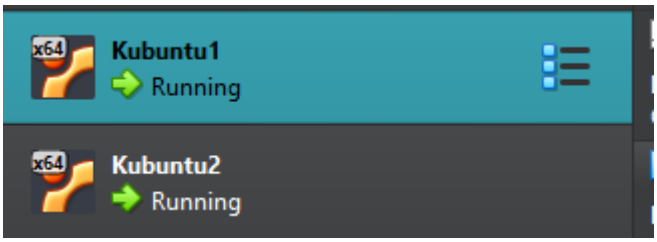
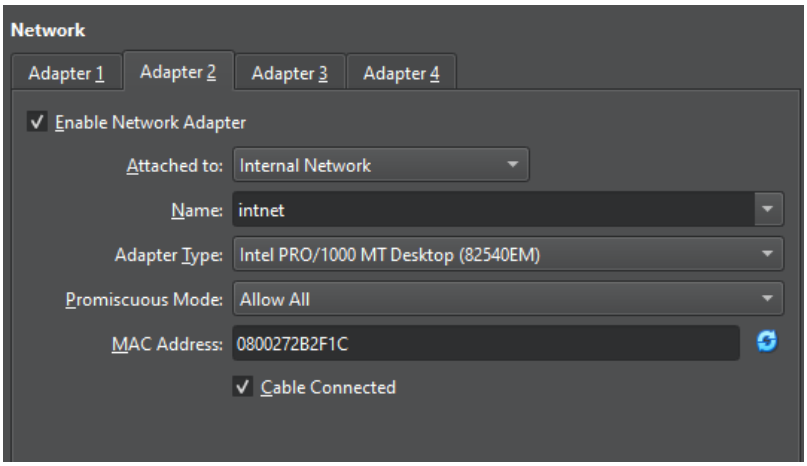
Практическая работа 6

Павлович Джурдже М4105

1. Установить 2 виртуалные машины. Я установил Kubuntu 24.04.01 на Virtualbox.



2. Настроить на них первый сетевой интерфейс - NAT по умолчанию, а второй – типа “внутренняя сеть”. Включите “неразборчивый режим”



3.**В самой ВМ настроить IP адрес ВМ на 10.10.10.3/29 и 10.10.10.4/29**

a:

ВМ1:

```
djurdje1@kubuntu1:~$ sudo ip addr add 10.10.10.3/29 dev enp0s8
[sudo] password for djurdje1:
djurdje1@kubuntu1:~$ sudo ip link set enp0s8 up
```

ВМ2:

```
djurdje2@kubuntu2:~$ sudo ip addr add 10.10.10.4/29 dev enp0s8
[sudo] password for djurdje2:
djurdje2@kubuntu2:~$ sudo ip link set enp0s8 up
```

b: **Оформить /etc/network/interfaces,*

ВМ1:

```
#loopback network interface
auto lo
iface lo inet loopback

#NAT network interface
auto enp0s3
iface enp0s3 inet dhcp

#internal network interface
auto enp0s8
iface enp0s8 inet static
    address 10.10.10.3
    netmask 255.255.255.248
    gateway 10.10.10.1
```

ВМ2:

```
  GNU nano 7.2
#loopback network interface
auto lo
iface lo inet loopback

#NAT network interface
auto enp0s3
iface enp0s3 inet dhcp

#internal network interface
auto enp0s8
iface enp0s8 inet static
    address 10.10.10.4
    netmask 255.255.255.248
    gateway 10.10.10.1
```

4. IP v4. Проверьте, что в машине включен форвардинг IP v4 пакетов

Потом нужно сделать перенаправление IPv4 постоянным: там уже есть команда, ее надо только откоментировать.

```
djurdje1@kubuntu1:~$ sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
djurdje1@kubuntu1:~$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
djurdje1@kubuntu1:~$ sudo nano /etc/sysctl.conf
```

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Проверка:

```
djurdje1@kubuntu1:~$ sudo sysctl -p
net.ipv4.ip_forward = 1
djurdje1@kubuntu1:~$
```

