

Отчет по лабораторной работе №5

Дисциплина: Сетевые технологии

Иванов Сергей Владимирович

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	Моделирование простейшей сети на базе коммутатора в GNS3 . .	5
2.2	Анализ трафика в GNS3 посредством Wireshark	8
2.3	Моделирование простейшей сети на базе маршрутизатора FRR в GNS3	12
2.4	Моделирование простейшей сети на базе маршрутизатора VyOS в GNS3	16
3	Выводы	21

Список иллюстраций

2.1	Запуск GNS3	5
2.2	Создание проекта	6
2.3	Размещение компонентов	6
2.4	Просмотр возможных команд	7
2.5	Задание ip-адреса	7
2.6	Задание ip-адреса	7
2.7	Проверка соединения	8
2.8	Запуск анализатора трафика	8
2.9	Протокол ARP	9
2.10	Просмотр опций команды ping	9
2.11	Эхо запрос в ICMP моде	9
2.12	Эхо запрос в ICMP моде	10
2.13	Эхо запрос в UDP моде	10
2.14	Эхо запрос в UDP моде	11
2.15	Эхо запрос в TCP моде	11
2.16	Эхо запрос в TCP моде	12
2.17	Построение сети	12
2.18	Включение захвата трафика	13
2.19	Настройка IP адресации PC1	13
2.20	Настройка IP адресации маршрутизатора	14
2.21	Проверка роутера FRR	14
2.22	Проверка подключения	15
2.23	Анализ ICMP запросов	15
2.24	Построение сети с VyOS	16
2.25	Включение захвата трафика	16
2.26	Настройка IP адресации PC1	17
2.27	Загрузка VyOS	17
2.28	Установка системы VyOS	17
2.29	Изменение имени устройства	18
2.30	Задаем IP адрес	18
2.31	Сохранение изменений	18
2.32	Просмотр информации об интерфейсах	19
2.33	Проверка подключения	19
2.34	Анализ эхо запросов	20

1 Цель работы

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

2 Выполнение лабораторной работы

2.1 Моделирование простейшей сети на базе коммутатора в GNS3

Запустим GNS3 VM и GNS3. (рис. 1)

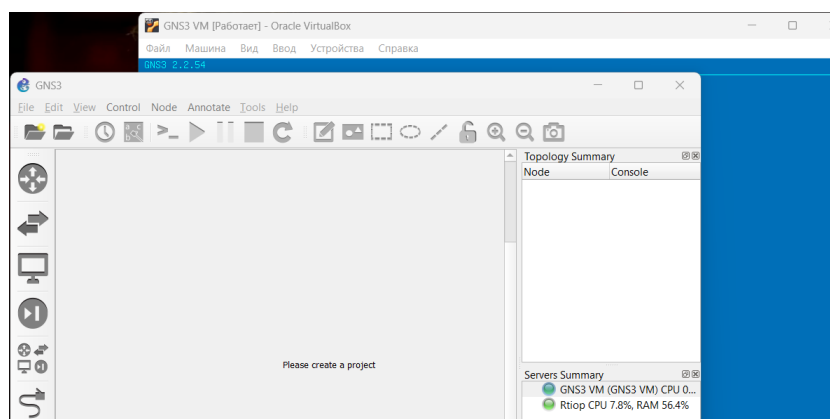


Рис. 2.1: Запуск GNS3

Создадим новый проект. (рис. 2)

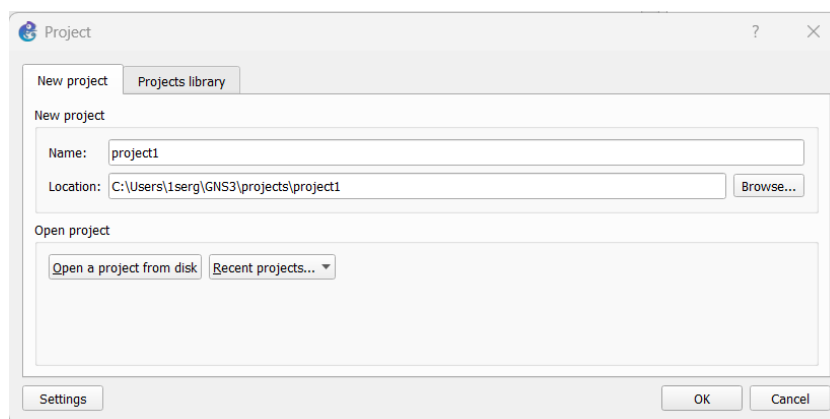


Рис. 2.2: Создание проекта

В рабочей области GNS3 разместим коммутатор Ethernet и два VPCS. Щёлкнув на устройстве правой кнопкой мыши выберем в меню Configure. Изменим название устройства, включив в имя устройства имя учётной записи. Соединим VPCS с коммутатором. Отобразим обозначение интерфейсов соединения (рис. 3)

