке, в которой перечислены различные параметры, такие как количество принятых и отправленных пакетов, количество ошибок, количество отброшенных пакетов и т. д. Кроме того, эти параметры могут быть разделены на два столбца: "rx" для полученных пакетов и "tx" для отправленных пакетов.

```
root@astra:~# cat /proc/net/dev
```

Inter-	Receive									Transmit							
face	bytes	packets	errs	drop	fifo	frame	compressed	multicast	bytes	packets	errs	drop	fifo	colls	carrier	compress	
lo:	5740	68	0	0	0	0	0	0	5740	68	0	0	0	0	0	0	
eth0:	446551	1947	0	0	0	0	0	0	34575	395	0	0	0	0	0	0	
eth1:	93240	633	0	0	0	0	0	0	18443	159	0	0	0	0	0	0	
eth2:	96795	699	0	0	0	0	0	0	48666	479	0	0	0	0	0	0	

Используем команду `arp` для получения кэш-таблицы протокола.

```
root@astra:~# arp -n
```

Address	Hwtype	Hwaddress	Flags	Mask	Iface
192.168.1.130	ether	00:0C:29:1A:3D:00	C		eth2
192.168.1.3	ether	00:0C:29:1A:3D:00	C		eth1
192.168.0.1	ether	28:00:00:00:00:00	C		eth0

Команда `netstat` в Linux используется для отображения различной информации о сетевых соединениях, маршрутизации и сетевых интерфейсах на компьютере.

Она может принимать следующие аргументы:

- -a - отображение всех соединений (включая неактивные);

```
root@astra:~# netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:ipp           0.0.0.0:*               LISTEN
tcp6       0      0 localhost:ipp           ::::*                  LISTEN
udp        0      0 0.0.0.0:45026           0.0.0.0:*
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
udp        0      0 localhost:xdmcp         0.0.0.0:*
udp        0      0 0.0.0.0:mdns            0.0.0.0:*
udp6       0      0 ::::mdns                ::::*
udp6       0      0 ::::56576               ::::*
raw6       0      0 ::::ipv6-icmp           ::::*                  7
raw6       0      0 ::::ipv6-icmp           ::::*                  7
raw6       0      0 ::::ipv6-icmp           ::::*                  7

Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State      I-Node  Path
unix   2      [ ]        DGRAM      33661     /run/user/1000/systemd/notify
unix   2      [ ]        DGRAM      33095     /run/user/999/systemd/notify
unix   2      [ ACC ]    STREAM     LISTENING  33664     /run/user/1000/systemd/private
unix   2      [ ACC ]    STREAM     LISTENING  33098     /run/user/999/systemd/private
unix   2      [ ACC ]    STREAM     LISTENING  33668     /run/user/1000/gnupg/S.gpg-agent
unix   2      [ ACC ]    STREAM     LISTENING  33102     /run/user/999/gnupg/S.gpg-agent.extra
unix   2      [ ACC ]    STREAM     LISTENING  33669     /run/user/1000/gnupg/S.dirmngr
unix   2      [ ACC ]    STREAM     LISTENING  33103     /run/user/999/gnupg/S.gpg-agent.ssh
unix   2      [ ACC ]    STREAM     LISTENING  33670     /run/user/1000/gnupg/S.gpg-agent.ssh
unix   2      [ ACC ]    STREAM     LISTENING  33104     /run/user/999/gnupg/S.dirmngr
unix   2      [ ACC ]    STREAM     LISTENING  33671     /run/user/1000/gnupg/S.gpg-agent.browser
unix   2      [ ACC ]    STREAM     LISTENING  33105     /run/user/999/gnupg/S.gpg-agent
unix   2      [ ACC ]    STREAM     LISTENING  33672     /run/user/1000/gnupg/S.gpg-agent.extra
unix   2      [ ACC ]    STREAM     LISTENING  33106     /run/user/999/gnupg/S.gpg-agent.browser
unix   2      [ ACC ]    STREAM     LISTENING  34329     /run/user/1000/qtsingleapp-flysea-d37-3e8
unix   2      [ ACC ]    STREAM     LISTENING  31372     /var/run/xdmctl/dmctl/socket
unix   2      [ ACC ]    STREAM     LISTENING  34340     /run/user/1000/qtsingleapp-flyref-8d28-3e8
unix   2      [ ACC ]    STREAM     LISTENING  34344     /run/user/1000/qtsingleapp-flyupd-177c-3e8
unix   2      [ ACC ]    STREAM     LISTENING  34619     /run/user/1000/pulse/native
unix   2      [ ACC ]    STREAM     LISTENING  33112     @/tmp/.X11-unix/X0
unix   2      [ ACC ]    STREAM     LISTENING  34908     /run/user/1000/qtsingleapp-qtnoti-684-3e8
unix   2      [ ACC ]    STREAM     LISTENING  33986     @/tmp/.ICE-unix/976
unix   2      [ ACC ]    STREAM     LISTENING  34045     @/tmp/dbus-qhN23AIFGb
unix   3      [ ]        DGRAM      27530     /run/systemd/notify
unix   2      [ ]        DGRAM      27531     /run/systemd/cgroups-agent
unix   2      [ ACC ]    STREAM     LISTENING  27534     /run/systemd/private
unix   2      [ ACC ]    SEQPACKET LISTENING  27540     /run/udev/control
unix   2      [ ]        DGRAM      27544     /run/systemd/journal/syslog
unix   2      [ ACC ]    STREAM     LISTENING  27545     /run/systemd/fck.progress
unix   2      [ ACC ]    STREAM     LISTENING  27546     /run/systemd/journal/stdout
unix   7      [ ]        DGRAM      27547     /run/systemd/journal/socket
```

