

Университет ИТМО

**Сети ЭВМ и телекоммуникации**  
**Лабораторная работа №4**

Выполнила: Калугина Марина  
Группа: Р3302

г. Санкт-Петербург

2020 г.

# Цель работы

**Цель работы** – изучение основных методов настройки маршрутизируемых компьютерных сетей на примере сети, состоящей из компьютеров под управлением ОС Linux.

## Ход работы

### Общее задание

#### 1. Вариант 4.

Изображение топологии для варианта 4 изображено на рисунке 1.

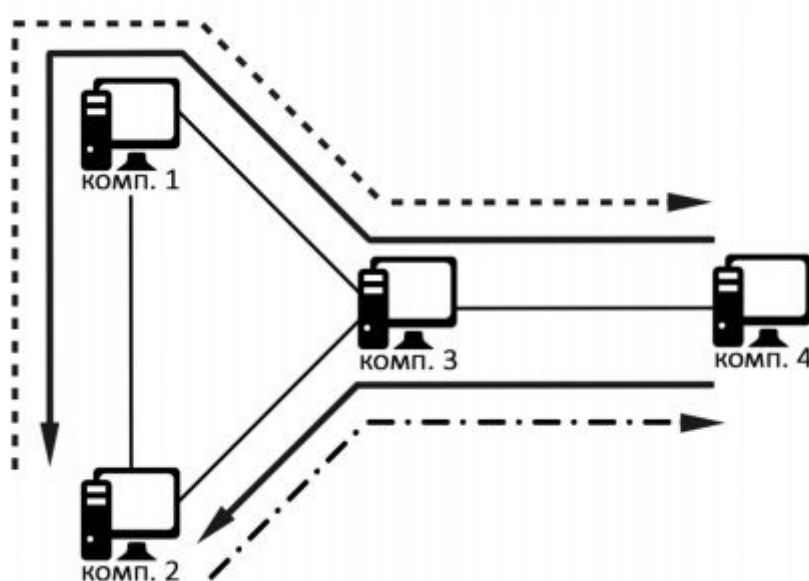


Рисунок 1.

IPv4: 9.6.Y.X

2. На всех адаптерах всех компьютеров в топологии, представленной в варианте, настроить IPv4-адреса и IPv6.

Компьютер 1

На рисунке 2 изображены адреса интерфейсов компьютера 1.

IPv4: 6.8.13.1/29, 6.8.12.1/29

IPv6: ::ffff:6.8.13.1/125, ::ffff:6.8.12.1/125

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    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 08:00:27:e4:5b:d0 brd ff:ff:ff:ff:ff:ff
    inet 6.8.13.1/29 brd 6.8.13.7 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.13.1/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fee4:5bd0/64 scope link
        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 08:00:27:36:98:54 brd ff:ff:ff:ff:ff:ff
    inet 6.8.12.1/29 brd 6.8.12.7 scope global eth1
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.12.1/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe36:9854/64 scope link
        valid_lft forever preferred_lft forever
```

Рисунок 2.

На рисунке 3 изображены адреса интерфейсов компьютера 2.

IPv4: 6.8.23.2/29, 6.8.12.2/29

IPv6: ::ffff:6.8.23.2/125, ::ffff:6.8.12.2/125

```

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    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 08:00:27:44:30:76 brd ff:ff:ff:ff:ff:ff
    inet 6.8.23.2/29 brd 6.8.23.7 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.23.2/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe44:3076/64 scope link
        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 08:00:27:a7:37:b6 brd ff:ff:ff:ff:ff:ff
    inet 6.8.12.2/29 brd 6.8.12.7 scope global eth1
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.12.2/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fea7:37b6/64 scope link
        valid_lft forever preferred_lft forever

```

Рисунок 3.

На рисунке 4 изображены адреса интерфейсов компьютера 3.

IPv4: 6.8.13.3/29, 6.8.23.3/29, 6.8.34.3/29

IPv6: ::ffff:6.8.13.3/125, 6.8.23.3/125, 6.8.34.3/125

```

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    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 08:00:27:7e:17:b2 brd ff:ff:ff:ff:ff:ff
    inet 6.8.13.3/29 brd 6.8.13.7 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.13.3/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7e:17b2/64 scope link
        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 08:00:27:95:93:dc brd ff:ff:ff:ff:ff:ff
    inet 6.8.23.3/29 brd 6.8.23.7 scope global eth1
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.23.3/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe95:93dc/64 scope link
        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 08:00:27:01:6c:4e brd ff:ff:ff:ff:ff:ff
    inet 6.8.34.3/29 brd 6.8.34.7 scope global eth2
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.34.3/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe01:6c4e/64 scope link
        valid_lft forever preferred_lft forever

```

Рисунок 4.

На рисунке 5 изображены адреса интерфейсов компьютера 4.

IPv4: 6.8.34.4/29

IPv6: ::ffff:6.8.34.4/29

```

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    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 08:00:27:1f:3e:b8 brd ff:ff:ff:ff:ff:ff
    inet 6.8.34.4/29 brd 6.8.34.7 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 ::ffff:6.8.34.4/125 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe1f:3eb8/64 scope link
        valid_lft forever preferred_lft forever

```

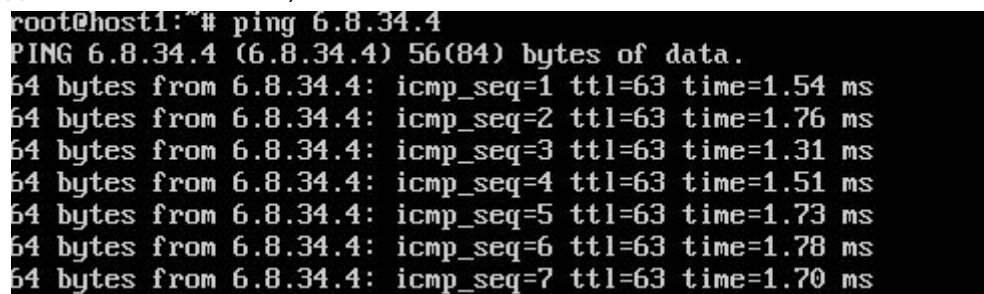
Рисунок 5.

3. На всех компьютерах настроить таблицы маршрутизации таким образом, чтобы обеспечивалась полная сетевая доступность.

Для проверки: на 1 компьютере запустить команды:

```
ping 6.8.12.2  
ping 6.8.23.2  
ping 6.8.13.3  
ping 6.8.34.3  
ping 6.8.23.3  
ping 6.8.34.4
```


На рисунке 6 изображен вывод команды для ping 6.8.34.4 (находящийся максимально далеко от машины 1)



```
root@host1:~# ping 6.8.34.4  
PING 6.8.34.4 (6.8.34.4) 56(84) bytes of data.  
64 bytes from 6.8.34.4: icmp_seq=1 ttl=63 time=1.54 ms  
64 bytes from 6.8.34.4: icmp_seq=2 ttl=63 time=1.76 ms  
64 bytes from 6.8.34.4: icmp_seq=3 ttl=63 time=1.31 ms  
64 bytes from 6.8.34.4: icmp_seq=4 ttl=63 time=1.51 ms  
64 bytes from 6.8.34.4: icmp_seq=5 ttl=63 time=1.73 ms  
64 bytes from 6.8.34.4: icmp_seq=6 ttl=63 time=1.78 ms  
64 bytes from 6.8.34.4: icmp_seq=7 ttl=63 time=1.70 ms
```

Рисунок 6

В ходе выполнения работы для проверки был написан скрипт (рисунок 10):



```
#!/bin/bash  
date  
cat nodes | while read output  
do  
    ping -c 1 "$output" > /dev/null  
    if [ $? -eq 0 ]; then  
        echo "node $output is up"  
    else  
        echo "node $output is down"  
    fi  
done  
  
cat nodes6 | while read output  
do  
    ping6 -c 1 "$output" > /dev/null  
    if [ $? -eq 0 ]; then  
        echo "node $output is up"  
    else  
        echo "node $output is down"  
    fi  
done
```

Рисунок 10

Компьютер 1 (Рис 11):

```
root@host1:~# ./ping_all.sh
Fri Apr 17 17:58:10 MSK 2020
node 6.8.12.1 is up
node 6.8.12.2 is up
node 6.8.23.2 is up
node 6.8.23.3 is up
node 6.8.13.1 is up
node 6.8.13.3 is up
node 6.8.34.3 is up
node 6.8.34.4 is up
node ::ffff:6:8:12:1 is up
node ::ffff:6:8:12:2 is up
node ::ffff:6:8:23:2 is up
node ::ffff:6:8:23:3 is up
node ::ffff:6:8:13:1 is up
node ::ffff:6:8:13:3 is up
node ::ffff:6:8:34:3 is up
node ::ffff:6:8:34:4 is up
```

Рисунок 11

Компьютер 2 (Рис 12):

```
root@host2:~# ./ping_all.sh
Fri Apr 17 17:57:43 MSK 2020
node 6.8.12.1 is up
node 6.8.12.2 is up
node 6.8.23.2 is up
node 6.8.23.3 is up
node 6.8.13.1 is up
node 6.8.13.3 is up
node 6.8.34.3 is up
node 6.8.34.4 is up
node ::ffff:6:8:12:1 is up
node ::ffff:6:8:12:2 is up
node ::ffff:6:8:23:2 is up
node ::ffff:6:8:23:3 is up
node ::ffff:6:8:13:1 is up
node ::ffff:6:8:13:3 is up
node ::ffff:6:8:34:3 is up
node ::ffff:6:8:34:4 is up
```

Рисунок 12

Компьютер 3 (Рисунок 13):

```
root@host3:~# ./ping_all.sh
Fri Apr 17 17:57:09 MSK 2020
node 6.8.12.1 is up
node 6.8.12.2 is up
node 6.8.23.2 is up
node 6.8.23.3 is up
node 6.8.13.1 is up
node 6.8.13.1 is up
node 6.8.34.3 is up
node 6.8.34.4 is up
node ::ffff:6:8:12:1 is up
node ::ffff:6:8:12:2 is up
node ::ffff:6:8:23:2 is up
node ::ffff:6:8:23:3 is up
node ::ffff:6:8:13:1 is up
node ::ffff:6:8:13:3 is up
node ::ffff:6:8:34:3 is up
node ::ffff:6:8:34:4 is up
```

Рисунок 14

Компьютер 4 (Рисунок 15):

```
root@host4:~# ./ping_all.sh
Fri Apr 17 18:00:34 MSK 2020
node 6.8.12.1 is up
node 6.8.12.2 is up
node 6.8.23.2 is up
node 6.8.23.3 is up
node 6.8.13.1 is up
node 6.8.13.3 is up
node 6.8.34.3 is up
node 6.8.34.4 is up
node ::ffff:6:8:12:1 is up
node ::ffff:6:8:12:2 is up
node ::ffff:6:8:23:2 is up
node ::ffff:6:8:23:3 is up
node ::ffff:6:8:13:1 is up
node ::ffff:6:8:13:3 is up
node ::ffff:6:8:34:3 is up
node ::ffff:6:8:34:4 is up
```

Рисунок 15

4. Изучить Linux-утилиту nc. Запустить её в режиме клиента на машине А и в режиме сервера – на машине Б, используя для передачи произвольный порт. Передать в виде текстового сообщение свое имя от Б к А.

Сервер запущен на машине 2 (команда nc -l 2339), клиент запускается на машине 4 (команда nc 6.8.12.2 2399).

На рисунке 16 и 17 изображен сервер и клиент соответственно.



```
root@host2:~# nc -l 2399
Hello, Marina
```

Рисунок 16

```
root@host4:~# nc 6.8.12.2 2399
Hello, Marina
```

Рисунок 17

5. Изучить назначение Linux-утилиты iptables и создать на компьютерах А и/или Б простейший Firewall с помощью этой утилиты и убедиться с помощью команды nc и ping, что настроенные правила фильтрации iptables работоспособны, для чего нужно сначала попытаться передать запрещенный пакет, а затем разрешенный.

- Запретить передачу только тех пакетов, которые отправлены на TCP-порт, заданный в настройках утилиты nc.

На машине 2(клиент) пишем команду:

```
iptables -A INPUT -p UDP --dport 2399 -j DROP
```

Передача запрещенного пакета:

Клиент “nc 6.8.12.2 2399” (Рис 18):

```
root@host4:~# nc 6.8.12.2 2399
1
2
```

Рисунок 18

Сервер “nc -l 2399”(Рис 19):

```
root@host2:~# nc -l 2399
^C
```

Рисунок 19

Передача разрешенного пакета (отправим TCP Сообщение на порт 2400):

Клиент “nc 6.8.12.2 2400” (Рис 20):

```
root@host4:~# nc 6.8.12.2 2400
1
2
```

Рисунок 20

Сервер “nc -l 2400” (Рис 21):

```
root@host2:~# nc -l 2400
1
2
```

Рисунок 21.

- Запретить приём только тех пакетов, которые отправлены с UDPпорта утилиты nc.

На машине 4(клиент) пишем команду:

```
iptables -A OUTPUT -p TCP --dport 2399 -j DROP
```

Передача запрещенного пакета

Клиент “nc -u -p 2399 6.8.12.2 2399” (Рис 22):

```
root@host4:~# nc -u -p 2399 6.8.12.2 2399
1
2
```

Рисунок 22

Сервер “nc -u -l 2399” (Рис 23):

```
root@host2:~# nc -u -l 2399
```

Рисунок 23

Передача разрешенного пакета:

Клиент “nc -u 6.8.12.2 2399” (Рис 24):

```
root@host4:~# nc -u -p 2400 6.8.12.2 2399
udp
```

Рисунок 24.

Сервер “nc -u -l 2399” (Рис 25):

```
root@host2:~# nc -u -l 2399
udp
```

Рисунок 25.

- Запретить передачу только тех пакетов, которые отправлены с IPадреса компьютера А.

На 2-й машине (сервер) написать команду:

```
iptables -A OUTPUT -s 6.8.12.2 -j DROP
```

Отправка запрещенного пакета:

Сервер "ping -I 6.8.12.2 6.8.34.4" (Рис 25):

```
root@host2:~# ping -I 6.8.12.2 6.8.34.4
PING 6.8.34.4 (6.8.34.4) from 6.8.12.2 : 56(84) bytes of data.
ping: sendmsg: Operation not permitted
```

Рисунок 25

- Запретить приём только тех пакетов, которые отправлены на IPадрес компьютера Б.

На 4-й машине (клиент) написать команду:

```
iptables -A INPUT -s 6.8.34.4 -j DROP
```

Отправка запрещенного пакета:

Сервер "ping -c1 -I 6.8.23.2 6.8.34.4" (Рис 26):

```
root@host2:~# ping -c1 -I 6.8.23.2 6.8.34.4
PING 6.8.34.4 (6.8.34.4) from 6.8.23.2 : 56(84) bytes of data.

--- 6.8.34.4 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

Рисунок 26

- Запретить приём и передачу ICMP-пакетов, размер которых превышает 1000 байт, а поле TTL при этом меньше 10.

Для запрета приема на машине 2 (сервер) написать команду:

```
iptables -A INPUT -p ICMP -m ttl --ttl-lt 10 -m length --length-gt 1000 -j DROP
```

Отправка запрещенного запроса "ping -c1 -s 10000 -t 10" (Рис 27)

```
root@host4:~# ping -c1 -s 10000 -t 10 6.8.23.2
PING 6.8.23.2 (6.8.23.2) 10000(10028) bytes of data.

--- 6.8.23.2 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

Рисунок 27

Отправка разрешенного запроса "ping -s 10 -t 100" (Рис 28):

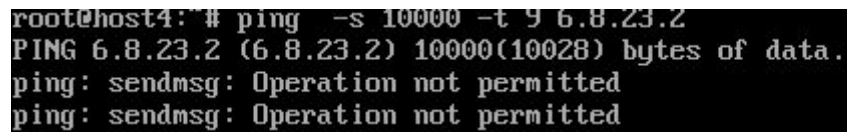
```
root@host4:~# ping -s 10 -t 100 6.8.23.2
PING 6.8.23.2 (6.8.23.2) 10(38) bytes of data.
18 bytes from 6.8.23.2: icmp_seq=1 ttl=63
18 bytes from 6.8.23.2: icmp_seq=2 ttl=63
```

Рисунок 29

Для запрета передачи на машине 4(клиент) написать команду:

```
iptables -A OUTPUT -p ICMP -m ttl ttl-lt 10 -m length 1000:65535 -j DROP
```

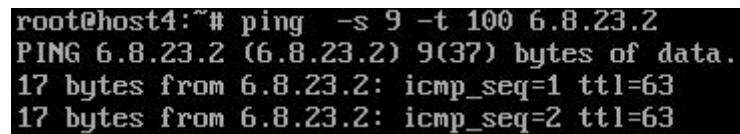
Отправка запрещенного запроса “ping -c1 -s 10000 -t 10” (Рис 30):



```
root@host4:~# ping -s 10000 -t 9 6.8.23.2
PING 6.8.23.2 (6.8.23.2) 10000(10028) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
```

Рисунок 30

Отправка разрешенного запроса “ping -s 9 -t 100” (Рисунок 31)



```
root@host4:~# ping -s 9 -t 100 6.8.23.2
PING 6.8.23.2 (6.8.23.2) 9(37) bytes of data.
17 bytes from 6.8.23.2: icmp_seq=1 ttl=63
17 bytes from 6.8.23.2: icmp_seq=2 ttl=63
```

Рисунок 31

## V1 для ipv4

На рисунке 32 представлено задание к варианту 4 лабораторной работы

### 4.4.4 Вариант 4

На рисунке 4.8 изображена топология сети и требуемый путь прохождения сетевых пакетов. В компьютере 2 имеется два сетевых адаптера, на которых должны быть настроены различные IP адреса: IP1 и IP2. IP1 — адрес интерфейса, подключенного к компьютеру 1, IP2 — адрес интерфейса, подключенного к компьютеру 3.

Необходимо настроить сеть таким образом, чтобы при отправлении ICMP Echo Request с компьютера 4 на IP1 пакет проходил через компьютер 3 и компьютер 1, а ICMP Echo Reply шел с компьютера 2 сразу через компьютер 3 (штрих-пунктирная линия на рисунке 4.8). При отправлении ICMP Echo Request с компьютера 4 на IP2 пакет должен пройти через компьютер 3 сразу на компьютер 2, а ICMP Echo Reply должен пройти с компьютера 2 на компьютер 1 и далее на компьютер 3 (штриховая линия на рисунке 4.8). Требования к составу отчёта см. в подразделе 4.5.

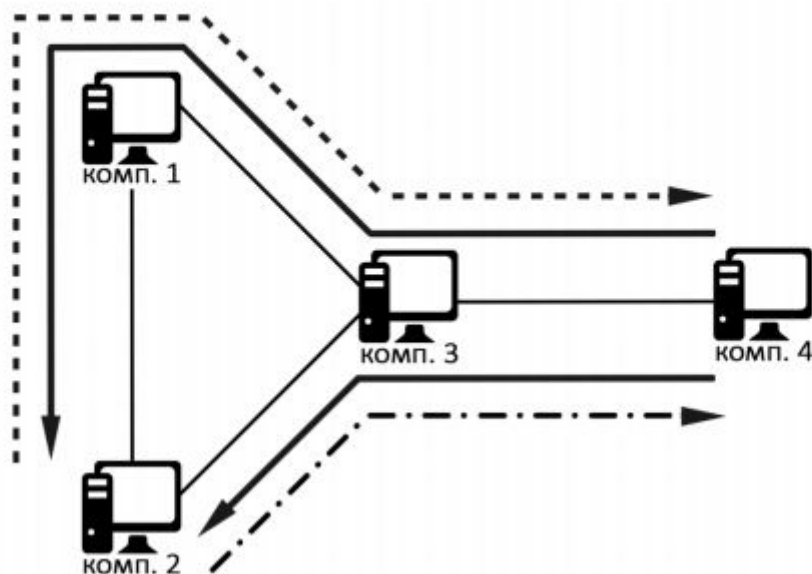


Рисунок 32.

Для настройки были исполнены следующие команды:

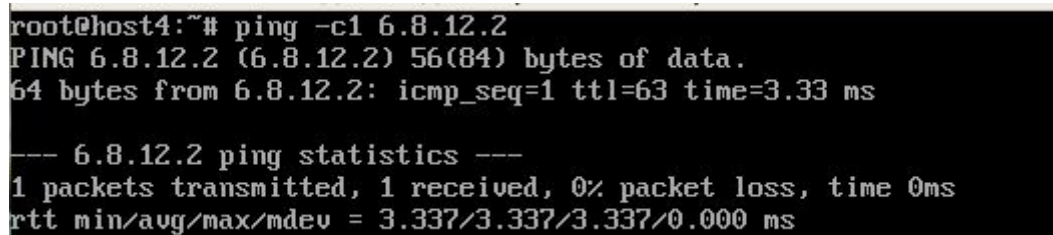
На машине 2 были созданы 2 таблицы для правил с роутингом, в зависимости от того, на какой адрес был отправлен пакет, пакет помечается 0x11 или 0x33 и в дальнейшем, в зависимости от метки происходит выбор таблицы для пересылки пакета:

```
# разные правила добавляем в разные таблицы
ip route add 6.8.34.0/29 via 6.8.12.1 table 100
ip route add 6.8.23.0/29 via 6.8.23.2 table 200
```

```
# Добавляем правила для масок
ip rule add fmark 0x11 table 100
ip rule add fmark 0x33 table 200
```

```
# Маркируем ICMP-Reply
iptables -t mangle -A OUTPUT -s 6.8.12.2 -d 6.8.34.4 -p icmp --icmp-type 0 -j MARK
--set-mark 0x33
iptables -t mangle -A OUTPUT -s 6.8.23.2 -d 6.8.34.4 -p icmp --icmp-type 0 -j MARK
--set-mark 0x11
```

Для проверки был отправлен один пакет с 4 копыютера на адрес 6.8.12.2 "ping -c1 6.8.12.2" (Рис 33).

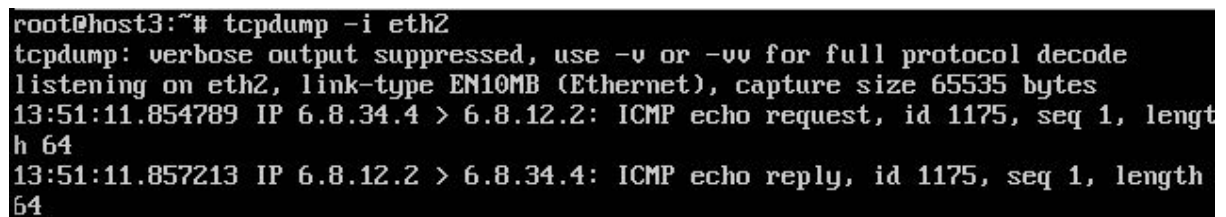


```
root@host4:~# ping -c1 6.8.12.2
PING 6.8.12.2 (6.8.12.2) 56(84) bytes of data.
64 bytes from 6.8.12.2: icmp_seq=1 ttl=63 time=3.33 ms

--- 6.8.12.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.337/3.337/3.337/0.000 ms
```

Рисунок 33

На компьютере 3 был запущен "tcpdump -i eth2" (Рисунок 34). Сюда пришел сначала ICMP-запрос, потом ICMP-ответ:



```
root@host3:~# tcpdump -i eth2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 65535 bytes
13:51:11.854789 IP 6.8.34.4 > 6.8.12.2: ICMP echo request, id 1175, seq 1, length 64
13:51:11.857213 IP 6.8.12.2 > 6.8.34.4: ICMP echo reply, id 1175, seq 1, length 64
```

Рисунок 34

На компьютере 2 был запущен “tcpdump -i eth1” (Рис 35). Через 2 машину прошел ICMP-запрос:

```
root@host1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
13:51:12.459191 IP 6.8.34.4 > 6.8.12.2: ICMP echo request, id 1175, seq 1, length 64
```

Рисунок 35

На рисунке 36 (и 33, 34, 35) изображено отправление одного пакета на адрес 6.8.12.2 (2-й машины): при отправлении ICMP Echo Request с компьютера 4 на IP1 = 6.8.12.2 пакет проходит через компьютер 3 и компьютер 1, а ICMP Echo Reply шел с компьютера 2 сразу через компьютер 3

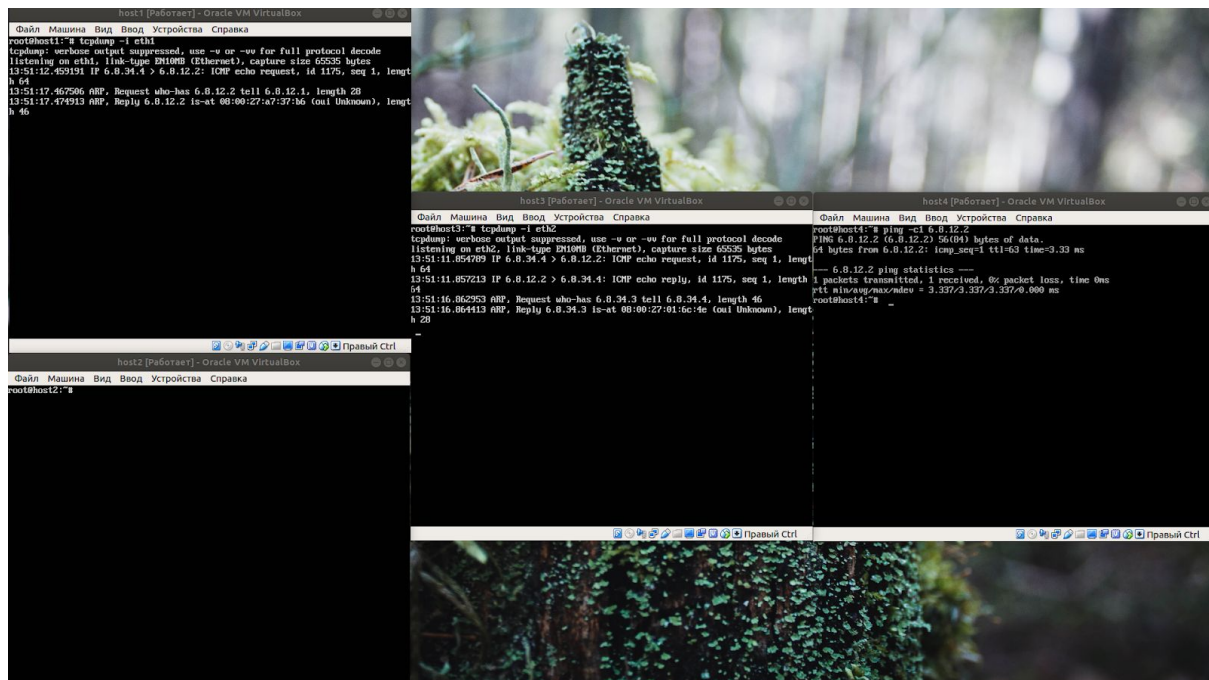


Рисунок 36

Для второй проверки с 4 машины был отправлен один пакет на адрес 6.8.23.2 “ping -c1 6.8.23.2” (Рисунок 37):

```

root@host4:~# ping -c1 6.8.12.2
PING 6.8.12.2 (6.8.12.2) 56(84) bytes of data.
64 bytes from 6.8.12.2: icmp_seq=1 ttl=63 time=3.33 ms

--- 6.8.12.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.337/3.337/3.337/0.000 ms
root@host4:~# ping -c1 6.8.23.2
PING 6.8.23.2 (6.8.23.2) 56(84) bytes of data.
64 bytes from 6.8.23.2: icmp_seq=1 ttl=62 time=2.80 ms

--- 6.8.23.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 2.800/2.800/2.800/0.000 ms

```

Рисунок 37

На компьютере 3 был запущен “tcpdump -i eth2” (Рисунок 38). Сюда пришел сначала ICMP-запрос, потом ICMP-ответ:

```

root@host3:~# tcpdump -i eth2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 65535 bytes
14:36:05.873709 IP 6.8.34.4 > 6.8.23.2: ICMP echo request, id 1187, seq 1, length 64
14:36:05.875544 IP 6.8.23.2 > 6.8.34.4: ICMP echo reply, id 1187, seq 1, length 64

```

Рисунок 38

На компьютере 2 был запущен “tcpdump -i eth1” (Рис 39). Сюда пришел ICMP-ответ :

```

root@host1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
14:36:06.478882 IP 6.8.23.2 > 6.8.34.4: ICMP echo reply, id 1187, seq 1, length 64

```

Рисунок 39

На рисунке 40 (а также рисунках 37-39) можно увидеть корректность работы. При отправлении ICMP Echo Request с компьютера 4 на IP2 = 6.8.23.2 пакет прошел через компьютер 3 сразу на компьютер 2, а ICMP Echo Reply прошел с компьютера 2 на компьютер 1 и далее на компьютер 3.



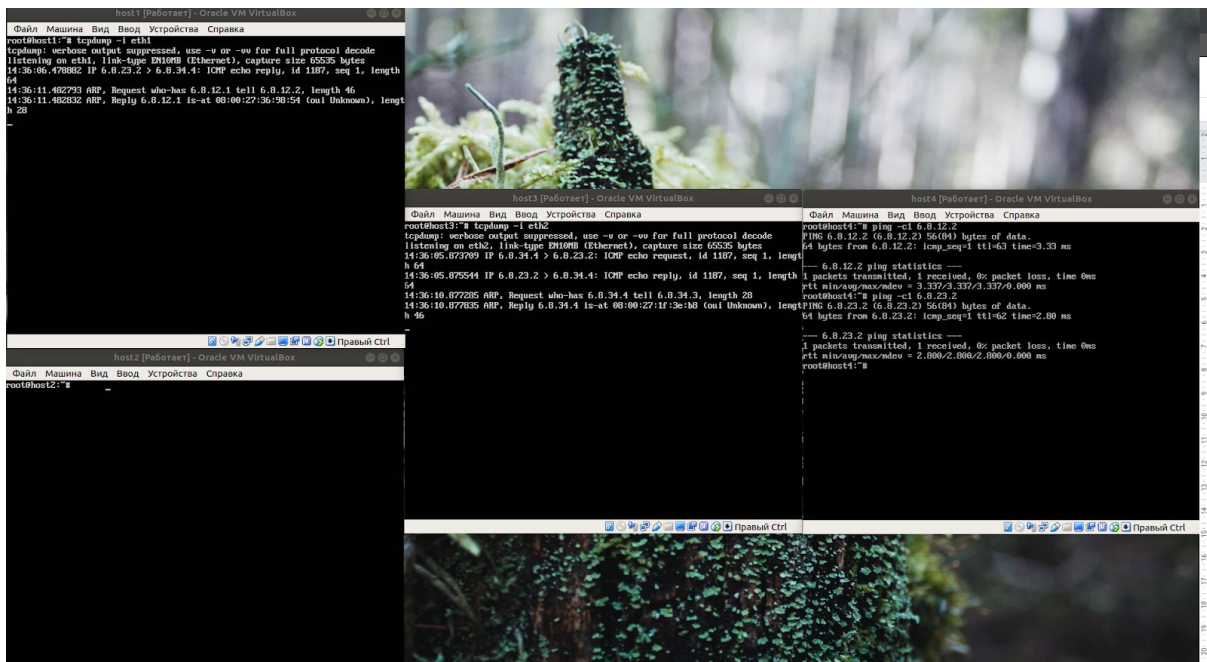


Рисунок 40

На рисунке 40(а) показан итоговый вывод iptables для машины 2. Как видно, у нас действительно есть 2 таблицы с разными правилами для пакетов разного типа.

```
root@host2:~# iptables -t mangle -L OUTPUT
Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
MARK       icmp -- 6.8.12.2                6.8.34.4             icmp echo-reply MA
RK set 0x33
MARK       icmp -- 6.8.23.2                6.8.34.4             icmp echo-reply MA
RK set 0x11
root@host2:~#
```

Рисунок 40(а).

На рисунке 40(б) показан вывод ip rule. Здесь указано, какими таблицами стоит пользоваться пакетам с определенной маркировкой для компьютера 2

```
root@host2:~# ip rule
0:      from all lookup local
32764:  from all fwmark 0x33 lookup 200
32765:  from all fwmark 0x11 lookup 100
32766:  from all lookup main
32767:  from all lookup default
```

Рисунок 40(б)

## V1 для ipv6

Для настройки были исполнены следующие команды:

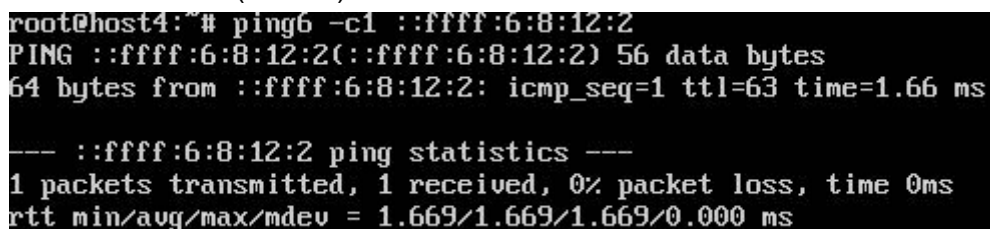
На машине 2 были созданы 2 таблицы для правил с роутингом, в зависимости от того, на какой адрес был отправлен пакет, пакет помечается 0x11 или 0x33 и в дальнейшем, в зависимости от метки происходит выбор таблицы для пересылки пакета:

```
# разные правила добавляем в разные таблицы
ip -6 route add ::ffff:6:8:34:0/29 via ::ffff:6:8:12:1 table 100
ip -6 route add ::ffff:6:8:23:0/29 via ::ffff:6:8:23:2 table 200
```

```
# Добавляем правила для масок
ip -6 rule add fmark 0x11 table 100
ip -6 rule add fmark 0x33 table 200
```

```
# Маркируем ICMP-Reply
ip6tables -t mangle -A OUTPUT -s ::ffff:6:8:12:2 -d 6:8:34:4 -p icmpv6 --icmpv6-type 129 -j
MARK --set-mark 0x33
ip6tables -t mangle -A OUTPUT -s ::ffff:6:8:23:2 -d ::ffff:6:8:34:4 -p icmpv6 --icmpv6-type 129
-j MARK --set-mark 0x11
```

Для проверки был отправлен один пакет с 4 компьютера на адрес ::ffff:6:8:12:2 “ping6 -c1 ::ffff:6:8:12:2” (Рис 41).

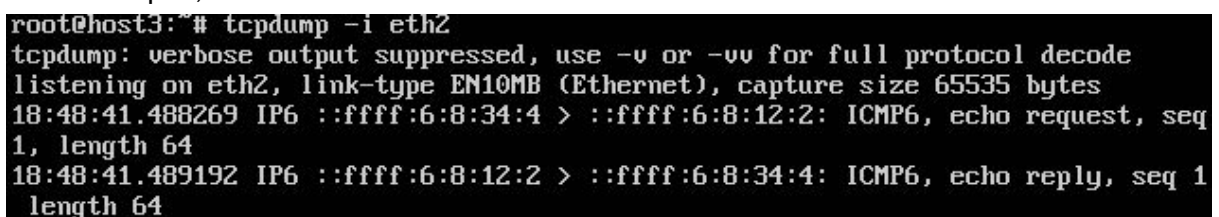


```
root@host4:~# ping6 -c1 ::ffff:6:8:12:2
PING ::ffff:6:8:12:2 (::ffff:6:8:12:2) 56 data bytes
64 bytes from ::ffff:6:8:12:2: icmp_seq=1 ttl=63 time=1.66 ms

--- ::ffff:6:8:12:2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.669/1.669/1.669/0.000 ms
```

Рисунок 41

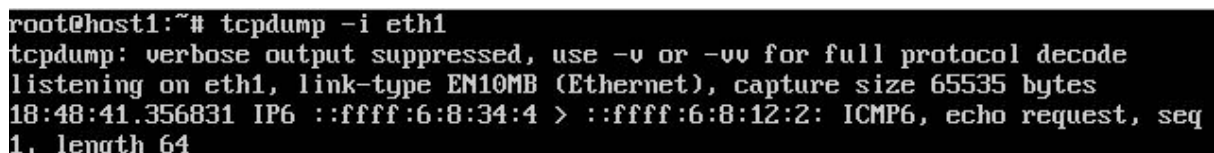
На компьютере 3 был запущен “tcpdump -i eth2” (Рисунок 42). Сюда пришел сначала ICMP6-запрос, потом ICMP6-ответ::



```
root@host3:~# tcpdump -i eth2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 65535 bytes
18:48:41.488269 IP6 ::ffff:6:8:34:4 > ::ffff:6:8:12:2: ICMP6, echo request, seq
1, length 64
18:48:41.489192 IP6 ::ffff:6:8:12:2 > ::ffff:6:8:34:4: ICMP6, echo reply, seq 1
length 64
```

Рисунок 42

На компьютере 1 был запущен “tcpdump -i eth1” (Рис 43). Через 2 машину прошел ICMP6-запрос:



```
root@host1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
18:48:41.356831 IP6 ::ffff:6:8:34:4 > ::ffff:6:8:12:2: ICMP6, echo request, seq
1, length 64
```

Рисунок 43

На рисунке 44 (и 41, 42, 43) изображено отправдение одного пакета на адрес ::ffff:6:8:12:2 (2-й машины): при отправлении ICMP Echo Request с компьютера 4 на IP1 = ::ffff:6:8:12:2 пакет проходит через компьютер 3 и компьютер 1, а ICMP Echo Reply шел с компьютера 2 сразу через компьютер 3

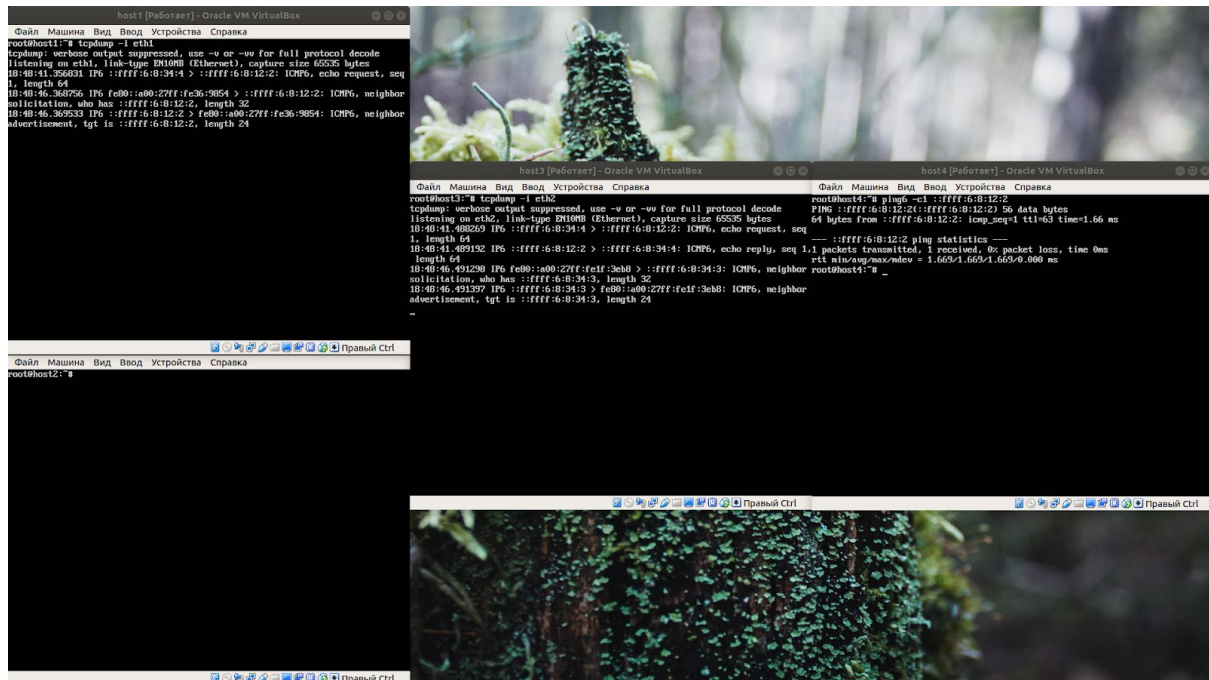


Рисунок 44

Для второй проверки с 4 машины был отправлен один пакет на адрес ::ffff:6:8:23:2 “ping -c1 ::ffff:6:8:23:2” (Рисунок 45):

```
root@host4:~# ping6 -c1 ::ffff:6:8:23:2
PING ::ffff:6:8:23:2(::ffff:6:8:23:2) 56 data bytes
64 bytes from ::ffff:6:8:23:2: icmp_seq=1 ttl=62 time=2.09 ms

--- ::ffff:6:8:23:2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 2.091/2.091/2.091/0.000 ms
```

Рисунок 45

На компьютере 3 был запущен “tcpdump -i eth2” (Рисунок 46). Сюда пришел сначала ICMP6-запрос, потом ICMP6-ответ:

```
root@host3:~# tcpdump -i eth2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 65535 bytes
18:57:55.906734 IP6 ::ffff:6:8:34:4 > ::ffff:6:8:23:2: ICMP6, echo request, seq 1, length 64
18:57:55.907850 IP6 ::ffff:6:8:23:2 > ::ffff:6:8:34:4: ICMP6, echo reply, seq 1, length 64
```

Рисунок 46

На компьютере 2 был запущен “tcpdump -i eth1” (Рис 47). Сюда пришел ICMP6-ответ :

```

root@host1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
18:57:55.775810 IP6 ::ffff:6:8:23:2 > ::ffff:6:8:34:4: ICMP6, echo reply, seq 1,
length 64

```

Рисунок 47

На рисунке 48 (а также рисунках 45-47) можно увидеть корректность работы. При отправлении ICMP6 Echo Request с компьютера 4 на IP2 = ::ffff:6:8:23:2 пакет прошел через компьютер 3 сразу на компьютер 2, а ICMP6 Echo Reply прошел с компьютера 2 на компьютер 1 и далее на компьютер 3.

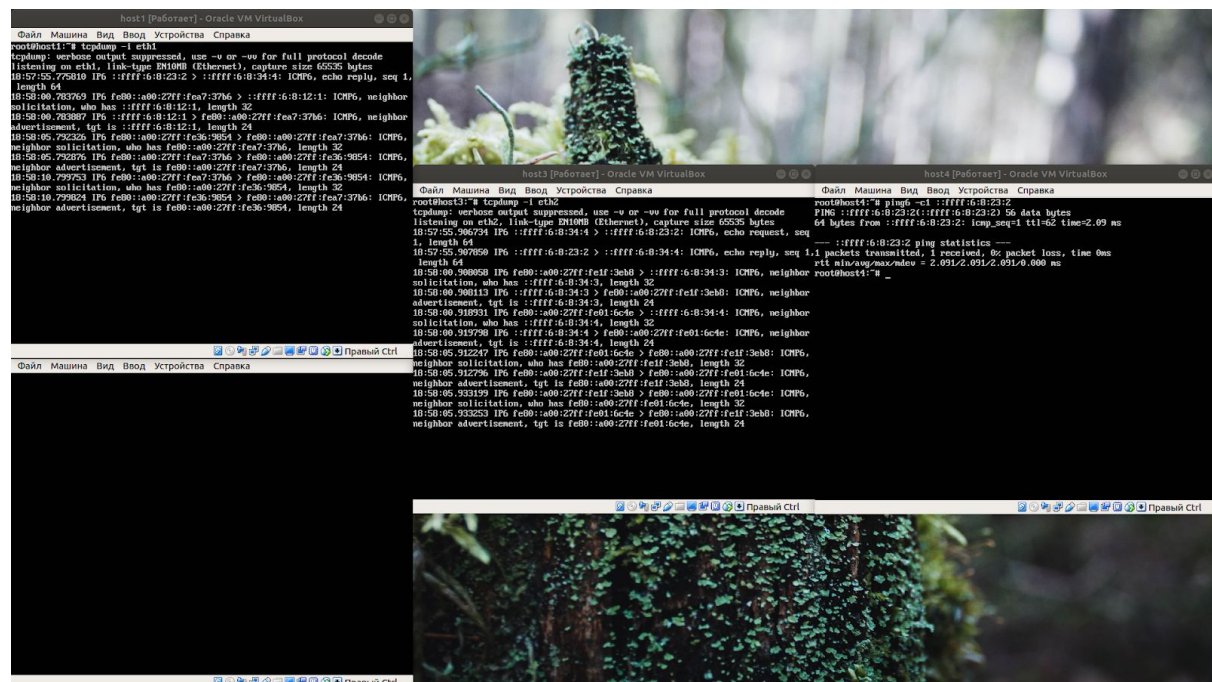


Рисунок 48

На рисунке 49 показан итоговый вывод iptables для машины 2. Как видно, у нас действительно есть 2 таблицы с разными правилами для пакетов разного типа.

```

root@host2:~# iptables -t mangle -L OUTPUT
Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
MARK       ipv6-icmp  ::ffff:6:8:12:2        ::ffff:6:8:34:4        ipv6-icmp echo
-reply MARK set 0x33
MARK       ipv6-icmp  ::ffff:6:8:23:2        ::ffff:6:8:34:4        ipv6-icmp echo
-reply MARK set 0x11

```

Рисунок 49

На рисунке 50 показан вывод ip -6 rule. Здесь указано, какими таблицами стоит пользоваться пакетам с определенной маркировкой для машины 2

```
root@host2:~# ip -6 rule
0:      from all lookup local
16383:  from all fwmark 0x11 lookup 100
16383:  from all fwmark 0x33 lookup 200
32766:  from all lookup main
```

Рисунок 50