5. **Установите пакет wireguard**

```
djurdje2@kubuntu2:~$ sudo apt install wireguard
Reading package lists... Done
```

ВМ1:

```
djurdje1@kubuntu1:~$ sudo sh -c 'wg genkey | tee /etc/wireguard/privatekey | wg pubkey | tee /etc/wireguard/publi
ckey'
mMPwOYTQALC07dcYkD4hcydad1zJhD2E2DJkpKMMETc=
djurdje1@kubuntu1:~$
```

ВМ2:

```
djurdje2@kubuntu2:~$ sudo sh -c 'wg genkey | tee /etc/wireguard/privatekey | wg pubkey | tee /etc/wireguard/pu
blickey'
pI9CQjKCDHsA5k16YcxUTwAgUPPr3IfysHnH9t/5DUw=
```

7. Сделайте конфиг в /etc/wireguard/wg0.conf

```
[Interface]

Address = 172.16.10.1/29

PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE

PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -t nat -D POSTROUTING -o enp0s3 -j MASQUERADE

ListenPort = 51820

PrivateKey = kG8OPKgMlzbKJIoSH3rvdlQVq84jXFQ/ttW3+NmDcHk=

[Peer]
PublicKey = pI9CQjKCDHsA5k16YcxUTwAgUPPr3IfysHnH9t/5DUw=
AllowedIPs = 172.16.10.3/32
```

Во флаге `-o` указывается интерфейс, который будет использоваться для туннеля в настройках WireGuard. Обычно это внешний сетевой интерфейс.

`PostUp` — это набор команд, которые выполняются после того, как интерфейс WireGuard был поднят, а PostDown - набор команд, которые выполняются после того опущения интерфейса WireGuard.

`iptables -A FORWARD -i %i -j ACCEPT`:

- Эта команда добавляет правило в цепочку `FORWARD` iptables, разрешающее передачу пакетов, поступающих на интерфейс WireGuard (`%i`). Это необходимо для маршрутизации трафика через туннель.
  `iptables -t nat -A POSTROUTING -o <INTERFACE> -j MASQUERADE`:
- Эта команда добавляет правило в таблицу NAT (Network Address Translation), которое выполняет **маскарадинг** для исходящих пакетов через указанный внешний интерфейс

1. Запустите туннель командой wg-quick up wg0
2. Проверьте статус туннеля командой wg show wg0
3. Приложите вывод в отчет

```
interface: wg0
  public key: mMPwOYTQALC07dcYkD4hcydad1zJhD2E2DJkpKMMETc=
  private key: (hidden)
  listening port: 51820

peer: pI9CQjKCDHsA5k16YcxUTwAgUPPr3IfysHnH9t/5DUw=
  endpoint: 10.10.10.4:49600
  allowed ips: 172.16.10.3/32
  latest handshake: 13 seconds ago
  transfer: 7.08 KiB received, 6.96 KiB sent
```

ВМ2:

```
  GNU nano 7.2                          /etc/wireguard/wg0.conf
[Interface]

PrivateKey = +MI8hjwa9FKWvMyxlZl3Ox4IhQCMs0QlvVLc4SpkMGs=

Address = 172.16.10.3/29

PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE

PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -t nat -D POSTROUTING -o enp0s3 -j MASQUERADE

[Peer]

PublicKey = mMPwOYTQALC07dcYkD4hcydad1zJhD2E2DJkpKMMETc=

Endpoint = 10.10.10.3:51820

AllowedIPs = 172.16.10.0/29

PersistentKeepalive = 20
```

```
djurdje2@kubuntu2:~$ sudo wg show wg0
interface: wg0
  public key: pI9CQjKCDHsA5k16YcxUTwAgUPPr3IfysHnH9t/5DUw=
  private key: (hidden)
  listening port: 49600

peer: mMPwOYTQALC07dcYkD4hcydad1zJhD2E2DJkpKMMETc=
  endpoint: 10.10.10.3:51820
  allowed ips: 172.16.10.0/29
  transfer: 0 B received, 296 B sent
  persistent keepalive: every 20 seconds
djurdje2@kubuntu2:~$
```