- -i - отображает статистику по интерфейсам (количество успешно принятых пакетов, количество ошибок, количество потерянных пакетов и количество потерянных пакетов из-за переполнения)


```

root@astra:~# netstat -i
Kernel Interface table

```

Iface	MTU	RX-OK	RX-ERR	RX-DRP	RX-OVR	TX-OK	TX-ERR	TX-DRP	TX-OVR	Flg
eth0	1500	3297	0	0	0	556	0	0	0	BMRU
eth1	1500	842	0	0	0	218	0	0	0	BMRU
eth2	1500	862	0	0	0	539	0	0	0	BMRU
lo	65536	68	0	0	0	68	0	0	0	LRU

- -n - отображение портов в числовом формате (без преобразования в имена сервисов);
- -p - отображение идентификатора процесса, управляющего соединением;
- -r - отображение таблицы маршрутизации;

```

root@astra:~# netstat -r
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
default	192.168.0.1	0.0.0.0	UG	0	0	0	eth0
net1	0.0.0.0	255.255.255.0	U	0	0	0	eth0
net2	0.0.0.0	255.255.255.128	U	0	0	0	eth1
net2	0.0.0.0	255.255.255.0	U	0	0	0	eth2

- -s - отображение статистики сетевых протоколов;

```

root@astra:~# netstat -s
Ip:
  Forwarding: 1
  3734 total packets received
  15 with invalid headers
  21 with invalid addresses
  691 forwarded
  0 incoming packets discarded
  2891 incoming packets delivered
  1176 requests sent out
Icmp:
  101 ICMP messages received
  0 input ICMP message failed
  ICMP input histogram:
    destination unreachable: 40
    echo requests: 37
    echo replies: 24
  158 ICMP messages sent
  0 ICMP messages failed
  ICMP output histogram:
    destination unreachable: 40
    time exceeded: 15
    echo requests: 66
    echo replies: 37
IcmpMsg:
  InType0: 24
  InType3: 40
  InType8: 37
  OutType0: 37
  OutType3: 40
  OutType8: 66
  OutType11: 15

```

- -t - отображение только TCP-соединений, применяется с командами выше;
- -u - отображение только UDP-соединений, применяется с командами выше.

Команда `nstat` также предоставляет статистику сетевых подключений, но использует протокол SNMP для сбора этой информации. Эта команда позволяет получить более детальную статистику по протоколам и типам ошибок:

```
root@astra:~# nstat
#kernel
IpInReceives          3737          0.0
IpInHdrErrors          15            0.0
IpInAddrErrors         21            0.0
IpForwDatagrams        691           0.0
IpInDelivers           2894          0.0
IpOutRequests          1176          0.0
IcmpInMsgs             101           0.0
IcmpInDestUnreaches    40            0.0
IcmpInEchos            37            0.0
IcmpInEchoReps         24            0.0
IcmpOutMsgs            158           0.0
IcmpOutDestUnreaches   40            0.0
IcmpOutTimeExcds       15            0.0
IcmpOutEchos           66            0.0
IcmpOutEchoReps        37            0.0
IcmpMsgInType0          24            0.0
IcmpMsgInType3          40            0.0
IcmpMsgInType8          37            0.0
IcmpMsgOutType0         37            0.0
IcmpMsgOutType3         40            0.0
IcmpMsgOutType8         66            0.0
IcmpMsgOutType11        15            0.0
TcpActiveOpens          11            0.0
TcpAttemptFails         6             0.0
TcpInSegs               61            0.0
TcpOutSegs              81            0.0
TcpRetransSegs          7             0.0
TcpOutRsts               6             0.0
UdpInDatagrams          2029          0.0
```

Команда `ss` (Socket Statistics) является альтернативой команде `netstat` и используется для отображения информации о сетевых сокетах (открытых соединениях) на Linux системах.

```
root@astra:~# ss
Netid State      Recv-Q Send-Q           Local Address:Port              Peer Address:Port
u_dgr ESTAB      0      0                /run/systemd/notify 27530             * 0
u_dgr ESTAB      0      0                /run/systemd/journal/socket 27547             * 0
u_dgr ESTAB      0      0                /run/systemd/journal/dev-log 27609             * 0
u_dgr ESTAB      0      0                               * 34537             * 27609
u_str ESTAB      0      0                /var/run/dbus/system_bus_socket 34522             * 34521
u_str ESTAB      0      0                @/tmp/.X11-unix/X0 34236             * 34235
u_str ESTAB      0      0                @/tmp/.X11-unix/X0 34384             * 34383
u_str ESTAB      0      0                               * 33895             * 33896
u_str ESTAB      0      0                               * 29919             * 29920
u_dgr ESTAB      0      0                               * 28452             * 28453
u_str ESTAB      0      0                @/tmp/.X11-unix/X0 34115             * 34114
u_str ESTAB      0      0                               * 33962             * 33967
u_str ESTAB      0      0                               * 32901             * 32900
u_str ESTAB      0      0                @/tmp/dbus-Spc5zd6ogZ 34739             * 34738
```

Конечно, в любых ситуациях команда в простейшем виде не так уж и полезна. Вывести список сокетов, ожидающих соединений.

```

root@astra:~# ss -l
Netid State      Recv-Q Send-Q           Local Address:Port           Peer Address:Port
n1      UNCONN      0      0                rtnl:avahi-daemon/444         *
n1      UNCONN      0      0                rtnl:kernel                   *
n1      UNCONN      0      0                rtnl:-1505754678             *
n1      UNCONN      0      0                rtnl:-1505754678             *
n1      UNCONN      0      0                rtnl:avahi-daemon/444         *
n1      UNCONN      4352   0                tcpdiag:ss/3041              *
n1      UNCONN      768    0                tcpdiag:kernel               *
n1      UNCONN      0      0                audit:-1675370174            *
n1      UNCONN      0      0                audit:systemd/1              *
n1      UNCONN      0      0                audit:kernel                 *
n1      UNCONN      0      0                audit:sudo/1312              *
n1      UNCONN      0      0                audit:systemd/1              *
n1      UNCONN      0      0                fiblookup:kernel             *
n1      UNCONN      0      0                connector:kernel              *
n1      UNCONN      0      0                uevent:Xorg/903              *

```

Весьма удобная возможность команды ss заключается в том, что она может **фильтровать вывод**, используя состояния TCP (или состояния жизненного цикла соединения). Благодаря использованию состояний облегчается фильтрация вывода ss. А именно, здесь доступны все стандартные состояния TCP:

1. established
2. syn-sent
3. syn-recv
4. fin-wait-1
5. fin-wait-2
6. time-wait
7. closed
8. close-wait
9. last-ack
10. listening
11. closing

Кроме того, ss распознаёт следующие идентификаторы состояний:

1. all (все вышеперечисленные состояния)
2. connected (все состояния, кроме ожидающих соединения и закрытых)
3. synchronized (все состояния, соответствующие установленным соединениям, за исключением syn-sent)
4. bucket (состояния, представляющие собой минисокеты, например — time-wait и syn-recv)
5. big (всё кроме того, что соответствует идентификатору bucket)

iproute2 — это набор утилит Linux, используемых для управления и настройки сетевых интерфейсов, IP-адресов, таблиц маршрутизации и других параметров, связанных с сетью. Он представляет собой более мощную и гибкую альтернативу старым утилитам ifconfig, route и arp.

Дерево объектов и команд утилиты ip ^[3] [скрыть]			
Утилита	Объект	Команды	Описание
ip	address	add del show flush	IP- или IPv6-адрес устройства
	addrlabel	add del list flush	Конфигурирование меток
	link	set show	Конфигурирование сетевых устройств
	maddr	show add del	Конфигурирование адресов групповой рассылки
	monitor	all LISTofOBJECTS	Мониторинг состояния устройств
	mroute	show	Кэш групповой маршрутизации
	neigh	add del change replace show flush	Кэш ARP или NDISC
	route	get list / flush add del change append replace monitor	Управление маршрутизацией
	rule	list add del flush	Правила маршрутизации
	tunnel	add change del show	Туннелирование через протокол IP

Некоторые из основных утилит, предоставляемых iproute2, включают:

- ip: используется для отображения и работы с сетевыми интерфейсами, IP-адресами и таблицами маршрутизации.

