

ЦЕЛЬ

...

ЗАДАНИЕ

...

Исходные данные

Имя: **ИВАН**, длина 4
Фамилия: **САРЖЕВСКИЙ**, длина 10
 $V_1 = 1 + ((4+10) \bmod 5) = 5$

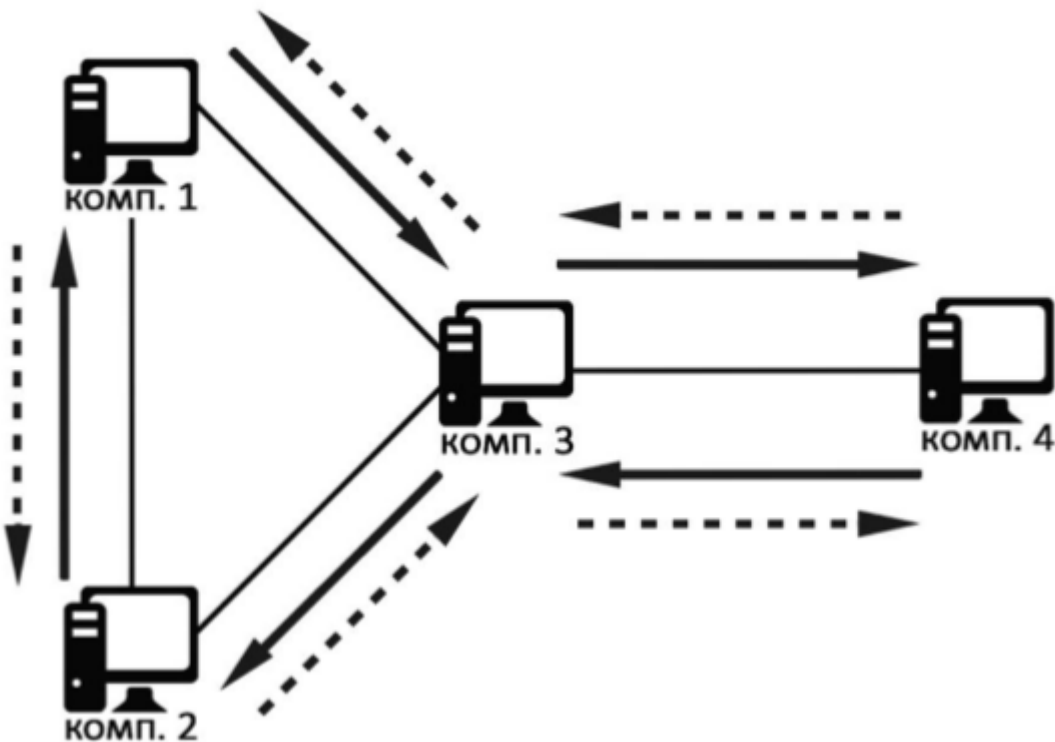


рис 1. Топология сети и схема прохождения трафика для варианта 5

На рисунке 1 изображена топология сети и требуемый путь прохождения сетевых пакетов. С компьютера 4 посылается ICMP Echo Request на адрес, который не существует в данной сети. На компьютерах 1, 2 и 3 должны быть настроены таблицы маршрутизации и правила NAT таким образом, чтобы пакет поочередно прошел через компьютеры 3, 2, 1 и снова пройдя через компьютер 3 пришел на компьютер 4 (сплошные линии на рисунке 4.4.5) с IP заголовком, в котором IP адрес источника и IP адрес назначения будут поменяны местами. Таким образом, компьютер 4 получит ICMP Echo Request на

свой локальный адрес и ответит на него. ICMP Echo Reply должен пройти обратный путь (4->3->1->2->3->4) и прийти на компьютер 4 (штриховые линии на рисунке 4.9) с поменяными местами адресами источника и назначения. В результате выполнения команды ping должна быть выведена информация об успешном выполнении. Т.о. компьютер 4 сам отвечает на собственные ICMP запросы, однако пакет проходит через внешнюю сеть маршрутизаторов

Выбор IPv4 и IPv6 адресов

4.10.X.Y/M:

Подсеть s1_s3:

- **s1:** 4.10.13.1/30 -- ::4.10.13.1/126
- **s3:** 4.10.13.2/30 -- ::4.10.13.2/126

Подсеть s1_s2:

- **s1:** 4.10.12.1/30 -- ::4.10.12.1/126
- **s2:** 4.10.12.2/30 -- ::4.10.12.2/126

Подсеть s2_s3:

- **s2:** 4.10.23.1/30 -- ::4.10.23.1/126
- **s3:** 4.10.23.2/30 -- ::4.10.23.2/126

Подсеть s3_s4:

- **s3:** 4.10.34.1/30 -- ::4.10.34.1/126
- **s4:** 4.10.34.2/30 -- ::4.10.34.2/126

Настройка сети

station_1

```
ip link set eth2 up                # включаем s1_s3
ip a add 4.10.13.1/30 dev eth2      # ipv4 s1_s3
ip -6 a add ::4.10.13.1/126 dev eth2 # ipv6 s1_s3

ip link set eth1 up                # включаем s1_s2
ip a add 4.10.12.1/30 dev eth1      # ipv4 s1_s2
ip -6 a add ::4.10.12.1/126 dev eth1 # ipv6 s1_s2
```

station_2

```
ip link set eth1 up                # включаем s1_s2
ip a add 4.10.12.2/30 dev eth1      # ipv4 s1_s2
ip -6 a add ::4.10.12.2/126 dev eth1 # ipv6 s1_s2
```

```
ip link set eth2 up                # включаем s2_s3
ip a add 4.10.23.1/30 dev eth2      # ipv4 s2_s3
ip -6 a add ::4.10.23.1/126 dev eth2 # ipv6 s2_s3
```

station_3

```
ip link set eth1 up                # включаем s1_s3
ip a add 4.10.13.2/30 dev eth1      # ipv4 s1_s3
ip -6 a add ::4.10.13.2/126 dev eth1 # ipv6 s1_s3

ip link set eth2 up                # включаем s3_s4
ip a add 4.10.34.1/30 dev eth2      # ipv4 s3_s4
ip -6 a add ::4.10.34.1/126 dev eth2 # ipv6 s3_s4

ip link set eth3 up                # включаем s2_s3
ip a add 4.10.23.2/30 dev eth3      # ipv4 s2_s3
ip -6 a add ::4.10.23.2/126 dev eth3 # ipv6 s2_s3
```

station_4

```
ip link set eth1 up                # включаем s3_s4
ip a add 4.10.34.2/30 dev eth1      # ipv4 s3_s4
ip -6 a add ::4.10.34.2/126 dev eth1 # ipv6 s3_s4
```

station_3

```
sysctl -w net.ipv4.ip_forward=1    # включаем ip_forward
sysctl -w "net.ipv4.conf.all.rp_filter=0" # отключаем фильтр пакетов, где
                                           # dest недостижим из текущего интерфейса

# включаем forwarding для ipv6
echo "net.ipv6.conf.all.forwarding=1" >> /etc/sysctl.conf
sysctl -p /etc/sysctl.conf
```

station_1

```
# добавляем роут 1->(3)->4
ip ro add 4.10.34.2 via 4.10.13.2
ip -6 ro a ::4.10.34.2/126 via ::4.10.13.2

apt-get install netcat6
```

station_2

```
# добавляем роут 2->(3)->4
ip ro add 4.10.34.2 via 4.10.23.2
ip -6 ro a ::4.10.34.2/126 via ::4.10.23.2
```

station_4

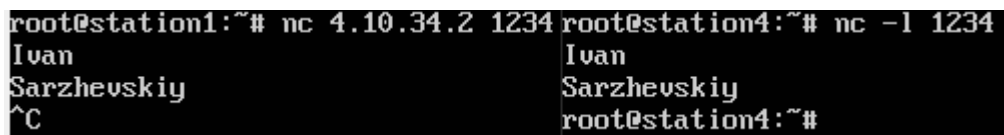
```
# добавляем роут 4->(3)->1
ip ro add 4.10.13.1 via 4.10.34.1
ip -6 ro a ::4.10.13.1/126 via ::4.10.34.1

# добавляем роут 4->(3)->2
ip ro add 4.10.23.2 via 4.10.34.1
ip -6 ro a ::4.10.23.2/126 via ::4.10.34.1

apt-get install netcat6
```

Проверка

Проверим настройку сети с помощью утилиты **nc**, в роли клиента будет компьютер 4, а сервера - компьютер 1.



```
root@station1:~# nc 4.10.34.2 1234
Ivan
Sarzhevskiy
^C
root@station4:~# nc -l 1234
Ivan
Sarzhevskiy
root@station4:~#
```

рис. 2 Результат выполнения команды **nc**, **ipv4**



```
root@station4:~# nc6 -l -p 1234
Ivan
Sarzhevskiy
root@station1:~# nc6 ::4.10.34.2 1234
Ivan
Sarzhevskiy
```

рис. 2 Результат выполнения команды **nc**, **ipv6**

Реализация простого Firewall'a

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

```
iptables -A OUTPUT -p tcp --destination-port 5555 -j DROP
```

```

root@station1:~# nc 4.10.34.2 5555
Ivan
Sarzhhevskiy
^C
root@station1:~# iptables -A OUTPUT -p tcp --destination-port 5555 -j DROP
root@station1:~# nc 4.10.34.2 5555
Ivan
Sarzhhevskiy
^C
root@station1:~# iptables -D OUTPUT -p tcp --destination-port 5555 -j DROP
root@station1:~# nc 4.10.34.2 5555
Ivan
Sarzhhevskiy
^C
root@station1:~#

```

рис. 3 firewall первое правило, source

```

root@station4:~# nc -l 5555
Ivan
Sarzhhevskiy
root@station4:~# nc -l 5555
^C
root@station4:~# nc -l 5555
Ivan
Sarzhhevskiy
root@station4:~#

```

рис. 4 firewall первое правило, destination

2. Запретить приём только тех пакетов, которые отправлены с UDP-порта утилиты nc.

```
iptables -A INPUT -p tcp --source-port 5555 -j DROP
```

```

root@station1:~# nc -p 5555 -u 4.10.34.2 5555
Ivan
Sarzhhevskiy
^C
root@station1:~# nc -p 6666 -u 4.10.34.2 5555
Ivan
Sarzhhevskiy
^C
root@station1:~#

```

рис. 5 firewall второе правило, source

```

root@station4:~# iptables -A INPUT -p udp --source-port 5555 -j DROP
root@station4:~# nc -l -u 5555
^C
root@station4:~# nc -l -u 5555
Ivan
Sarzhhevskiy
^C
root@station4:~#

```

рис. 6 firewall второе правило, destination

3. Запретить передачу только тех пакетов, которые отправлены с IP-адреса компьютера А.

```
iptables -A OUTPUT -s 4.10.13.1 -j DROP
```

```
root@station1:~# iptables -A OUTPUT -s 4.10.13.1 -j DROP
root@station1:~# ping 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 4.10.34.2 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3010ms

root@station1:~# iptables -D OUTPUT -s 4.10.13.1 -j DROP
root@station1:~# ping 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 56(84) bytes of data.
64 bytes from 4.10.34.2: icmp_seq=1 ttl=63 time=0.272 ms
64 bytes from 4.10.34.2: icmp_seq=2 ttl=63 time=1.06 ms
64 bytes from 4.10.34.2: icmp_seq=3 ttl=63 time=0.999 ms
^C
--- 4.10.34.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.272/0.777/1.062/0.359 ms
```

рис. 7 firewall третье правило, source

4. Запретить приём только техпакетов, которые отправлены на IP-адрес компьютера Б.

```
iptables -A INPUT -d 4.10.34.2 -j DROP
```

```
root@station1:~# ping 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 56(84) bytes of data.
^C
--- 4.10.34.2 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3023ms

root@station1:~# ping 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 56(84) bytes of data.
64 bytes from 4.10.34.2: icmp_seq=1 ttl=63 time=0.432 ms
64 bytes from 4.10.34.2: icmp_seq=2 ttl=63 time=0.997 ms
64 bytes from 4.10.34.2: icmp_seq=3 ttl=63 time=5.11 ms
64 bytes from 4.10.34.2: icmp_seq=4 ttl=63 time=1.09 ms
^C
--- 4.10.34.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 0.432/1.911/5.119/1.869 ms
root@station1:~#
```

рис. 8 firewall четвертое правило, source

```
root@station4:~# iptables -A INPUT -d 4.10.34.2 -j DROP
root@station4:~# iptables -D INPUT -d 4.10.34.2 -j DROP
```

рис. 9 firewall четвертое правило, destination

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

```
iptables -A INPUT -m ttl --ttl-lt 10 -m length --length 1001:65535 -p icmp -j DROP
iptables -A OUTPUT -m ttl --ttl-lt 10 -m length --length 1001:65535 -p icmp -j DROP
```

```
root@station1:~# ping -s 1500 -t 3 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 1500(1528) bytes of data.
^C
--- 4.10.34.2 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3000ms

root@station1:~# ping -s 1500 -t 3 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 1500(1528) bytes of data.
^C
--- 4.10.34.2 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4008ms

root@station1:~# ping -s 100 -t 3 4.10.34.2
PING 4.10.34.2 (4.10.34.2) 100(128) bytes of data.
108 bytes from 4.10.34.2: icmp_seq=1 ttl=63 time=0.425 ms
108 bytes from 4.10.34.2: icmp_seq=2 ttl=63 time=1.01 ms
108 bytes from 4.10.34.2: icmp_seq=3 ttl=63 time=0.994 ms
108 bytes from 4.10.34.2: icmp_seq=4 ttl=63 time=1.09 ms
^C
--- 4.10.34.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 0.425/0.881/1.097/0.268 ms
root@station1:~#
```

рис. 10 firewall пятое правило, source

```
root@station4:~# iptables -A INPUT -m length --length 1001:65535 -m ttl --ttl-lt 10 -p icmp -j DROP
root@station4:~# iptables -A OUTPUT -m length --length 1001:65535 -m ttl --ttl-lt 10 -p icmp -j DROP

root@station4:~# ping -s 1500 -t 3 4.10.13.1
PING 4.10.13.1 (4.10.13.1) 1500(1528) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 4.10.13.1 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1009ms

root@station4:~# ping -s 100 -t 3 4.10.13.1
PING 4.10.13.1 (4.10.13.1) 100(128) bytes of data.
108 bytes from 4.10.13.1: icmp_seq=1 ttl=63 time=0.438 ms
108 bytes from 4.10.13.1: icmp_seq=2 ttl=63 time=0.986 ms
108 bytes from 4.10.13.1: icmp_seq=3 ttl=63 time=0.990 ms
^C
--- 4.10.13.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.438/0.804/0.990/0.261 ms
root@station4:~#
```

Вариативная часть

Пусть несуществующий ip - 4.10.44.4/30.

station_4

```
ip link set eth1 up                # включаем s3_s4
ip a add 4.10.34.2/30 dev eth1      # ipv4 s3_s4

ip ro add 4.10.44.4 via 4.10.31.1   # роут 4->(3)->\X\
```

station_3

```
ip link set eth1 up                # включаем s1_s3
ip link set eth2 up                # включаем s3_s4
ip link set eth3 up                # включаем s2_s3

ip a add 4.10.13.2/30 dev eth1      # ipv4 s1_s3
ip a add 4.10.34.1/30 dev eth2      # ipv4 s3_s4
ip a add 4.10.23.2/30 dev eth3      # ipv4 s2_s3

# 0xbada55 путь ping
iptables -t mangle -A PREROUTING -p icmp --icmp-type ping -s 4.10.34.2 -j MARK --
set-mark 0xbada55
ip ro add table 666 4.10.34.2 via 4.10.13.1 dev eth1
ip rule add fwmark 0xbada55 table 666

# deadd00d путь pong
iptables -t mangle -A PREROUTING -p icmp --icmp-type pong -s 4.10.34.2 -j MARK --
set-mark 0xdeadd00d
ip ro add table 999 4.10.34.2 via 4.10.23.2 dev eth3
ip rule add fwmark 0xdeadd00d table 999

# меняем местами source и destination
iptables -t nat -A PREROUTING -p icmp -s 4.10.34.2 -d 4.10.44.4 -j DNAT --to
4.10.34.2
iptables -t nat -A POSTROUTING -p icmp -s 4.10.34.2 -d 4.10.34.2 -j SNAT --to
4.10.44.4

# перенаправляем ping на station_4
iptables -t mangle -A PREROUTING -p icmp --icmp-type ping -s 4.10.44.4 -j TEE --
gateway 4.10.34.2

sysctl -w net.ipv4.ip_forward=1    # включаем ip_forward
sysctl -w "net.ipv4.conf.all.rp_filter=0" # отключаем фильтр пакетов, где
```


интерфейсаа

dest недостижим из текущего

station_1

```
ip link set eth2 up                # включаем s1_s3
ip link set eth1 up                # включаем s1_s2

ip a add 4.10.13.1/30 dev eth2      # ipv4 s1_s3
ip a add 4.10.12.1/30 dev eth1      # ipv4 s1_s2

# перенаправляем ping на station_2
iptables -t mangle -A PREROUTING -p icmp --icmp-type ping -s 4.10.44.4 -j TEE --
gateway 4.10.12.2
```

station_2

```
ip link set eth1 up                # включаем s1_s2
ip link set eth2 up                # включаем s2_s3

ip a add 4.10.12.2/30 dev eth1      # ipv4 s1_s2
ip a add 4.10.23.1/30 dev eth2      # ipv4 s2_s3

# перенаправляем ping на station_3
iptables -t mangle -A PREROUTING -p icmp --icmp-type ping -s 4.10.44.4 -j TEE --
gateway 4.10.23.2
# перенаправляем pong на station_1
iptables -t mangle -A PREROUTING -p icmp --icmp-type pong -s 4.10.44.4 -j TEE --
gateway 4.10.12.1
```