Проверка на ВМ1:

```
djurdje1@kubuntu1:~$ sudo nano /etc/wireguard/wg0.conf
djurdje1@kubuntu1:~$ ping 172.16.10.3
PING 172.16.10.3 (172.16.10.3) 56(84) bytes of data.
64 bytes from 172.16.10.3: icmp_seq=1 ttl=64 time=0.886 ms
64 bytes from 172.16.10.3: icmp_seq=2 ttl=64 time=0.342 ms
64 bytes from 172.16.10.3: icmp_seq=3 ttl=64 time=0.392 ms
64 bytes from 172.16.10.3: icmp_seq=4 ttl=64 time=0.410 ms
64 bytes from 172.16.10.3: icmp_seq=5 ttl=64 time=0.415 ms
64 bytes from 172.16.10.3: icmp_seq=6 ttl=64 time=0.382 ms
```

Проверка на ВМ2:

```
djurdje2@kubuntu2:~$ ping 172.16.10.1
PING 172.16.10.1 (172.16.10.1) 56(84) bytes of data.
64 bytes from 172.16.10.1: icmp_seq=1 ttl=64 time=0.385 ms
64 bytes from 172.16.10.1: icmp_seq=2 ttl=64 time=0.362 ms
64 bytes from 172.16.10.1: icmp_seq=3 ttl=64 time=0.383 ms
64 bytes from 172.16.10.1: icmp_seq=4 ttl=64 time=0.412 ms
64 bytes from 172.16.10.1: icmp_seq=5 ttl=64 time=0.428 ms
64 bytes from 172.16.10.1: icmp_seq=6 ttl=64 time=0.370 ms
```

15. Поднять на втором сервере базу данных Postgres, сделать так, чтобы она была доступна только по адресу тунеля. Показать, что с первого сервера получится сделать подключение.

Установка PostgreSQL (на VM2): (он установится и запустится автоматически)

```
sudo apt update
sudo apt install postgresql
```

- Откройте файл (путь может отличаться, например у меня postgresql 16 поэтаму мой путь - `/etc/postgresql/14/main/postgresql.conf`).
- Найдите строку `#listen_adress` которая закоментированная и измените ее.

```
# - Connection Settings -

#listen_addresses = 'localhost'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for
```

```
listen_addresses = '172.16.10.3'
                                        # comma-separated list of addresses;
```

Сейчас PostgreSQL слушать только на адресе WireGuard.

В файле `/etc/postgresql/16/main/pg_hba.conf` нужно добавить разрешение для сети WireGuard:

```
host    replication     all             ::1/128                 scram-sha-256
host    all             all             172.16.10.0/29          md5
```

Также нужно после этого перезапустить PostGreSQL.Теперь PostgreSQL слушает только на `172.16.10.3:5432`.

Проверка соединения с первой машины (VM1):

Подключитесь по туннелю:

```
psql -h 172.16.10.3 -U postgres -d postgres
```

```
djurdje1@kubuntu1:~$ psql -h 172.16.10.3 -U postgres -d postgres
Password for user postgres:
psql (16.6 (Ubuntu 16.6-0ubuntu0.24.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

postgres=#
```

16.**Запишите подключение с использованием команды tcpdump, полученный файл откройте в WireShark. Покажите трафик, который шел через тунель, как выглядят пакеты**

