

Лабораторная работа № 5

Простые сети в GNS3. Анализ трафика

Абд эль хай мохамад

Содержание

Цель работы	2
Задание	2
Выполнение лабораторной работы	2
1. Моделирование простейшей сети на базе коммутатора в GNS3	2
2. Анализ трафика в GNS3 посредством Wireshark	4
3. Моделирование простейшей сети на базе маршрутизатора FRR в GNS3	9
4. Моделирование простейшей сети на базе маршрутизатора VyOS в GNS3	12
Вывод	14

Список иллюстраций

Рисунок № 1 включил виртуальную машину.....	2
Рисунок № 2 простая сеть	3
Рисунок № 3 Назначенный IP-адрес для PC1.....	3
Рисунок № 4 Назначенный IP-адрес для PC2.....	4
Рисунок № 5 пинг ПК2 с ПК1.....	4
Рисунок № 6 ARP-запрос.....	5
Рисунок № 7 ARP-запрос.....	5
Рисунок № 8 Пинг PC1.....	6
Рисунок № 9 ICMP-запрос.....	6
Рисунок № 10 ICMP-ответ.....	6
Рисунок № 11 пропинговать PC 1 в режиме UDP.....	7
Рисунок № 12 UDP-запрос.....	7
Рисунок № 13 UDP-ответ.....	7
Рисунок № 14 Пинг PC 1.....	8
Рисунок № 15 TCP-трафик.....	8
Рисунок № 16 TCP-трафик.....	9
Рисунок № 17 Использование маршрутизатора FRR.....	9
Рисунок № 18 Настройка IP-адресации для PC1.....	10
Рисунок № 19 Настройка маршрутизатора FRR.....	10
Рисунок № 20 Настройка маршрутизатора FRR.....	11
Рисунок № 21 Настройка маршрутизатора FRR.....	11
Рисунок № 22 проверка связи с маршрутизатором FRR.....	12
Рисунок № 23 Настройка маршрутизатора VYOS.....	13
Рисунок № 24 Настройка маршрутизатора VYOS.....	14
Рисунок № 25 пинг маршрутизатора VYOS.....	14

Цель работы

Построение простейших моделей сети на базе коммутатора и маршрутизаторов FRR и VyOS в GNS3, анализ трафика посредством Wireshark.

Задание

- Моделирование простейшей сети на базе коммутатора в GNS3
- Анализ трафика в GNS3 посредством Wireshark
- Моделирование простейшей сети на базе маршрутизатора FRR в GNS3
- Моделирование простейшей сети на базе маршрутизатора VyOS в GNS3

Выполнение лабораторной работы

1. Моделирование простейшей сети на базе коммутатора в GNS3

1.1 Запустил GNS3 VM и GNS3. Создал новый проект под названием Lab05.