```

root@astra:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:3f:5b:ef brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.108/24 brd 192.168.0.255 scope global noprefixroute dynamic eth0
        valid_lft 68094sec preferred_lft 68094sec
    inet6 fe80::e606:f3c:9ae7:4b33/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:24:58:b4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.2/25 brd 192.168.1.127 scope global noprefixroute eth1
        valid_lft forever preferred_lft forever
    inet 192.168.246.129/24 brd 192.168.246.255 scope global noprefixroute dynamic eth1
        valid_lft 1336sec preferred_lft 1336sec
    inet6 fe80::868e:3418:ff8b:9475/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:3b:11:e7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.131/24 brd 192.168.1.255 scope global noprefixroute eth2
        valid_lft forever preferred_lft forever
    inet 192.168.246.128/24 brd 192.168.246.255 scope global noprefixroute dynamic eth2
        valid_lft 967sec preferred_lft 967sec
    inet6 fe80::e76:7490:c618:65f5/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

```

Для настройки роутинга служит команда `ip route`. Выполненная без параметров, она покажет список текущих правил маршрутизации

```
default via 192.168.0.1 dev eth0 proto dhcp metric 100
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.108 metric 100
192.168.1.0/25 dev eth1 proto kernel scope link src 192.168.1.2 metric 101
192.168.1.0/24 dev eth2 proto kernel scope link src 192.168.1.131 metric 102
```

Изначально предопределены таблицы *local*, *main* и *default*. В таблицу *local* ядро заносит записи для локальных IP адресов (чтобы трафик на эти IP-адреса оставался локальным и не пытался уходить во внешнюю сеть), а также для бродкастов. Таблица *main* является основной, и именно она используется, если в команде не указано какую таблицу использовать (т.е. выше мы видели именно таблицу *main*). Таблица *default* изначально пуста.

```
root@astra:~# ip route show table local
local 127.0.0.0/8 dev lo proto kernel scope host src 127.0.0.1
local 127.0.0.1 dev lo proto kernel scope host src 127.0.0.1
broadcast 127.255.255.255 dev lo proto kernel scope link src 127.0.0.1
local 192.168.0.108 dev eth0 proto kernel scope host src 192.168.0.108
broadcast 192.168.0.255 dev eth0 proto kernel scope link src 192.168.0.108
local 192.168.1.2 dev eth1 proto kernel scope host src 192.168.1.2
broadcast 192.168.1.127 dev eth1 proto kernel scope link src 192.168.1.2
local 192.168.1.131 dev eth2 proto kernel scope host src 192.168.1.131
broadcast 192.168.1.255 dev eth2 proto kernel scope link src 192.168.1.131
```

Правила, по которым ядро выбирает в какую таблицу отправлять пакеты:

```
root@astra:~# ip rule
0:      from all lookup local
32766:  from all lookup main
32767:  from all lookup default
```

Число в начале строки – идентификатор правила, *from all* – условие, означает пакеты с любых адресов, *lookup* указывает в какую таблицу направлять пакет. Если пакет попадает под несколько правил, то он проходит их все по порядку возрастания идентификатора. Конечно, если пакет подпадет под какую-либо запись маршрутизации, то последующие записи маршрутизации и последующие правила он уже проходить не будет.

Возможные условия:

1. *from* – мы уже рассматривали выше, это проверка отправителя пакета.
2. *to* – получатель пакета.
3. *iif* – имя интерфейса, на который пришел пакет.

4. `oif` – имя интерфейса, с которого уходит пакет. Это условие действует только для пакетов, исходящих из локальных сокетов, привязанных к конкретному интерфейсу.
 5. `tos` – значение поля TOS IP-пакета.
 6. `fwmark` – проверка значения FWMARK пакета. Это условие дает потрясающую гибкость правил. При помощи правил `iptables` можно отфильтровать пакеты по огромному количеству признаков и установить определенные значения FWMARK. А затем эти значения учитывать при роутинге.
- `bridge` (мост): используется для настройки мостов Ethernet.
 - `tc`: используется для настройки параметров управления трафиком и качества обслуживания.

Выбор команды для вывода статистики зависит от ситуации, степени знакомства с каждой из них и возможности установки дополнительных пакетов. Команда `netstat` является стандартной и может помочь при разных вопросах. Когда необходимо подробнее поработать со статистикой протоколов – `nstat`. Если есть необходимость получить подробную информацию о портах – `ss`.

Пример: проверить используемые порты.

Создадим еще одну терминальную сессию, в которой запустим прослушивание порта 8080 по TCP.

```
poly@astra:~$ nc -l 8080
```

Socket Statistic:

```
root@astra:~# ss -tulnp
```

Netid	State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port
udp	UNCONN	0	0	*:45026	*:*
users: (("avahi-daemon", pid=444, fd=14))					
udp	UNCONN	0	0	*:68	*:*
users: (("dhclient", pid=550, fd=6))					
udp	UNCONN	0	0	*:68	*:*
users: (("dhclient", pid=549, fd=6))					
udp	UNCONN	0	0	*:68	*:*
users: (("dhclient", pid=559, fd=6))					
udp	UNCONN	0	0	127.0.0.1:177	*:*
users: (("fly-dm", pid=503, fd=8))					
udp	UNCONN	0	0	*:5353	*:*
users: (("avahi-daemon", pid=444, fd=12))					
udp	UNCONN	0	0	:::5353	:::*
users: (("avahi-daemon", pid=444, fd=13))					
udp	UNCONN	0	0	:::56576	:::*
users: (("avahi-daemon", pid=444, fd=15))					
tcp	LISTEN	0	1	*:8080	*:*
users: (("nc", pid=2860, fd=3))					
tcp	LISTEN	0	5	127.0.0.1:631	*:*
users: (("cupsd", pid=1189, fd=7))					
tcp	LISTEN	0	5	:::1:631	:::*
users: (("cupsd", pid=1189, fd=6))					

Netstat:

```

root@astra:~# netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:631           0.0.0.0:*                LISTEN      1189/cupsd
tcp        0      0 0.0.0.0:8080            0.0.0.0:*                LISTEN      2860/nc
tcp6       0      0 :::1:631                :::*                    LISTEN      1189/cupsd
udp        0      0 0.0.0.0:45026           0.0.0.0:*                -           444/avahi-daemon: r
udp        0      0 0.0.0.0:68              0.0.0.0:*                -           550/dhclient
udp        0      0 0.0.0.0:68              0.0.0.0:*                -           549/dhclient
udp        0      0 0.0.0.0:68              0.0.0.0:*                -           559/dhclient
udp        0      0 127.0.0.1:177           0.0.0.0:*                -           503/fly-dm
udp        0      0 0.0.0.0:5353            0.0.0.0:*                -           444/avahi-daemon: r
udp6       0      0 :::5353                 :::*                    -           444/avahi-daemon: r
udp6       0      0 :::56576                :::*                    -           444/avahi-daemon: r

```

Вывод

В ходе выполнения лабораторной работы, были настроены сетевые интерфейсы на трех узлах на ОС Astra. Особенность в том, что два узла были клиентами, а один выступал в роли маршрутизатора. Были настроены таблицы маршрутизации так, что клиенты, находящиеся в разных сетях, могли устанавливать соединение друг с другом, а так же иметь доступ в интернет.

Рассмотрена настройка сетевых интерфейсов и таблиц маршрутизации с помощью команд: `ifconfig`, `route`, `iptables`.

Так же настроены следующие конфигурационные файлы: `/etc/hostname`, `/etc/hosts`, `/etc/resolv.conf`, `networks`, `ip_forward` и другие.