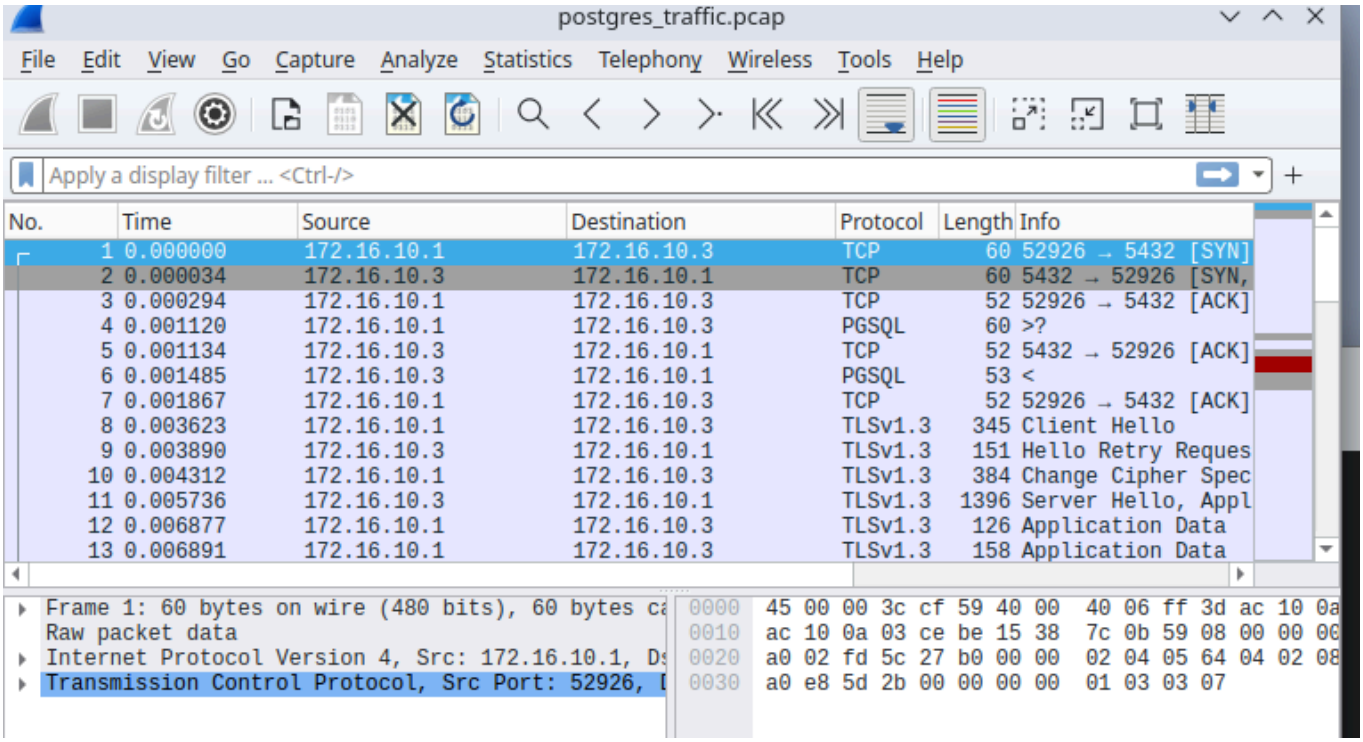
На второй машине (VM2), где работает PostgreSQL, запустил команду:
sudo tcpdump -i wg0 -w postgres_traffic.pcap

Параллельно, на первой машине (VM1) мы подключились к PostgreSQL и выполнили несколько SQL-команд (например, \dt, \t)

```
djurdje2@kubuntu2:~$ sudo tcpdump -i wg0 -w postgres_traffic.pcap
tcpdump: listening on wg0, link-type RAW (Raw IP), snapshot length 262144 bytes
^C44 packets captured
44 packets received by filter
0 packets dropped by kernel
djurdje2@kubuntu2:~$
```

После завершения захвата трафика, нужно установить Wireshark.

```
sudo apt install wireshark
```



17. **Написать iptables правило, заменяющее ДНС Google (8.8.8.8) на ДНС Яндекса (77.88.8.8). Найдите способ доказать, что правило работает (подсказка; счетчики)**