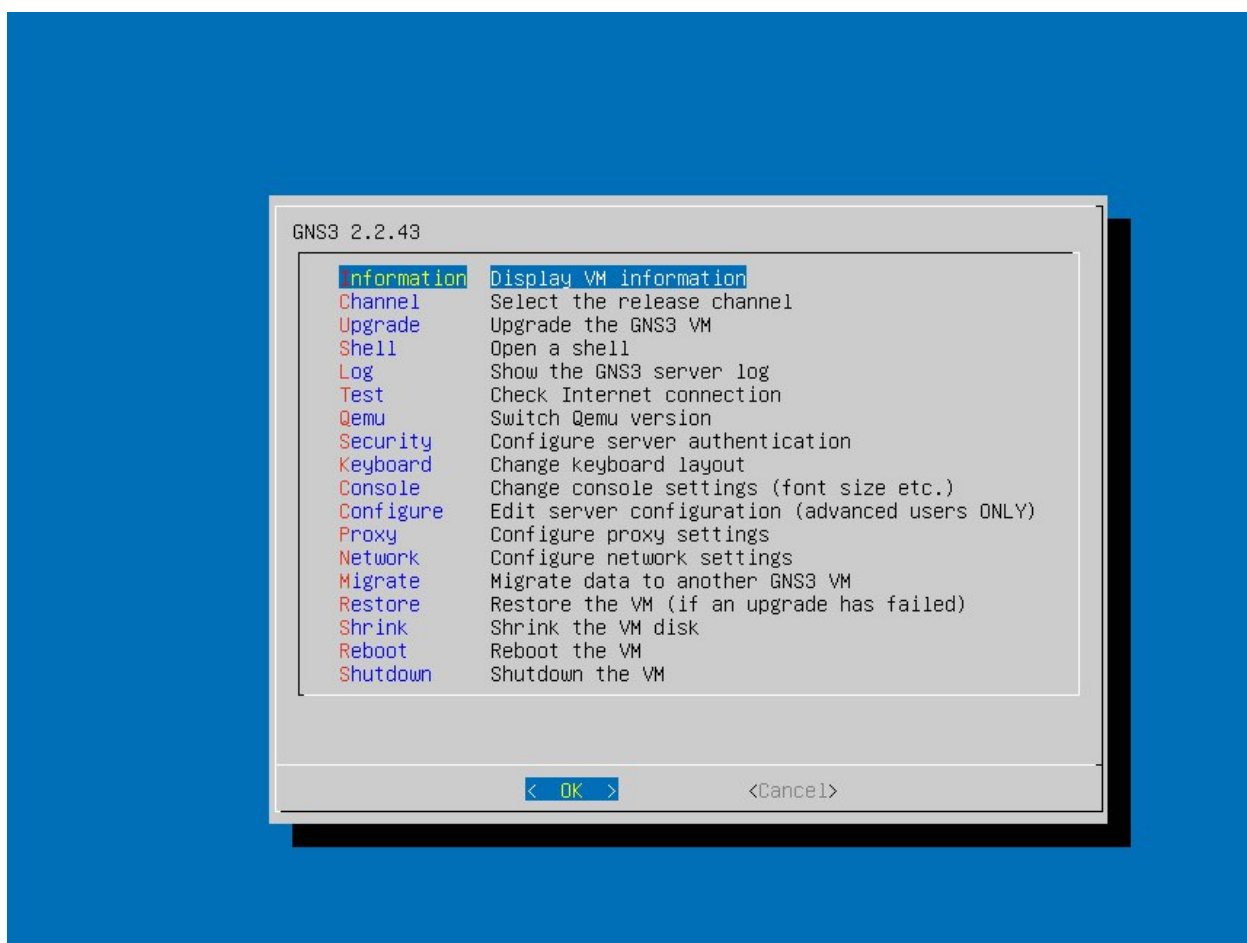


Рисунок № 1 включил виртуальную машину

1.2 В рабочей области GNS3 разместил коммутатор Ethernet и два VPCS. Изменил название устройства. Коммутатору присвоил название msk-maabedelhay-sw-01. Соединил VPCS с коммутатором. Отобразил обозначение интерфейсов соединения.



Рисунок № 2 простая сеть

1.3 Задал IP-адреса VPCS. Для просмотра синтаксиса возможных для ввода команд набрал `/?`. Для задания IP-адреса 192.168.1.11 в сети 192.168.1.0/24 ввел `ip 192.168.1.11/24 192.168.1.1` (рис. 4). 192.168.1.1 — адрес шлюза. Для сохранения конфигурации ввел команду `save` (рис. 4). Аналогичным образом задал IP-адрес 192.168.1.12 для PC-2 (рис. 5).

```
PC1> ip 192.168.1.11/24 192.168.1.1
Checking for duplicate address...
PC1 : 192.168.1.11 255.255.255.0 gateway 192.168.1.1
PC1> |
```

Рисунок № 3 Назначенный IP-адрес для PC1

```
PC2> ip 192.168.1.12/24 192.168.1.1
Checking for duplicate address...
PC2 : 192.168.1.12 255.255.255.0 gateway 192.168.1.1

PC2> save
Saving startup configuration to startup.vpc
. done

PC2> |
```

Рисунок № 4 Назначенный IP-адрес для PC2

1.4 Проверил работоспособность соединения между PC-1 и PC-2 с помощью команды ping 192.168.1.12/11 . Видим, что пинг успешно проходит.

```
PC1> ping 192.168.1.12

84 bytes from 192.168.1.12 icmp_seq=1 ttl=64 time=0.172 ms
84 bytes from 192.168.1.12 icmp_seq=2 ttl=64 time=0.249 ms
84 bytes from 192.168.1.12 icmp_seq=3 ttl=64 time=0.285 ms
|
```

Рисунок № 5 пинг ПК2 с ПК1

2. Анализ трафика в GNS3 посредством Wireshark

2.1 Запустил на соединении между PC-1 и коммутатором анализатор трафика.

Запустил Wireshark, в проекте GNS3 на соединении появился значок лупы.

2.2 Мы видим, что отображаются только ARP-запросы без ответов. Опрос происходит после поиска роутера (протокол ICMPv6). Устройствам назначаются частные MAC-адреса, которые доступны для такого типа сетевого моделирования.

3	0.050691	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
4	0.070167	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
5	1.051007	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
6	1.070857	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
7	2.051519	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
8	2.072116	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)

```

> Frame 3: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface -, id 0
> Ethernet II, Src: Private_66:68:00 (00:50:79:66:68:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Address Resolution Protocol (request/gratuitous ARP)
  - Hardware type: Ethernet (1)
  - Protocol type: IPv4 (0x0800)
  - Hardware size: 6
  - Protocol size: 4
  - Opcode: request (1)
  - [Is gratuitous: True]
  - Sender MAC address: Private_66:68:00 (00:50:79:66:68:00)
  - Sender IP address: 192.168.1.11
  - Target MAC address: Broadcast (ff:ff:ff:ff:ff:ff)
  - Target IP address: 192.168.1.11

```

Рисунок № 6 ARP-запрос

3	0.050691	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
4	0.070167	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
5	1.051007	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
6	1.070857	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
7	2.051519	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
8	2.072116	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)

```

> Frame 4: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface -, id 0
> Ethernet II, Src: Private_66:68:01 (00:50:79:66:68:01), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Address Resolution Protocol (request/gratuitous ARP)
  - Hardware type: Ethernet (1)
  - Protocol type: IPv4 (0x0800)
  - Hardware size: 6
  - Protocol size: 4
  - Opcode: request (1)
  - [Is gratuitous: True]
  - Sender MAC address: Private_66:68:01 (00:50:79:66:68:01)
  - Sender IP address: 192.168.1.12
  - Target MAC address: Broadcast (ff:ff:ff:ff:ff:ff)
  - Target IP address: 192.168.1.12

```

Рисунок № 7 ARP-запрос

2.3 В терминале PC-2 я посмотрел информацию о параметрах команды ping, набрав ping /?. Затем я сделал один эхо-запрос в режиме ICMP к узлу PC-1 ping 192.168.1.11 -с 1 - (режим ICMP является значением по умолчанию).

```
PC2> ping 192.168.1.11 -c 1

84 bytes from 192.168.1.11 icmp_seq=1 ttl=64 time=0.288 ms

PC2> |
```

Рисунок № 8 Пинг PC1

Видим эхо-запрос от PC-2 к PC-1. Развернув информацию запроса, мы видим, что второе конечное устройство, имеющее ранее установленный IP-адрес 192.168.1.12, отправляет запрос первому устройству с IP-адресом 192.168.1.11.

Также отображается ответ от PC-1 к PC-2. Развернув информацию ответа, мы видим, что первое конечное устройство, имеющее ранее установленный IP-адрес 192.168.1.11, отправляет ответ второму устройству с IP-адресом 192.168.1.12.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.12	192.168.1.11	ICMP	98	Echo (ping) request id=0x0d1e, seq=1/256, ttl=64 (reply in 2)
2	0.000149	192.168.1.11	192.168.1.12	ICMP	98	Echo (ping) reply id=0x0d1e, seq=1/256, ttl=64 (request in 1)

Header Checksum: 0xd934 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.1.12
Destination Address: 192.168.1.11

Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x12ed [correct]
[Checksum Status: Good]
Identifier (BE): 3358 (0x0d1e)
Identifier (LE): 7693 (0x1e0d)
Sequence Number (BE): 1 (0x0001)
Sequence Number (LE): 256 (0x0100)
[Response frame: 2]

Data (56 bytes)

0000 00 50 79 66 68 00 00 50 79
0010 00 54 1e 0d 00 00 40 01 d9
0020 01 0b 08 00 12 ed 0d 1e 00
0030 0e 0f 10 11 12 13 14 15 16
0040 1e 1f 20 21 22 23 24 25 26
0050 2e 2f 30 31 32 33 34 35 36
0060 3e 3f

Рисунок № 9 ICMP-запрос

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.12	192.168.1.11	ICMP	98	Echo (ping) request id=0x0d1e, seq=1/256, ttl=64 (reply in 2)
2	0.000149	192.168.1.11	192.168.1.12	ICMP	98	Echo (ping) reply id=0x0d1e, seq=1/256, ttl=64 (request in 1)

Header Checksum: 0xd934 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.1.11
Destination Address: 192.168.1.12

Internet Control Message Protocol
Type: 0 (Echo (ping) reply)
Code: 0
Checksum: 0x1aed [correct]
[Checksum Status: Good]
Identifier (BE): 3358 (0x0d1e)
Identifier (LE): 7693 (0x1e0d)
Sequence Number (BE): 1 (0x0001)
Sequence Number (LE): 256 (0x0100)
[Request frame: 1]
[Response time: 0,149 ms]

0000 00 50 79 66 68 01 00 50 79
0010 00 54 1e 0d 00 00 40 01 d9
0020 01 0c 00 00 1a ed 0d 1e 00
0030 0e 0f 10 11 12 13 14 15 16
0040 1e 1f 20 21 22 23 24 25 26
0050 2e 2f 30 31 32 33 34 35 36
0060 3e 3f

Рисунок № 10 ICMP-ответ

2.4 Теперь сделаем один эхо-запрос в режиме UDP к узлу PC-1. Для этого введите команду: ping 192.168.1.11 -2 -c 1.

```
PC2> ping 192.168.1.11 -2 -c 1

84 bytes from 192.168.1.11 udp_seq=1 ttl=64 time=0.214 ms

PC2> |
```

Рисунок № 11 пропинговать PC 1 в режиме UDP

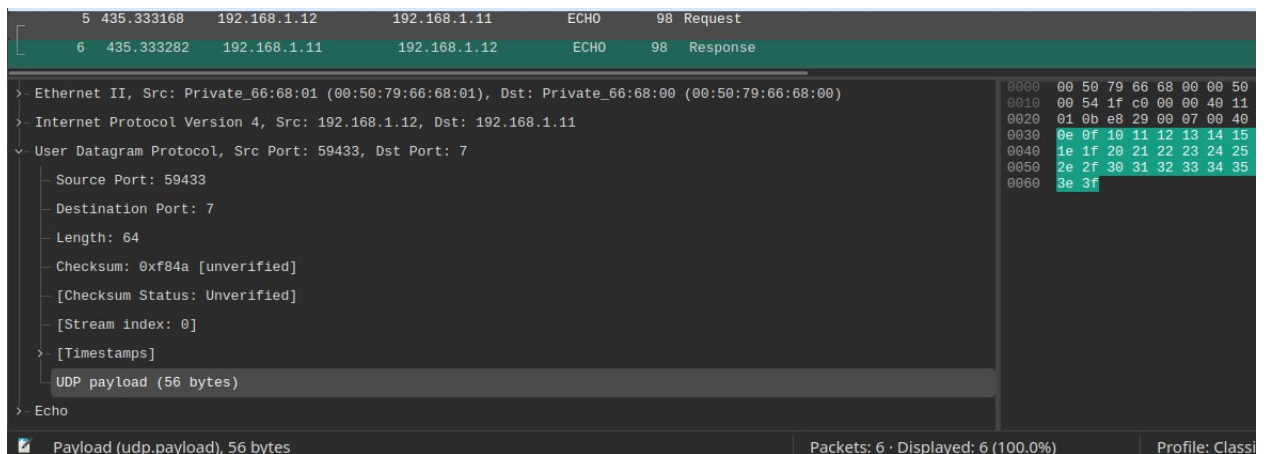


Рисунок № 12 UDP-запрос

У нас есть эхо-запрос. Развернем информацию и увидим, что запрос действительно был отправлен вторым конечным устройством ПК-2, имеющим адрес 192.168.1.12, для первого устройства ПК-1 — 192.168.1.11.

Мы можем наблюдать эхо-ответ от первого устройства 192.168.1.11 ко второму 192.168.1.12.

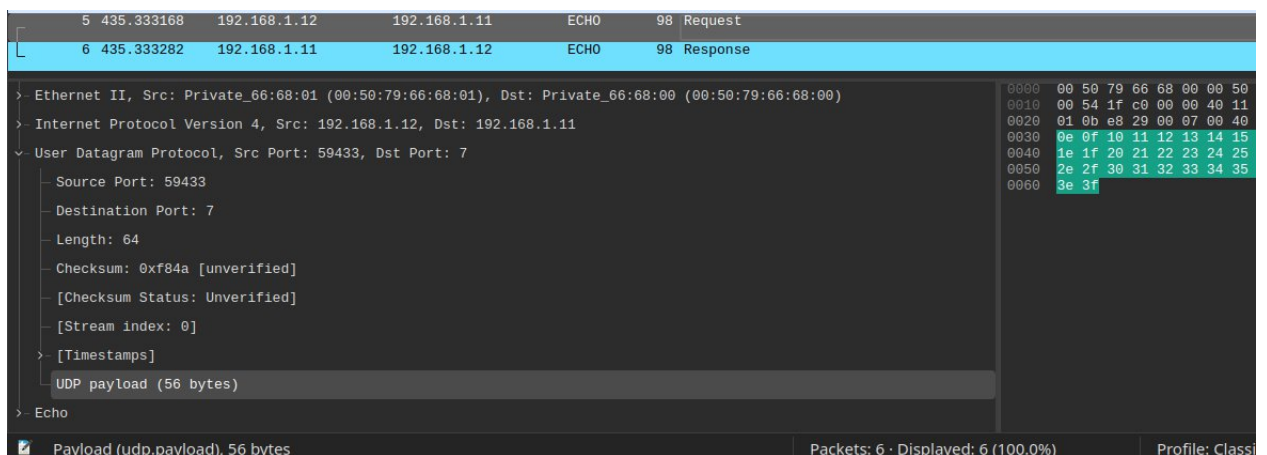


Рисунок № 13 UDP-ответ

2.5 Теперь сделаем один эхо-запрос в режиме TCP к узлу PC-1. Для этого введите команду: ping 192.168.1.11 -3 -c 1

```
PC2> ping 192.168.1.11 -3 -c 1

Connect    7@192.168.1.11 seq=1 ttl=64 time=1.051 ms
SendData   7@192.168.1.11 seq=1 ttl=64 time=1.124 ms
Close      7@192.168.1.11 seq=1 ttl=64 time=2.214 ms

PC2> |
```

Рисунок № 14 Пинг PC 1

9	783.521334	192.168.1.12	192.168.1.11	TCP	74	21043 → 7 [SYN] Seq=0 Win=2920 Len=0 MSS=1460 TSval=1697390876 TSecr=0
10	783.521498	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [SYN, ACK] Seq=0 Ack=1 Win=2920 Len=0
11	783.522413	192.168.1.12	192.168.1.11	TCP	66	21043 → 7 [ACK] Seq=1 Ack=1 Win=2920 Len=0 TSval=1697390876 TSecr=0
12	783.522613	192.168.1.12	192.168.1.11	ECHO	122	Request
13	783.522726	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [ACK] Seq=1 Ack=57 Win=2920 Len=0
14	783.523875	192.168.1.12	192.168.1.11	TCP	66	21043 → 7 [FIN, PSH, ACK] Seq=57 Ack=1 Win=2920 Len=0 TSval=1697390876 TSecr=0
15	783.523981	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [ACK] Seq=1 Ack=58 Win=2920 Len=0
16	783.523997	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [FIN, ACK] Seq=1 Ack=58 Win=2920 Len=0
17	783.526128	192.168.1.12	192.168.1.11	TCP	66	21043 → 7 [ACK] Seq=58 Ack=2 Win=2920 Len=0 TSval=1697390876 TSecr=0

.... 0... = Acknowledgment: Not set	0000 00 50 79 66 68 00 00 50 79
.... 0... = Push: Not set	0010 00 3c 21 1c 00 00 40 06 d6
.... 0... = Reset: Not set	0020 01 0b 52 33 00 07 74 c2 22
> 1... = Syn: Set	0030 0b 68 4c 08 00 00 02 04 05
.... 0... = Fin: Not set	0040 21 1c 00 00 00 00 01 03 03
[TCP Flags:S.]	
Window: 2920	

Ready to load or capture Packets: 17 - Displayed: 17 (100.0%) Profile: Classic

Рисунок № 15 TCP-трафик

Эхо-запрос в режиме TCP на первый узел

На этот раз вы можете наблюдать рукопожатие во время эхо-запроса в режиме TCP. В первой строке на рисунке вторая конечная точка (192.168.1.12) отправляет синхросигнал первой (192.168.1.11). Затем ПК-1 отправляет ответ согласия с флагами SYN и ACK. В третьей строке ПК-2 отвечает на ответ, подтверждает соединение, устанавливает сессию и выставляет флаг ACK. Четвертая строка отправляет через порт ECHO пакет — запрос. Далее ПК-1 сообщает, что пакет получен — флаг ACK. После этого ПК-2 отправляет ПК-1 запрос на завершение сеанса с флагами FIN, PSH и ACK. ПК-1 соглашается — флаг ACK, а затем отправляет согласие на завершение сеанса для ПК-1 — флаги FIN и ACK. В конечном итоге ПК-2 подтверждает, что сессия закрыта и она завершена — флаг ACK.

9	783.521334	192.168.1.12	192.168.1.11	TCP	74	21043 → 7 [SYN, Seq=0 Win=2920 Len=0 MSS=1460 TSval=1697390876 TSecr=0
10	783.521498	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [SYN, ACK] Seq=0 Ack=1 Win=2920 Len=0
11	783.522413	192.168.1.12	192.168.1.11	TCP	66	21043 → 7 [ACK] Seq=1 Ack=1 Win=2920 Len=0 TSval=1697390876 TSecr=0
12	783.522613	192.168.1.12	192.168.1.11	ECHO	122	Request
13	783.522726	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [ACK] Seq=1 Ack=57 Win=2920 Len=0
14	783.523875	192.168.1.12	192.168.1.11	TCP	66	21043 → 7 [FIN, PSH, ACK] Seq=57 Ack=1 Win=2920 Len=0 TSval=1697390876 TSecr=0
15	783.523981	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [ACK] Seq=1 Ack=58 Win=2920 Len=0
16	783.523997	192.168.1.11	192.168.1.12	TCP	54	7 → 21043 [FIN, ACK] Seq=1 Ack=58 Win=2920 Len=0
17	783.526128	192.168.1.12	192.168.1.11	TCP	66	21043 → 7 [ACK] Seq=58 Ack=2 Win=2920 Len=0 TSval=1697390876 TSecr=0

.... 1 = Acknowledgment: Set	0000 00 50 79 66 68 01 00 50 79
.... 0 = Push: Not set	0010 00 28 21 1f 00 00 40 06 d6
.... 0 = Reset: Not set	0020 01 0c 00 07 52 33 2d 13 7c
.... 0 = Syn: Not set	0030 0b 68 4a 7f 00 00
> 1 = Fin: Set	
> [TCP Flags:A...F]	
Window: 2920	

Ready to load or capture Packets: 17 · Displayed: 17 (100.0%) Profile: Classic

Рисунок № 16 TCP-трафик

3. Моделирование простейшей сети на базе маршрутизатора FRR в GNS3

3.4 Включил захват трафика на соединении между коммутатором и маршрутизатором.

3.53.3.

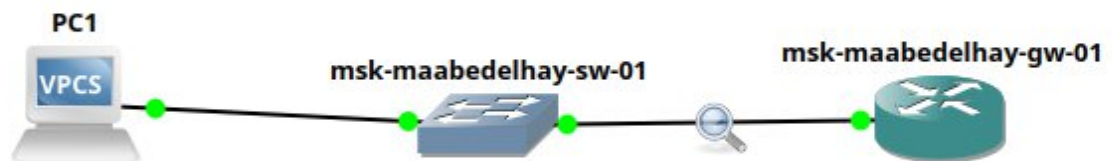


Рисунок № 17 Использование маршрутизатора FRR

3.6 Настроил IP-адресацию для интерфейса узла PC1.

```

PC1> ip 192.168.1.10/24 192.168.1.1
Checking for duplicate address...
PC1 : 192.168.1.10 255.255.255.0 gateway 192.168.1.1

PC1> save
Saving startup configuration to startup.vpc
. done

PC1> show ip

NAME       : PC1[1]
IP/MASK    : 192.168.1.10/24
GATEWAY    : 192.168.1.1
DNS        :
MAC        : 00:50:79:66:68:00
LPORT     : 20004
RHOST:PORT : 127.0.0.1:20005
MTU        : 1500

```

Рисунок № 18 Настройка IP-адресации для PC1

3.7 Настроим IP-адресацию для интерфейса локальной сети маршрутизатора

```

frr# configure terminal
frr(config)# hostname msk-maabeldelhay-gw-01
msk-maabeldelhay-gw-01(config)# exit
msk-maabeldelhay-gw-01# write memory
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]

msk-maabeldelhay-gw-01# configure terminal
msk-maabeldelhay-gw-01(config)# interface eth0
msk-maabeldelhay-gw-01(config-if)# ip address 192.168.1.1/24
msk-maabeldelhay-gw-01(config-if)# no shutdown
msk-maabeldelhay-gw-01(config-if)# exit
msk-maabeldelhay-gw-01(config)# exit
msk-maabeldelhay-gw-01# write memory
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]
msk-maabeldelhay-gw-01#

```

Рисунок № 19 Настройка маршрутизатора FRR

3.8 Проверим конфигурацию маршрутизатора и настройки IP-адресации

```
msk-maabedelhay-gw-01# show running-config
Building configuration...

Current configuration:
!
frr version 8.2.2
frr defaults traditional
hostname frr
hostname msk-maabedelhay-gw-01
service integrated-vtysh-config
!
interface eth0
 ip address 192.168.1.1/24
exit
!
end
```

Рисунок № 20 Настройка маршрутизатора FRR

```
msk-maabedelhay-gw-01# show interface brief
Interface      Status  VRF      Addresses
-----
eth0           up      default  192.168.1.1/24
eth1           down    default
eth2           down    default
eth3           down    default
eth4           down    default
eth5           down    default
eth6           down    default
eth7           down    default
lo             up      default
pimreg         up      default

msk-maabedelhay-gw-01#
```

Рисунок № 21 Настройка маршрутизатора FRR

3.9 Как вы можете видеть на терминале PC-1, в качестве шлюза для ПК установлен IP-адрес маршрутизатора FRR. и мне удалось отправить запрос ECHO на этот IP.

```

PC1> show ip

NAME       : PC1[1]
IP/MASK    : 192.168.1.10/24
GATEWAY    : 192.168.1.1
DNS        :
MAC        : 00:50:79:66:68:00
LPORT     : 20004
RHOST:PORT : 127.0.0.1:20005
MTU        : 1500

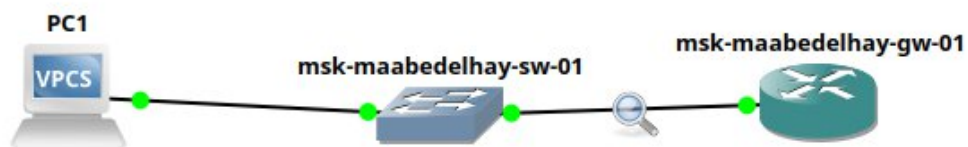
PC1> ping 192.168.1.1

84 bytes from 192.168.1.1 icmp_seq=1 ttl=64 time=12.369 ms
84 bytes from 192.168.1.1 icmp_seq=2 ttl=64 time=2.001 ms
84 bytes from 192.168.1.1 icmp_seq=3 ttl=64 time=1.981 ms
^C
PC1>

```

Рисунок № 22 проверка связи с маршрутизатором FRR

4. Моделирование простейшей сети на базе маршрутизатора VyOS в GNS3



4.7 Перешл в режим конфигурирования:

```

vyos@vyos$ configure
vyos@vyos#

```

Изменил имя устройства:

```

vyos@vyos#set system host-name msk-maabedelhay-gw-01

```

Задал IP-адрес на интерфейсе eth0:

```

vyos@vyos#set interfaces ethernet eth0 address 192.168.1.1/24

```

```
vyos@vyos:~$ configure
[edit]
vyos@vyos# set system host-name msk-maabedelhay-gw-01
[edit]
vyos@vyos# set interfaces ethernet eth0 address 192.168.1.1/24

Configuration path: [intefaces] is not valid
Set failed

[edit]
vyos@vyos# set interfaces ethernet eth0 address 192.168.1.1/24
[edit]
vyos@vyos# |
```

Рисунок № 23 Настройка маршрутизатора VYOS

Посмотрел изменения, внесенные в конфигурацию:

```
vyos@vyos# compare
```

Применил изменения в конфигурации и сохранил саму конфигурацию:

```
vyos@vyos# commit
```

```
vyos@vyos# save
```

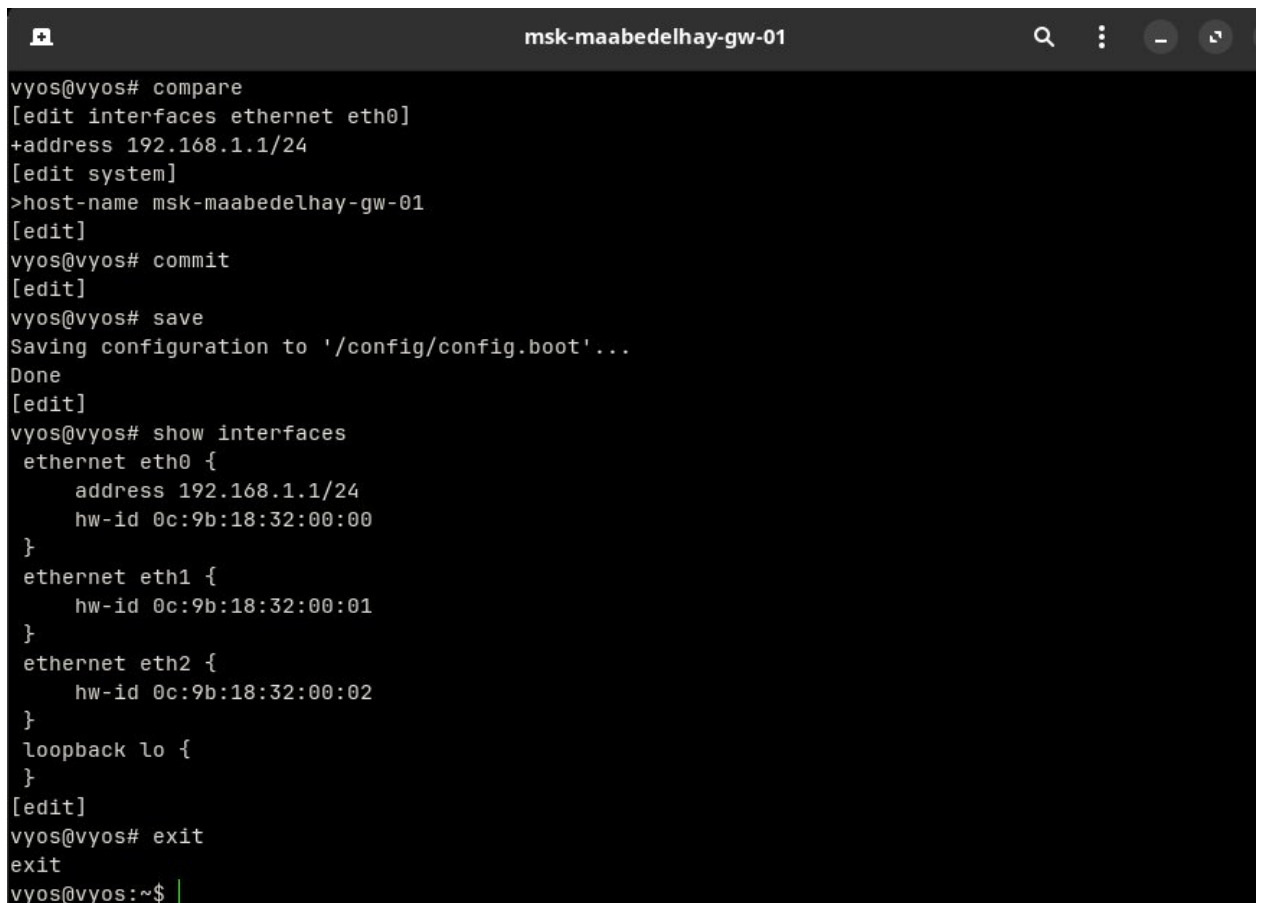
Просмотрел информацию об интерфейсах роутера:

```
vyos@vyos# show interfaces
```

Выход из режима настройки:

```
vyos@vyos# exit
```

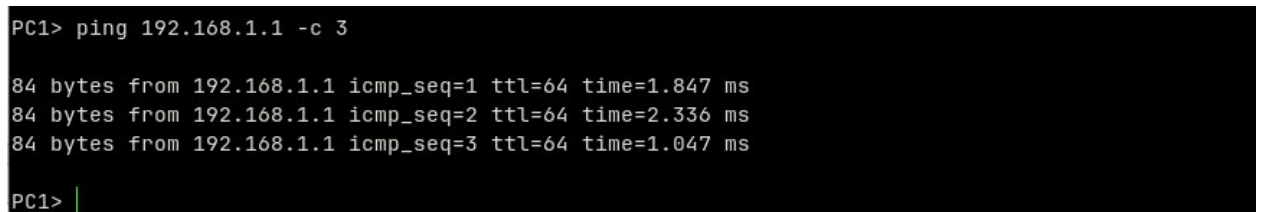
```
vyos@vyos$
```



```
vyos@vyos# compare
[edit interfaces ethernet eth0]
+address 192.168.1.1/24
[edit system]
>host-name msk-maabelhay-gw-01
[edit]
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
vyos@vyos# show interfaces
  ethernet eth0 {
    address 192.168.1.1/24
    hw-id 0c:9b:18:32:00:00
  }
  ethernet eth1 {
    hw-id 0c:9b:18:32:00:01
  }
  ethernet eth2 {
    hw-id 0c:9b:18:32:00:02
  }
  loopback lo {
  }
[edit]
vyos@vyos# exit
exit
vyos@vyos:~$
```

Рисунок № 24 Настройка маршрутизатора VYOS

4.8 Проверил соединение. PC-1 смог успешно выполнить проверку связи с адресом маршрутизатора 192.168.1.1.



```
PC1> ping 192.168.1.1 -c 3

84 bytes from 192.168.1.1 icmp_seq=1 ttl=64 time=1.847 ms
84 bytes from 192.168.1.1 icmp_seq=2 ttl=64 time=2.336 ms
84 bytes from 192.168.1.1 icmp_seq=3 ttl=64 time=1.047 ms

PC1>
```

Рисунок № 25 пинг маршрутизатора VYOS

Вывод

Построил простейшие модели сети на базе коммутатора и маршрутизаторов FRR и VyOS в GNS3, проанализировал трафик посредством Wireshark.