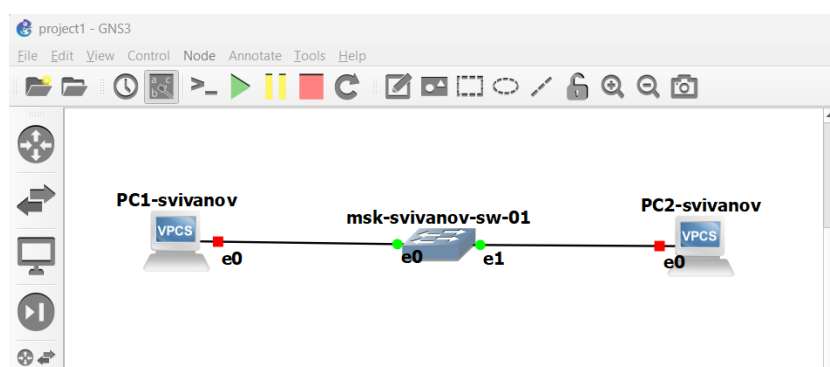
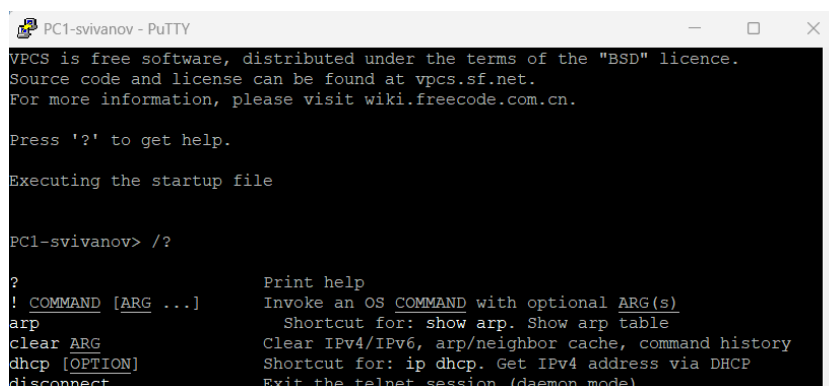


Рис. 2.3: Размещение компонентов

Зададим IP-адреса VPCS. Для этого с помощью меню, вызываемого правой кнопкой мыши, запустим Start, например, PC-1, затем вызовем его терминал Console . Для просмотра синтаксиса возможных для ввода команд наберем /?. (рис. 4)



```
PC1-svivanov - PuTTY
VPCS is free software, distributed under the terms of the "BSD" licence.
Source code and license can be found at vpcs.sf.net.
For more information, please visit wiki.freecode.com.cn.

Press '?' to get help.

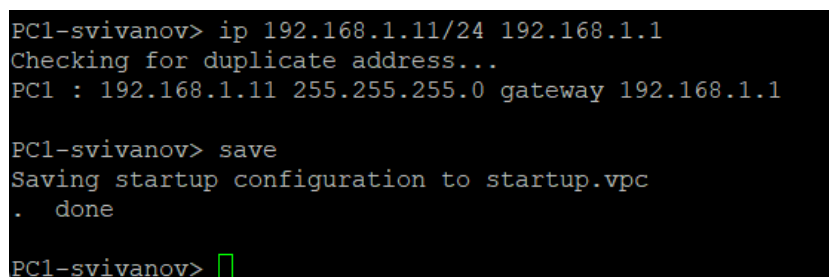
Executing the startup file

PC1-svivanov> /?

? Print help
! COMMAND [ARG ...] Invoke an OS COMMAND with optional ARG(s)
arp Shortcut for: show arp. Show arp table
clear ARG Clear IPv4/IPv6, arp/neighbor cache, command history
dhcp [OPTION] Shortcut for: ip dhcp. Get IPv4 address via DHCP
disconnect Exit the telnet session. (daemon mode)
```

Рис. 2.4: Просмотр возможных команд

Для задания IP-адреса 192.168.1.11 в сети 192.168.1.0/24 введем: ip 192.168.1.11/24 192.168.1.1 (рис. 5)



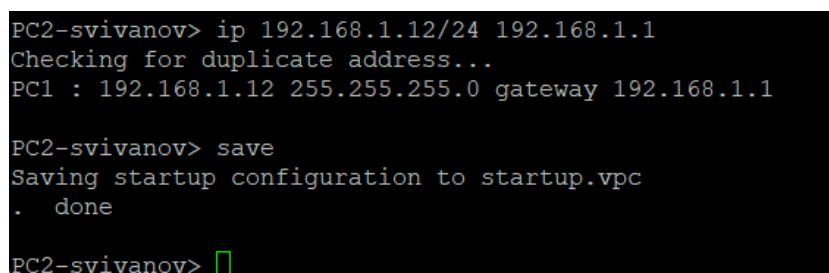
```
PC1-svivanov> ip 192.168.1.11/24 192.168.1.1
Checking for duplicate address...
PC1 : 192.168.1.11 255.255.255.0 gateway 192.168.1.1

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

PC1-svivanov> █
```

Рис. 2.5: Задание ip-адреса

Аналогичным образом зададим IP-адрес 192.168.1.12 для PC-2. (рис. 6)



```
PC2-svivanov> ip 192.168.1.12/24 192.168.1.1
Checking for duplicate address...
PC1 : 192.168.1.12 255.255.255.0 gateway 192.168.1.1

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

PC2-svivanov> █
```

Рис. 2.6: Задание ip-адреса

Проверим работоспособность соединения между PC-1 и PC-2 с помощью команды ping. Видим, что соединение работоспособно. (рис. 7)

```

PC1-svivanov> ping 192.168.1.12
84 bytes from 192.168.1.12 icmp_seq=1 ttl=64 time=0.267 ms
84 bytes from 192.168.1.12 icmp_seq=2 ttl=64 time=0.277 ms
84 bytes from 192.168.1.12 icmp_seq=3 ttl=64 time=0.409 ms
84 bytes from 192.168.1.12 icmp_seq=4 ttl=64 time=0.302 ms
84 bytes from 192.168.1.12 icmp_seq=5 ttl=64 time=0.501 ms
PC1-svivanov>

```

Рис. 2.7: Проверка соединения

2.2 Анализ трафика в GNS3 посредством Wireshark

Запустим на соединении между PC-1 и коммутатором анализатор трафика. Для этого щёлкнем правой кнопкой мыши на соединении, выберем в меню Start capture. Запустится Wireshark, а в проекте GNS3 на соединении появится значок лупы. (рис. 8)

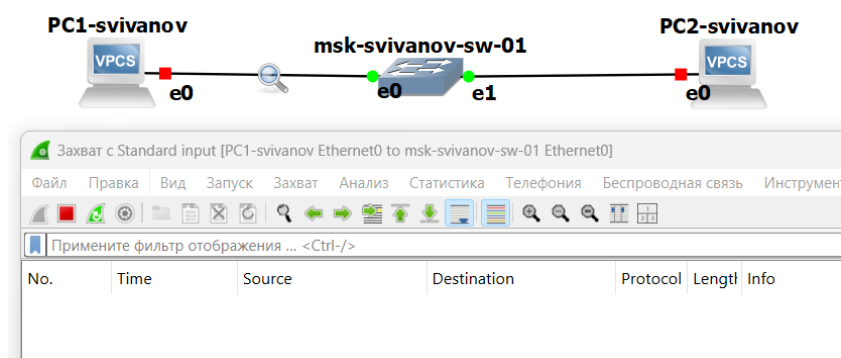


Рис. 2.8: Запуск анализатора трафика

В проекте GNS3 стартуем все узлы. В окне Wireshark отобразится информация по протоколу ARP. Проанализируем полученную информацию.

Отправляются Gratuitous запросы от PC1 К PC2. Это широковещательные сообщения для проверки уникальности своего IP адреса в сети. Также отправляются запросы от PC2 к PC1. (рис. 9)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	::	ff02::2	ICMPv6	62	Router Solicitation
2	0.020520	::	ff02::2	ICMPv6	62	Router Solicitation
3	0.050760	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
4	0.071479	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
5	1.052607	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
6	1.072931	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)
7	2.053091	Private_66:68:01	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.12 (Request)
8	2.073677	Private_66:68:00	Broadcast	ARP	64	Gratuitous ARP for 192.168.1.11 (Request)

> Frame 3: Packet, 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface 0
 > Ethernet II, Src: Private_66:68:01 (00:50:79:66:68:01), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
 > Source: Private_66:68:01 (00:50:79:66:68:01)
 Type: ARP (0x0806)
 [Stream index: 2]
 Padding: 00000000000000000000000000000000
 Frame check sequence: 0x00000000 [unverified]
 [FCS Status: Unverified]
 > 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

Рис. 2.9: Протокол ARP

В терминале PC-2 посмотрим информацию по опциям команды ping, введя ping /?. (рис. 10)

```

PC2-svivanov - PuTTY
PC2-svivanov> ping /?

ping HOST [OPTION ...]
  Ping the network HOST. HOST can be an ip address or name
Options:
  -1                ICMP mode, default
  -2                UDP mode
  -3                TCP mode
  -c count          Packet count, default 5
  -D                Set the Don't Fragment bit
  -f FLAG           Tcp header FLAG |C|E|U|A|P|R|S|F|
                   bits |7 6 5 4 3 2 1 0|
  -i ms             Wait ms milliseconds between sending each packet
  -l size           Data size
  -P protocol       Use IP protocol in ping packets
                   1 - ICMP (default), 17 - UDP, 6 - TCP
  -p port           Destination port
  -s port           Source port
  -T ttl            Set ttl, default 64
  -t               Send packets until interrupted by Ctrl+C
  -w ms            Wait ms milliseconds to receive the response
  
```

Рис. 2.10: Просмотр опций команды ping

Затем сделаем один эхо-запрос в ICMP-моду к узлу PC-1. (рис. 11)

```

PC2-svivanov> ping 192.168.1.11 1 -c 1
84 bytes from 192.168.1.11 icmp_seq=1 ttl=64 time=0.843 ms

PC2-svivanov> █
  
```

Рис. 2.11: Эхо запрос в ICMP моде

Отправляется ICMP запрос (request) от PC2 к PC1. PC1 отвечает (reply) с теми же id и seq. ICMP используется для проверки доступности узлов. (рис. 12)

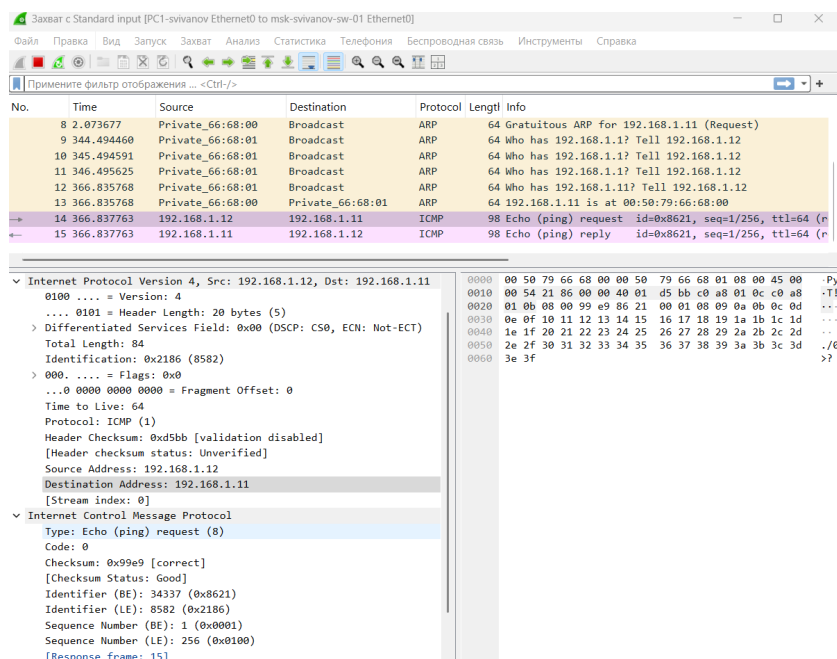


Рис. 2.12: Эхо запрос в ICMP режиме

Сделаем один эхо-запрос в UDP-режиме к узлу PC-1. (рис. 13)

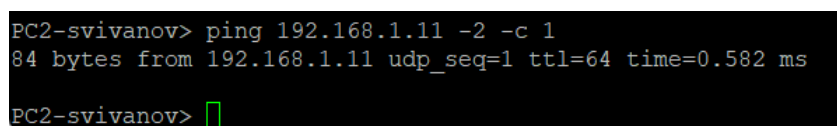


Рис. 2.13: Эхо запрос в UDP режиме

Отправляется UDP пакет от PC2 к PC1. Порт источника 14503. Порт назначения 7. Длина: 64 байта. Отправленные данные (полезная нагрузка): 56 байт. PC1 отправил ответный UDP пакет (Response). (рис. 14)

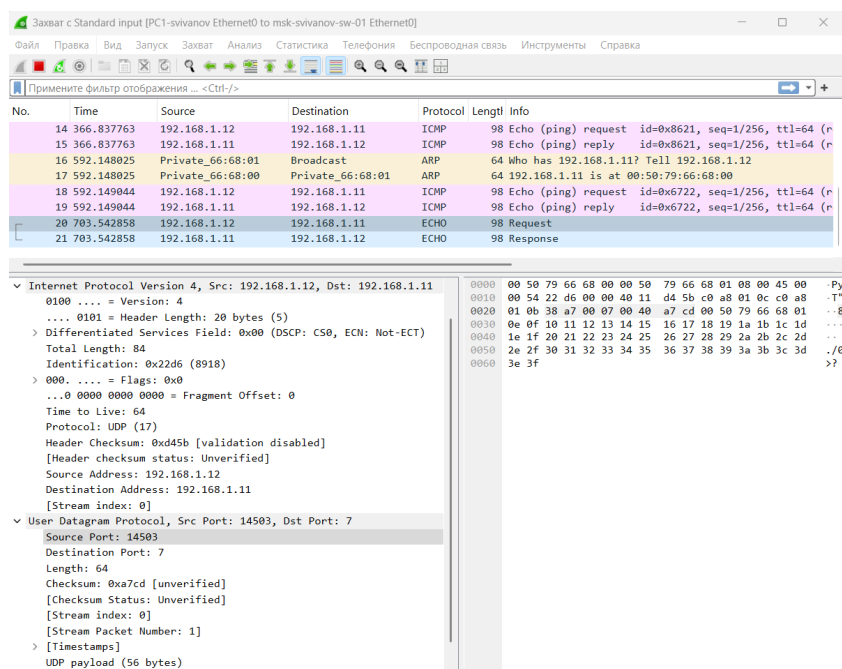


Рис. 2.14: Эхо запрос в UDP режиме

Сделаем один эхо-запрос в TCP-режиме к узлу PC-1. (рис. 15)

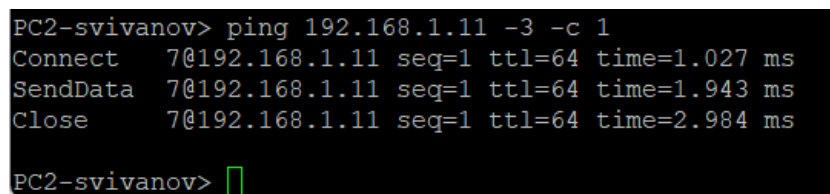


Рис. 2.15: Эхо запрос в TCP режиме

Устанавливается TCP соединение:

[SYN] от PC2 (порт 28458) к PC1 (порт 7)

[SYN, ACK] в ответ от PC1 к PC2

[ACK] завершает 3-х стороннее рукопожатие

Затем начинается передача данных, после чего аналогичным образом завершается соединение, добавляются флаги [FIN] которые говорят о завершении сеанса. (рис. 16)

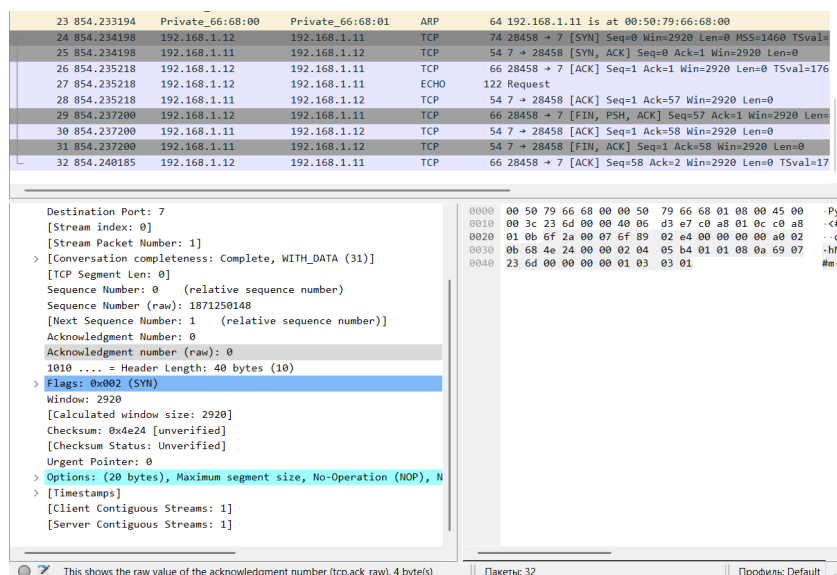


Рис. 2.16: Эхо запрос в TCP режиме

2.3 Моделирование простейшей сети на базе маршрутизатора FRR в GNS3

Создадим новый проект. В рабочей области GNS3 разместим VPCS, коммутатор Ethernet и маршрутизатор FRR. Изменим отображаемые названия устройств. (рис. 17)

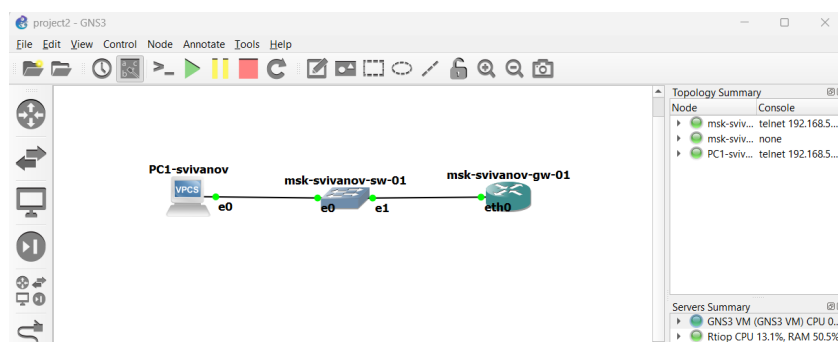


Рис. 2.17: Построение сети

Включим захват трафика на соединении между коммутатором и маршрутизатором. (рис. 18)

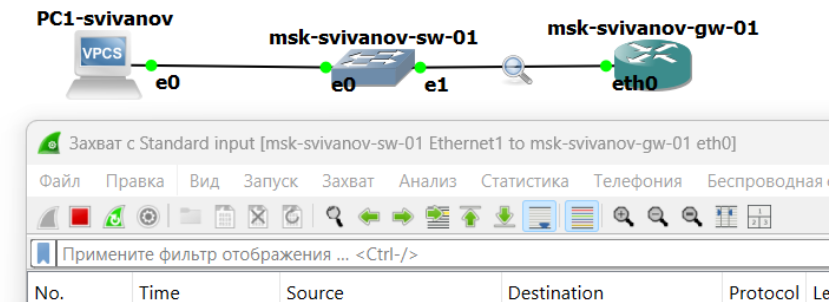


Рис. 2.18: Включение захвата трафика

Запустим все устройства проекта. Откроем консоль всех устройств проекта. Настроим IP-адресацию для интерфейса узла PC1:

```
ip 192.168.1.10/24 192.168.1.1
```

```
save
```

```
show ip (рис. 19)
```

```
PC1-svivanov> 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-svivanov> save
Saving startup configuration to startup.vpc
. done

PC1-svivanov> show ip

NAME       : PC1-svivanov[1]
IP/MASK    : 192.168.1.10/24
GATEWAY    : 192.168.1.1
DNS        :
MAC        : 00:50:79:66:68:00
```

Рис. 2.19: Настройка IP адресации PC1

Настроим IP-адресацию для интерфейса локальной сети маршрутизатора:
(рис. 20)

```

frr# configure terminal
frr(config)# hostname msk-user-gw-01
msk-user-gw-01(config)# exit
msk-user-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-user-gw-01# configure terminal
msk-user-gw-01(config)# interface eth0
msk-user-gw-01(config-if)# ip address 192.168.1.1/24
msk-user-gw-01(config-if)# no shutdown
msk-user-gw-01(config-if)# exit
msk-user-gw-01(config)# exit
msk-user-gw-01# memory
% Unknown command: memory
msk-user-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-user-gw-01# █

```

Рис. 2.20: Настройка IP адресации маршрутизатора

Проверим конфигурацию маршрутизатора и настройки IP-адресации:

msk-user-gw-01# show running-config

msk-user-gw-01# show interface brief (рис. 21)

```

msk-user-gw-01# show running-config
Building configuration...

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

```

Interface	Status	VRF	Addresses
eth0	up	default	192.168.1.1/24
eth1	down	default	
eth2	down	default	

Рис. 2.21: Проверка роутера FRR

Проверим подключение. Узел PC1 должен успешно отправлять эхо-запросы на адрес маршрутизатора 192.168.1.1. Все отправилось корректно. (рис. 22)

```

PC1-svivanov> ping 192.168.1.1
84 bytes from 192.168.1.1 icmp_seq=1 ttl=64 time=7.227 ms
84 bytes from 192.168.1.1 icmp_seq=2 ttl=64 time=2.750 ms
84 bytes from 192.168.1.1 icmp_seq=3 ttl=64 time=2.128 ms
84 bytes from 192.168.1.1 icmp_seq=4 ttl=64 time=1.714 ms
84 bytes from 192.168.1.1 icmp_seq=5 ttl=64 time=1.162 ms

PC1-svivanov> █

```

Рис. 2.22: Проверка подключения

В окне Wireshark проанализируем полученную информацию.

Видим, что отправилось 5 пар ICMP запросов. В каждой паре:

- 1) ICMP запрос (request) от PC1 к маршрутизатору.
- 2) ICMP ответ (reply) от маршрутизатора к PC1 с теми же id и seq. Это говорит о том, что подключение между PC1 и маршрутизатором успешное, PC1 корректно отправляет эхо-запросы, а маршрутизатор отвечает на них. (рис. 23)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.10	192.168.1.1	ICMP	98	Echo (ping) request id=0xd428, seq=1/256, ttl=64 (req)
2	0.008600	192.168.1.1	192.168.1.10	ICMP	98	Echo (ping) reply id=0xd428, seq=1/256, ttl=64 (rec)
3	1.010384	192.168.1.10	192.168.1.1	ICMP	98	Echo (ping) request id=0xd528, seq=2/512, ttl=64 (req)
4	1.012342	192.168.1.1	192.168.1.10	ICMP	98	Echo (ping) reply id=0xd528, seq=2/512, ttl=64 (rec)
5	2.013788	192.168.1.10	192.168.1.1	ICMP	98	Echo (ping) request id=0xd628, seq=3/768, ttl=64 (req)
6	2.015789	192.168.1.1	192.168.1.10	ICMP	98	Echo (ping) reply id=0xd628, seq=3/768, ttl=64 (rec)
7	3.019162	192.168.1.10	192.168.1.1	ICMP	98	Echo (ping) request id=0xd728, seq=4/1024, ttl=64 (req)
8	3.020163	192.168.1.1	192.168.1.10	ICMP	98	Echo (ping) reply id=0xd728, seq=4/1024, ttl=64 (rec)
9	4.022324	192.168.1.10	192.168.1.1	ICMP	98	Echo (ping) request id=0xd828, seq=5/1280, ttl=64 (req)
10	4.024340	192.168.1.1	192.168.1.10	ICMP	98	Echo (ping) reply id=0xd828, seq=5/1280, ttl=64 (rec)
11	5.039288	0c:c3:95:2a:00:00	Private_66:68:00	ARP	60	Who has 192.168.1.10? Tell 192.168.1.1
12	5.039288	Private_66:68:00	0c:c3:95:2a:00:00	ARP	60	192.168.1.10 is at 00:50:79:66:68:00

Frame 1: Packet, 98 bytes on wire (784 bits), 98 bytes captured (784) on interface eth0	
Ethernet II, Src: Private_66:68:00 (00:50:79:66:68:00), Dst: 0c:c3:95:2a:00:00	
Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.1	
Internet Control Message Protocol	
Type: Echo (ping) request (8)	0000 0c c3 95 2a 00 00 00 50 79 66 68 00 00 00 45 00 ...
Code: 0	0010 00 54 28 d4 00 00 40 01 ce 79 c0 a8 01 0a c0 a8 ...T(
Checksum: 0x4be2 [correct]	0020 01 01 08 00 4b e2 d4 28 00 01 08 09 0a 0b 0c 0d ...
[Checksum Status: Good]	0030 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d ...
Identifier (BE): 54312 (0xd428)	0040 1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d ...
Identifier (LE): 10452 (0x28d4)	0050 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d .../0
Sequence Number (BE): 1 (0x0001)	0060 3e 3f ...>
Sequence Number (LE): 256 (0x0100)	
[Response frame: 2]	
Data (56 bytes)	

Рис. 2.23: Анализ ICMP запросов

2.4 Моделирование простейшей сети на базе маршрутизатора VyOS в GNS3

Создадим новый проект. В рабочей области GNS3 разместим VPCS, коммутатор Ethernet и маршрутизатор VyOS. Изменим отображаемые названия устройств. (рис. 24)

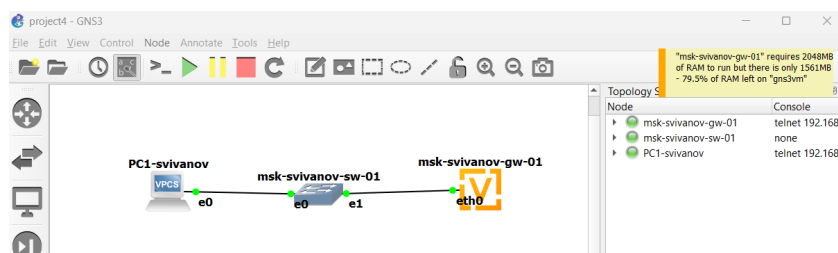


Рис. 2.24: Построение сети с VyOS

Включим захват трафика на соединении между коммутатором и маршрутизатором. (рис. 25)

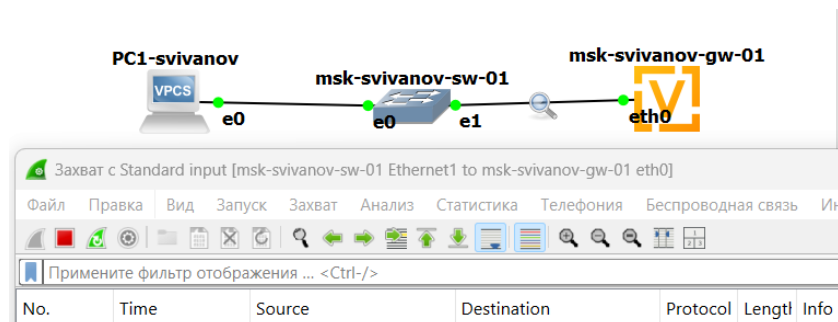


Рис. 2.25: Включение захвата трафика

Запустим все устройства проекта. Откроем консоль всех устройств проекта. Настроим IP-адресацию для интерфейса узла PC1:

```
ip 192.168.1.10/24 192.168.1.1
```

```
save
```

```
show ip (рис. 26)
```



```

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

PC1-svivanov> show ip

NAME       : PC1-svivanov[1]
IP/MASK    : 192.168.1.10/24
GATEWAY    : 192.168.1.1

```

Рис. 2.26: Настройка IP адресации PC1

Настройте маршрутизатор VyOS. После загрузки введем логин vyos и пароль vyos: (рис. 27)

```

vyos login: vyos
Password:
Welcome to VyOS!

Check out project news at https://blog.vyos.io
and feel free to report bugs at https://phabricator.vyos.net

You can change this banner using "set system login banner post-login" com

VyOS is a free software distribution that includes multiple components,
you can check individual component licenses under /usr/share/doc/*/copyri
vyos@vyos:~$ █

```

Рис. 2.27: Загрузка VyOS

Установим систему на диск: vyos@vyos:~\$ install image. В моем случае система уже установлена. (рис. 28)

```

vyos@vyos:~$ install image
You are trying to install from an already installed system. An ISO
image file to install or URL must be specified.
Exiting...
vyos@vyos:~$ █

```

Рис. 2.28: Установка системы VyOS

Перейдем в режим конфигурирования: vyos@vyos\$ configure

Изменим имя устройства: vyos@vyos#set system host-name msk-user-gw-01
(рис. 29)

```
vyos@vyos:~$ configure
[edit]
vyos@vyos# system host-name msk-user-gw-01

Invalid command: [system]

[edit]
vyos@vyos# set system host-name msk-user-gw-01
[edit]
vyos@vyos#
```

Рис. 2.29: Изменение имени устройства

Зададим IP-адрес на интерфейсе eth0: vyos@vyos# set interfaces ethernet eth0 address 192.168.1.1/24

Посмотрим внесённые в конфигурацию изменения: vyos@vyos# compare (рис. 30)

```
vyos@vyos# set interfaces ethernet eth0 address 192.168.1.1/24
[edit]
vyos@vyos# compare
[edit interfaces ethernet eth0]
+address 192.168.1.1/24
[edit system]
>host-name msk-user-gw-01
[edit]
```

Рис. 2.30: Задаем IP адрес

Применим изменения в конфигурации и сохраним саму конфигурацию:

vyos@vyos# commit

vyos@vyos# save (рис. 31)

```
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
vyos@vyos#
```

Рис. 2.31: Сохранение изменений

Посмотрим информацию об интерфейсах маршрутизатора: vyos@vyos# show interfaces (рис. 32)

```
vyos@vyos# show interfaces
  ethernet eth0 {
    address 192.168.1.1/24
    hw-id 0c:44:58:03:00:00
  }
  ethernet eth1 {
    hw-id 0c:44:58:03:00:01
  }
  ethernet eth2 {
    hw-id 0c:44:58:03:00:02
  }
```

Рис. 2.32: Просмотр информации об интерфейсах

Выйдем из режима конфигурирования. Проверим подключение. Узел PC1 должен успешно отправлять эхо-запросы на адрес маршрутизатора 192.168.1.1. (рис. 33)

```
PC1-svivanov> ping 192.168.1.1
84 bytes from 192.168.1.1 icmp_seq=1 ttl=64 time=1.256 ms
84 bytes from 192.168.1.1 icmp_seq=2 ttl=64 time=0.934 ms
84 bytes from 192.168.1.1 icmp_seq=3 ttl=64 time=1.326 ms
84 bytes from 192.168.1.1 icmp_seq=4 ttl=64 time=1.151 ms
84 bytes from 192.168.1.1 icmp_seq=5 ttl=64 time=1.381 ms
PC1-svivanov> █
```

Рис. 2.33: Проверка подключения

В окне Wireshark проанализируем полученную информацию. (рис. 34)

Сначала PC1 отправляет широковещательный ARP запрос чтобы узнать MAC адрес маршрутизатора. Маршрутизатор отвечает, и теперь PC1 знает его MAC адрес.

Далее видим, что отправилось 5 пар ICMP запросов. В каждой паре:

- 1) ICMP запрос (request) от PC1 к маршрутизатору.
- 2) ICMP ответ (reply) от маршрутизатора к PC1 с теми же id и seq. Это говорит о том, что подключение между PC1 и маршрутизатором успешное, PC1 корректно отправляет эхо-запросы, а маршрутизатор отвечает на них.

В конце аналогично маршрутизатор запрашивает MAC адрес PC1, и тот ему отвечает.

14	329.999063	Private_66:68:00	Broadcast	ARP	64	Who has 192.168.1.1? Tell 192.168.1.10
15	330.001558	0c:44:58:03:00:00	Private_66:68:00	ARP	42	192.168.1.1 is at 0c:44:58:03:00:00
→	16	330.004063	192.168.1.10	192.168.1.1	ICMP	98 Echo (ping) request id=0xa95f, seq=1/256, ttl=64 (r
←	17	330.005051	192.168.1.1	192.168.1.10	ICMP	98 Echo (ping) reply id=0xa95f, seq=1/256, ttl=64 (r
	18	331.006329	192.168.1.10	192.168.1.1	ICMP	98 Echo (ping) request id=0xaa5f, seq=2/512, ttl=64 (r
	19	331.006329	192.168.1.1	192.168.1.10	ICMP	98 Echo (ping) reply id=0xaa5f, seq=2/512, ttl=64 (r
	20	332.010155	192.168.1.10	192.168.1.1	ICMP	98 Echo (ping) request id=0xab5f, seq=3/768, ttl=64 (r
	21	332.011177	192.168.1.1	192.168.1.10	ICMP	98 Echo (ping) reply id=0xab5f, seq=3/768, ttl=64 (r
	22	333.012596	192.168.1.10	192.168.1.1	ICMP	98 Echo (ping) request id=0xac5f, seq=4/1024, ttl=64 (
	23	333.013603	192.168.1.1	192.168.1.10	ICMP	98 Echo (ping) reply id=0xac5f, seq=4/1024, ttl=64 (
	24	334.016262	192.168.1.10	192.168.1.1	ICMP	98 Echo (ping) request id=0xad5f, seq=5/1280, ttl=64 (
	25	334.017268	192.168.1.1	192.168.1.10	ICMP	98 Echo (ping) reply id=0xad5f, seq=5/1280, ttl=64 (
	26	335.180493	0c:44:58:03:00:00	Private_66:68:00	ARP	42 Who has 192.168.1.10? Tell 192.168.1.1
	27	335.180493	Private_66:68:00	0c:44:58:03:00:00	ARP	42 192.168.1.10 is at 00:50:79:66:68:00

> Frame 17: Packet, 98 bytes on wire (784 bits), 98 bytes captured (784) on interface 0

> Ethernet II, Src: 0c:44:58:03:00:00 (0c:44:58:03:00:00), Dst: Private_66:68:00:00:00:00 (0c:44:58:03:00:00:00)

> Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.10

> Internet Control Message Protocol

Type: Echo (ping) reply (0)

Code: 0

Checksum: 0x7eab [correct]

[Checksum Status: Good]

Identifier (BE): 43359 (0xa95f)

Identifier (LE): 24489 (0x5fa9)

Sequence Number (BE): 1 (0x0001)

Sequence Number (LE): 256 (0x0100)

[Request frame: 16]

[Response time: 0.988 ms]

> Data (56 bytes)

0000 00 50 79 66 68 00 0c 44 58 03 00 00 08 00 45 00 Py

0010 00 54 c3 75 00 00 40 01 33 d8 c0 a8 01 01 c0 a8 .T

0020 01 0a 00 00 7e ab a9 5f 00 01 08 09 0a 0b 0c 0d ...

0030 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d ...

0040 1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d ..

0050 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d ./e

0060 3e 3f >>

Рис. 2.34: Анализ эхо запросов

3 Выводы

В ходе выполнения лабораторной работы мы построили простейшие модели сети на базе коммутатора и маршрутизаторов FRR и VyOS в GNS3, и проанализировали трафик посредством Wireshark.