**На машине (в этом случае BM1) добавляем правило:

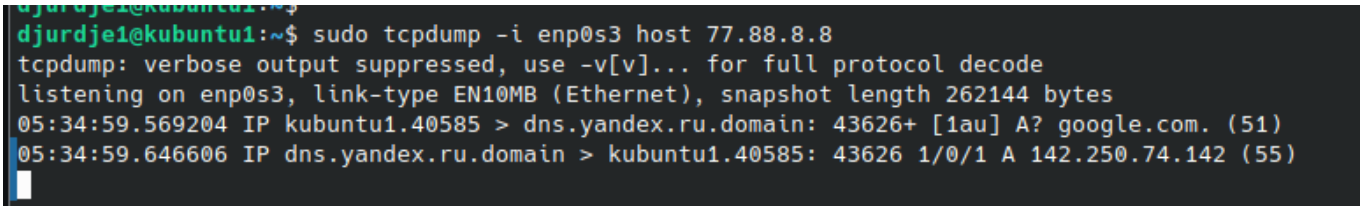
```
sudo iptables -t nat -A OUTPUT -p udp -d 8.8.8.8 --dport 53 -j DNAT --to-
destination 77.88.8.8`
```

Далее был выполнен DNS-запрос:

```
dig @8.8.8.8 google.com
```

Проверка работы правила iptables через tcpdump:

выполнен захват сетевого трафика с помощью команды:



Через эту команду видно что правило iptables успешно перенаправило трафик на 8.8.8.8 к 77.88.8.8, и запросы DNS обрабатывались сервером Яндекса.

17.**Напишите iptables правило, запрещающее вашей машине подключаться на сайт itmo.ru. 18.Докажите работу правила**

Сначала нужно узнать IP сайта itmo.ru. Можно сделать с командой

```
dig itmo.ru
```

В секции ANSWER указан IP адрес.

```
;itmo.ru.                       IN      A

;; ANSWER SECTION:
itmo.ru.                4885    IN      A       51.250.120.146
```

Добавление правила iptables:

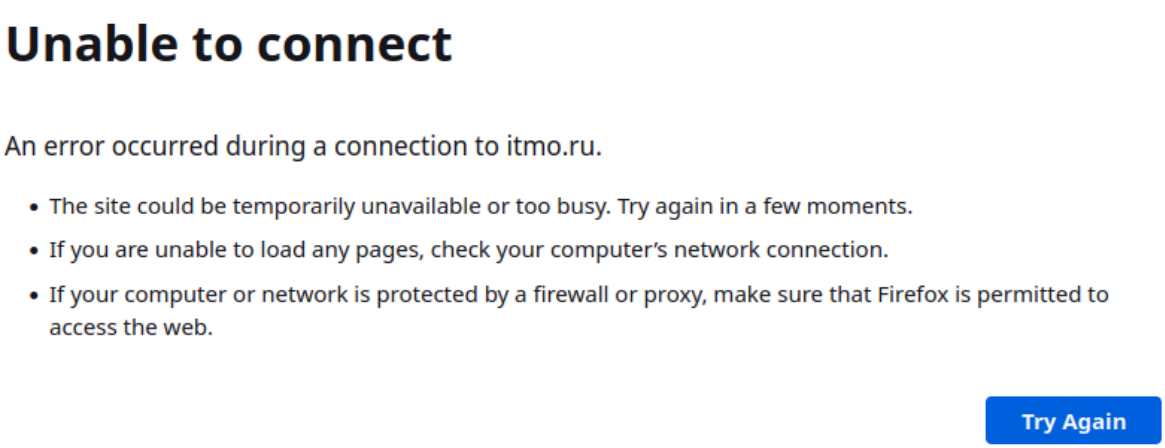
```
sudo iptables -A OUTPUT -p tcp -d 51.250.120.146 -j REJECT
```

```
djurdje1@kubuntu1:~$ sudo iptables -A OUTPUT -p tcp -d 51.250.120.146 -j REJECT
[sudo] password for djurdje1:
djurdje1@kubuntu1:~$
```

`-j REJECT`: Запрещает соединение и отправляет ответ об отказе.

Проверка:

```
djurdje1@kubuntu1:~$ curl itmo.ru
curl: (7) Failed to connect to itmo.ru port 80 after 98 ms: Couldn't connect to server
djurdje1@kubuntu1:~$
```



На второй машине где не добавил правило можно подключится: