

# **Отчёт по лабораторной работе №5**

**Сетевые технологии**

Ищенко Ирина НПИбд-02-22

# **Содержание**

<b>1 Цель работы</b>	<b>5</b>
<b>2 Выполнение лабораторной работы</b>	<b>6</b>
<b>3 Выводы</b>	<b>25</b>

# Список иллюстраций

2.1 Топология простейшей сети в GNS3 . . . . .	6
2.2 Задание IP-адреса для PC-1 . . . . .	7
2.3 Задание IP-адреса для PC-2 . . . . .	7
2.4 Пингование PC-2 . . . . .	8
2.5 Остановка всех узлов . . . . .	8
2.6 Захват трафика, старт узлов . . . . .	8
2.7 Информация по протоколу ARP . . . . .	9
2.8 Эхо-запрос в ICMP-моде . . . . .	10
2.9 Полученная информация по эхо-запросу в ICMP-моде к узлу PC-1 . .	11
2.10 Эхо-ответ . . . . .	12
2.11 Эхо-запрос в UDP-моде . . . . .	12
2.12 Полученная информация по эхо-запросу в UDP-моде к узлу PC-1 . .	13
2.13 Эхо-ответ . . . . .	14
2.14 Эхо-запрос в TCP-моде . . . . .	15
2.15 Полученная информация по эхо-запросу в TCP-моде к узлу PC-1 . .	16
2.16 Полученная информация по эхо-запросу в TCP-моде к узлу PC-1 . .	17
2.17 Полученная информация по эхо-запросу в TCP-моде к узлу PC-1 . .	18
2.18 Топология сети с маршрутизатором FRR . . . . .	19
2.19 Терминал маршрутизатора FRR . . . . .	19
2.20 Топология сети с маршрутизатором VyOS . . . . .	20
2.21 Настройка IP-адресации для интерфейса узла PC-1 . . . . .	20
2.22 Логин, проверка установки системы на диск . . . . .	21
2.23 Режим конфигурирования . . . . .	21
2.24 Режим конфигурирования . . . . .	22
2.25 Режим конфигурирования . . . . .	22
2.26 Пингование маршрутизатора . . . . .	23
2.27 Полученная информация в Wireshark по ICMP-сообщению . . . . .	24

# **Список таблиц**

# **1 Цель работы**

Построить простейшие модели сетей на базе коммутатора и маршрутизаторов FRR и VyOS в GNS3, проанализировать трафик посредством Wireshark.

## 2 Выполнение лабораторной работы

Для начала запустим GNS3 VM и GNS3, а также создадим новый проект. В рабочей области GNS3 разместим коммутатор Ethernet и два VPCS. В меню Configure изменим название устройства, включив в имя устройства имя моей учётной записи. Коммутатору присвоим название msk-ioithenko-sw-01. Соединим VPCS с коммутатором и отобразим обозначение интерфейсов соединения (рис. fig. 2.1).

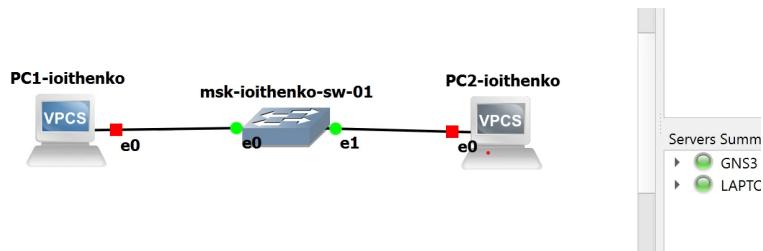


Рис. 2.1: Топология простейшей сети в GNS3

Зададим IP-адреса VPCS. Для этого с помощью меню, вызываемого правой кнопкой мыши, запустим Start PC-1, затем вызовем его терминал Console. Для просмотра синтаксиса возможных для ввода команд можно набрать ?. 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 Для сохранения конфигурации введем команду save (рис. fig. 2.2).

```

PC1-ioithenko - PuTTY
ip ARG ... [OPTION]      Configure the current VPC's IP settings. See ip ?
load [FILENAME]          Load the configuration/script from the file FILENAME
ping HOST [OPTION ...]   Ping HOST with ICMP (default) or TCP/UDP. See ping ?
quit                      Quit program
relay ARG ...             Configure packet relay between UDP ports. See relay ?
rlogin [ip] port          Telnet to port on host at ip (relative to host PC)
save [FILENAME]           Save the configuration to the file FILENAME
set ARG ...               Set VPC name and other options. Try set ?
show [ARG ...]            Print the information of VPCs (default). See show ?
sleep [seconds] [TEXT]    Print TEXT and pause running script for seconds
trace HOST [OPTION ...]  Print the path packets take to network HOST
version                  Shortcut for: show version

To get command syntax help, please enter '?' as an argument of the command.

VPCS> 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

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

VPCS>

```

Рис. 2.2: Задание IP-адреса для PC-1

Аналогичным образом зададим IP-адрес 192.168.1.12 для PC-2. Пингуем соответственно IP-адрес PC-1. Получаем эхо-ответ от PC-1. Значит соединение наших PC работоспособно (рис. fig. 2.3).

```

PC2-ioithenko - PuTTY
For more information, please visit wiki.freecode.com.cn.

Press '?' to get help.

Executing the startup file

Hostname is too long. (Maximum 12 characters)

VPCS> 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

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

VPCS> ping 192.168.1.11
84 bytes from 192.168.1.11 icmp_seq=1 ttl=64 time=2.194 ms
84 bytes from 192.168.1.11 icmp_seq=2 ttl=64 time=1.582 ms
84 bytes from 192.168.1.11 icmp_seq=3 ttl=64 time=1.753 ms
84 bytes from 192.168.1.11 icmp_seq=4 ttl=64 time=3.804 ms
84 bytes from 192.168.1.11 icmp_seq=5 ttl=64 time=1.647 ms

VPCS>

```

Рис. 2.3: Задание IP-адреса для PC-2

Проверим работоспособность соединения между PC-1 и PC-2 с помощью команды ping. В терминале PC-1 введем команду ping и IP-адрес, присвоенный PC-2. Получаем эхо-ответ от PC-2 (возвращены 5 пакетов) (рис. fig. 2.4).

```

PC1-ioithenko - PuTTY
set ARG ...           Set VPC name and other options. Try set ?
show [ARG ...]        Print the information of VPCs (default). See show ?
sleep [seconds] [TEXT] Print TEXT and pause running script for seconds
trace HOST [OPTION ...] Print the path packets take to network HOST
version               Shortcut for: show version

To get command syntax help, please enter '?' as an argument of the command.

VPCS> 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

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

VPCS> ping 192.168.1.12
84 bytes from 192.168.1.12 icmp_seq=1 ttl=64 time=2.098 ms
84 bytes from 192.168.1.12 icmp_seq=2 ttl=64 time=1.978 ms
84 bytes from 192.168.1.12 icmp_seq=3 ttl=64 time=1.923 ms
84 bytes from 192.168.1.12 icmp_seq=4 ttl=64 time=1.612 ms
84 bytes from 192.168.1.12 icmp_seq=5 ttl=64 time=2.767 ms

VPCS>

```

Рис. 2.4: Пингование PC-2

Остановим в проекте все узлы (рис. fig. 2.5).

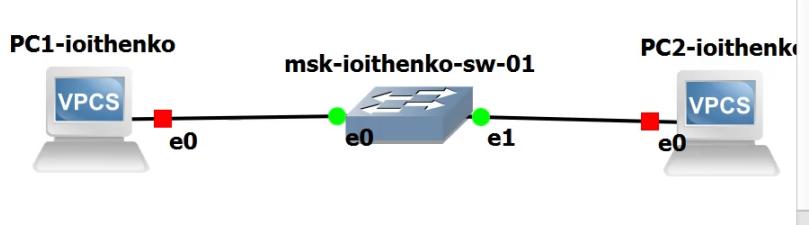


Рис. 2.5: Остановка всех узлов

Запустим на соединении между PC-1 и коммутатором анализатор трафика. Для этого щёлкнем правой кнопкой мыши на соединении, выберем в меню Start capture. После этого запустился Wireshark, а в проекте GNS3 на соединении появился значок лупы. В проекте GNS3 стартуем все узлы (рис. fig. 2.6).

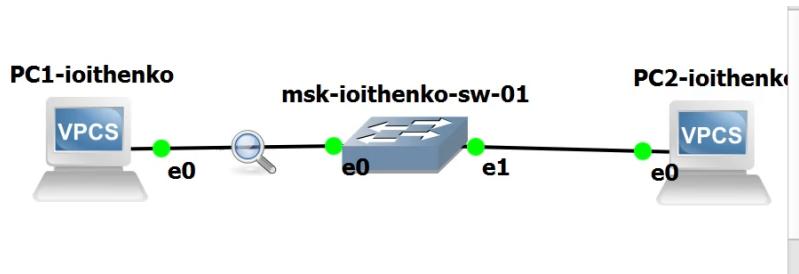


Рис. 2.6: Захват трафика, старт узлов

В окне Wireshark отобразится информация по протоколу ARP (рис. fig. 2.7). В

поле кадра физического уровня мы можем узнать длину кадра (в моем случае было 64 бита). В поле канального уровня можем посмотреть мас-адреса источника и получателя. По нулевому и первому битам можем определить тип мас-адресов (получатель – локально администрируемый и широковещательный; источник – глобально администрируемый и индивидуальный).

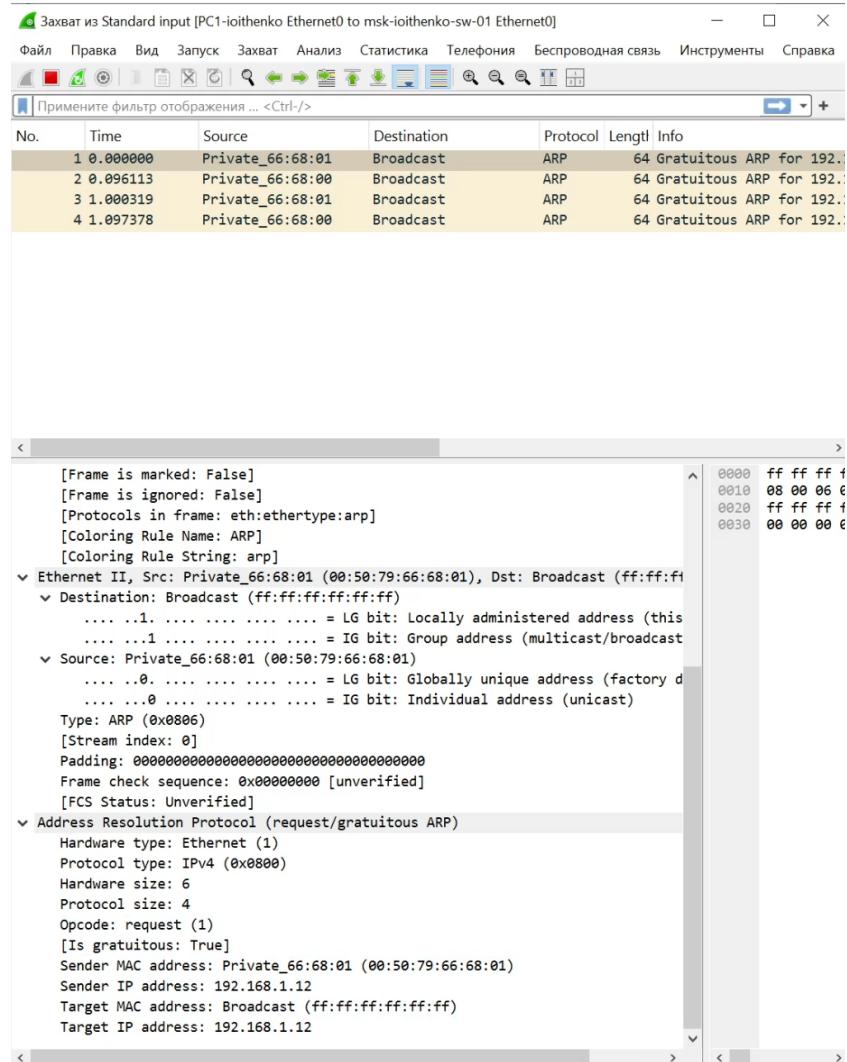


Рис. 2.7: Информация по протоколу ARP

В терминале PC-2 посмотрим информацию по опциям команды ping. Затем сделаем один эхо-запрос в ICMP-mode к узлу PC-1. Для этого используем опцию -1 (рис. fig. 2.8).

```
VPCS> ping 192.168.1.11 -l  
84 bytes from 192.168.1.11 icmp_seq=1 ttl=64 time=1.833 ms  
84 bytes from 192.168.1.11 icmp_seq=2 ttl=64 time=2.260 ms  
84 bytes from 192.168.1.11 icmp_seq=3 ttl=64 time=1.811 ms  
84 bytes from 192.168.1.11 icmp_seq=4 ttl=64 time=1.695 ms  
84 bytes from 192.168.1.11 icmp_seq=5 ttl=64 time=1.423 ms
```

Рис. 2.8: Эхо-запрос в ICMP-моде

Далее откроем Wireshark и проанализируем эхо-запрос по протоколу ICMP (рис. fig. 2.9) и (рис. fig. 2.10). В поле канального уровня можем посмотреть мас-адреса источника и получателя. По нулевому и первому битам можем определить тип мас-адресов (получатель и источник - глобально администрируемые и одиночные, так как биты равны 0). В поле сетевого уровня указан протокол ICMP и IP-адреса источника (192.168.1.12, то есть PC-2) и получателя (192.168.1.11, то есть PC-2).

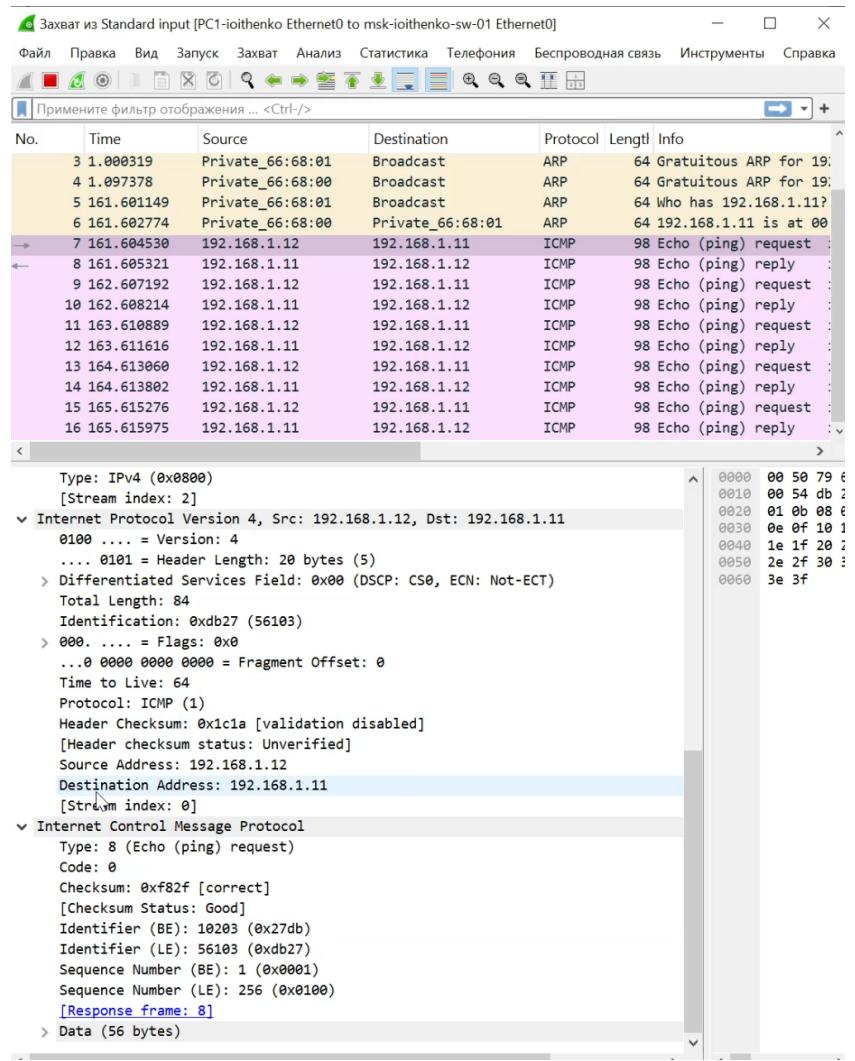


Рис. 2.9: Полученная информация по эхо-запросу в ICMP-моде к узлу PC-1

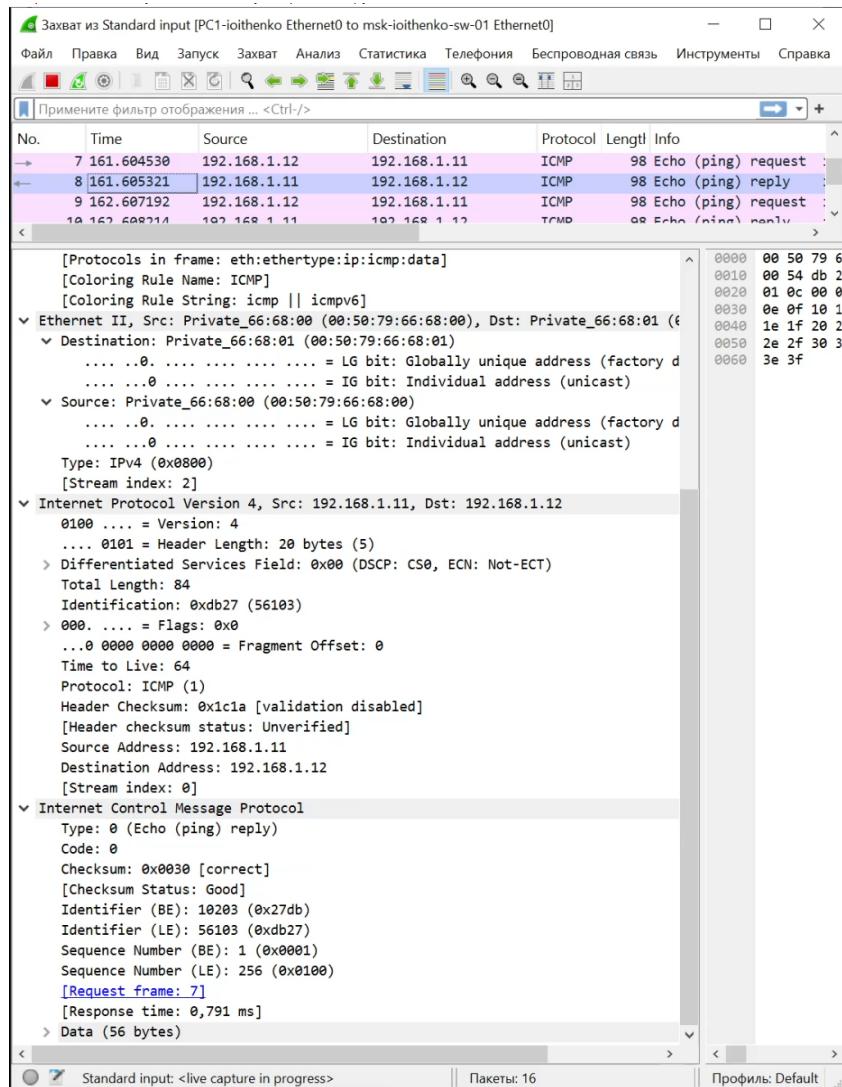


Рис. 2.10: Эхо-ответ

Сделаем один эхо-запрос в UDP-моде к узлу РС-1. Для этого используем опцию -2 (рис. fig. 2.11).

```
VPCS> ping 192.168.1.11 -2
84 bytes from 192.168.1.11 udp_seq=1 ttl=64 time=2.028 ms
84 bytes from 192.168.1.11 udp_seq=2 ttl=64 time=1.957 ms
84 bytes from 192.168.1.11 udp_seq=3 ttl=64 time=2.253 ms
84 bytes from 192.168.1.11 udp_seq=4 ttl=64 time=1.948 ms
84 bytes from 192.168.1.11 udp_seq=5 ttl=64 time=2.669 ms
```

Рис. 2.11: Эхо-запрос в UDP-моде

Далее откроем Wireshark и проанализируем эхо-запрос по протоколу UDP (рис.

fig. 2.12) и (рис. fig. 2.13). В поле канального уровня можем посмотреть мас-адреса источника и получателя. По нулевому и первому битам можем определить тип мас-адресов (получатель и источник - глобально администрируемые и одиночные, так как биты равны 0). В поле сетевого уровня указан протокол UDP и IP-адреса источника (192.168.1.12, то есть PC-2) и получателя (192.168.1.11, то есть PC-2). В поле протокола UDP указаны порты источника (17622) и получателя (7).

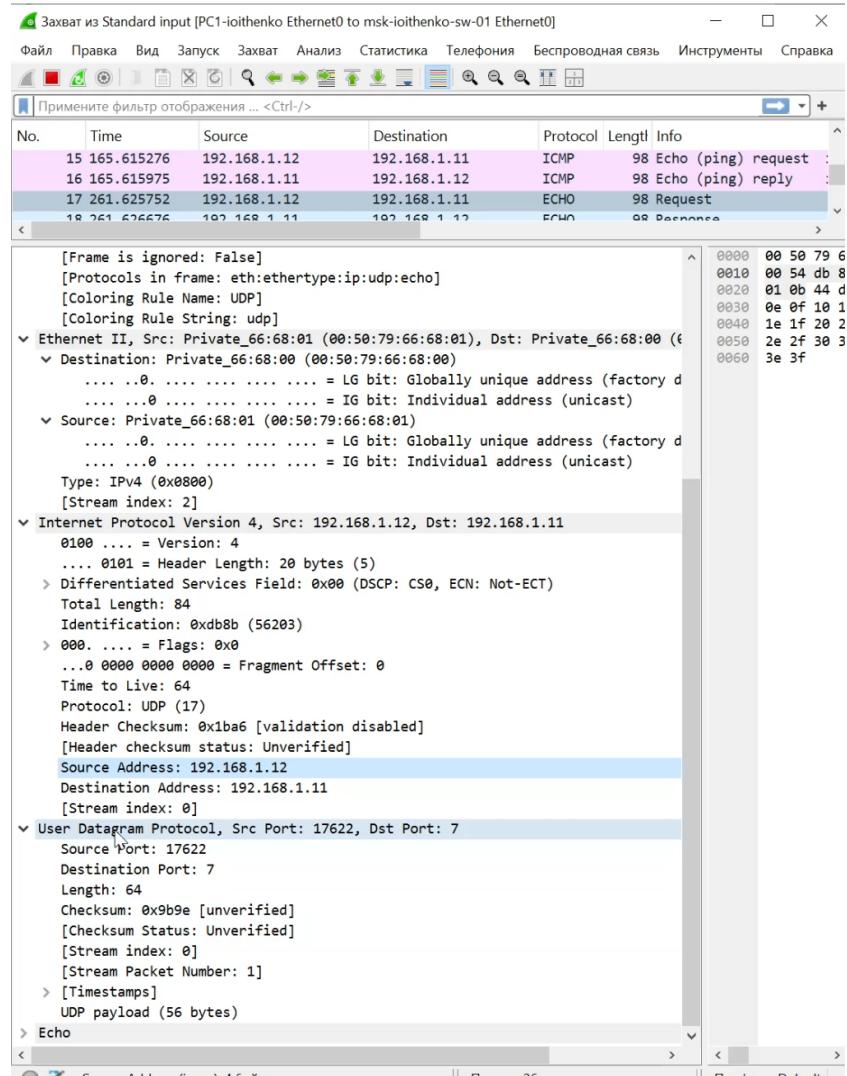


Рис. 2.12: Полученная информация по эхо-запросу в UDP-моде к узлу PC-1

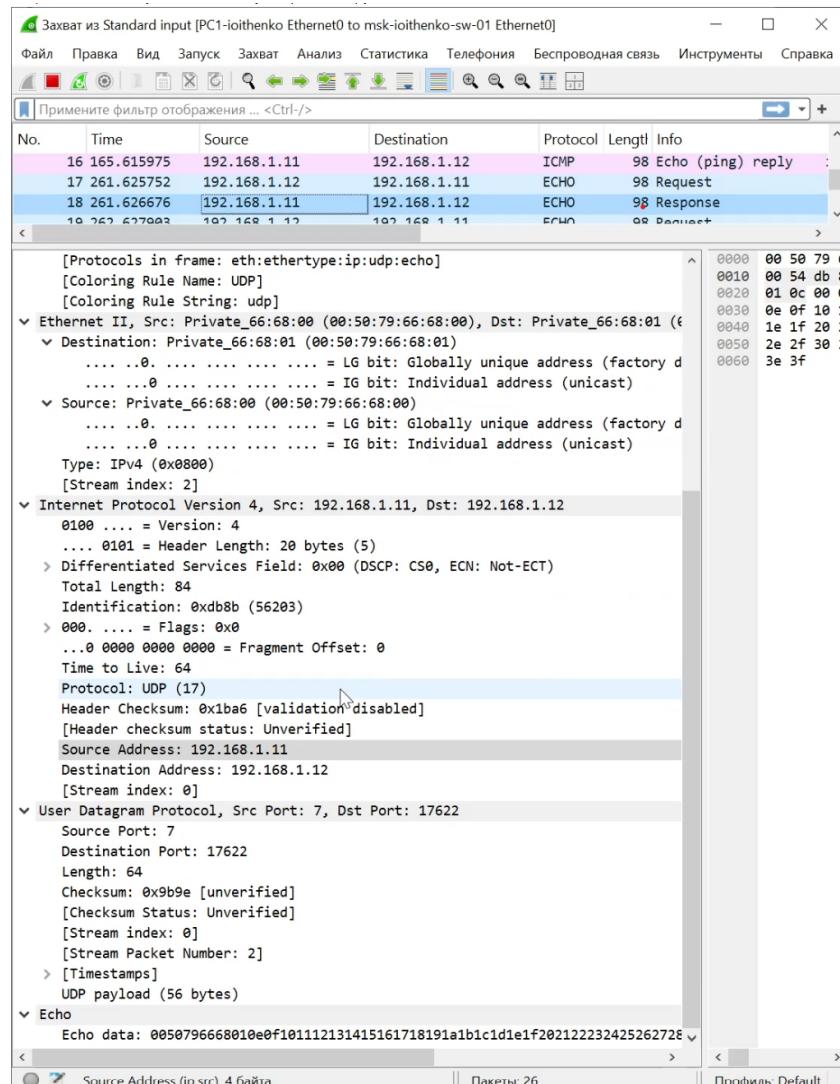


Рис. 2.13: Эхо-ответ

Сделаем один эхо-запрос в TCP-моде к узлу PC-1. Для этого используем опцию -3 (рис. fig. 2.14).

```
VPCS> ping 192.168.1.11 -3
Connect 7@192.168.1.11 seq=1 ttl=64 time=5.556 ms
SendData 7@192.168.1.11 seq=1 ttl=64 time=2.111 ms
Close 7@192.168.1.11 seq=1 ttl=64 time=3.191 ms
Connect 7@192.168.1.11 seq=2 ttl=64 time=4.305 ms
SendData 7@192.168.1.11 seq=2 ttl=64 time=2.131 ms
Close 7@192.168.1.11 seq=2 ttl=64 time=4.309 ms
Connect 7@192.168.1.11 seq=3 ttl=64 time=3.283 ms
SendData 7@192.168.1.11 seq=3 ttl=64 time=3.378 ms
Close 7@192.168.1.11 seq=3 ttl=64 time=4.270 ms
Connect 7@192.168.1.11 seq=4 ttl=64 time=2.145 ms
SendData 7@192.168.1.11 seq=4 ttl=64 time=2.472 ms
Close 7@192.168.1.11 seq=4 ttl=64 time=5.371 ms
Connect 7@192.168.1.11 seq=5 ttl=64 time=7.589 ms
SendData 7@192.168.1.11 seq=5 ttl=64 time=5.494 ms
Close 7@192.168.1.11 seq=5 ttl=64 time=4.358 ms

VPCS>
```

Рис. 2.14: Эхо-запрос в TCP-моде

Далее откроем Wireshark и проанализируем эхо-запрос по протоколу TCP. В поле канального уровня можем посмотреть mac-адреса источника и получателя. По нулевому и первому битам можем определить тип mac-адресов (получатель и источник - глобально администрируемые и одиночные, так как биты равны 0). В поле сетевого уровня указан протокол TCP и IP-адреса источника (192.168.1.12, то есть PC-2) и получателя (192.168.1.11, то есть PC-2).

В поле протокола TCP можем узнать порты источника (20665) и получателя (7). А также посмотреть, как работает handshake протокола TCP. На первом шаге установлен флаг SYN (рис. fig. 2.15), а также Порядковому номеру (Sequence Number) присвоено начальное 32-битное значение ISSa (в нашем случае 890542105).

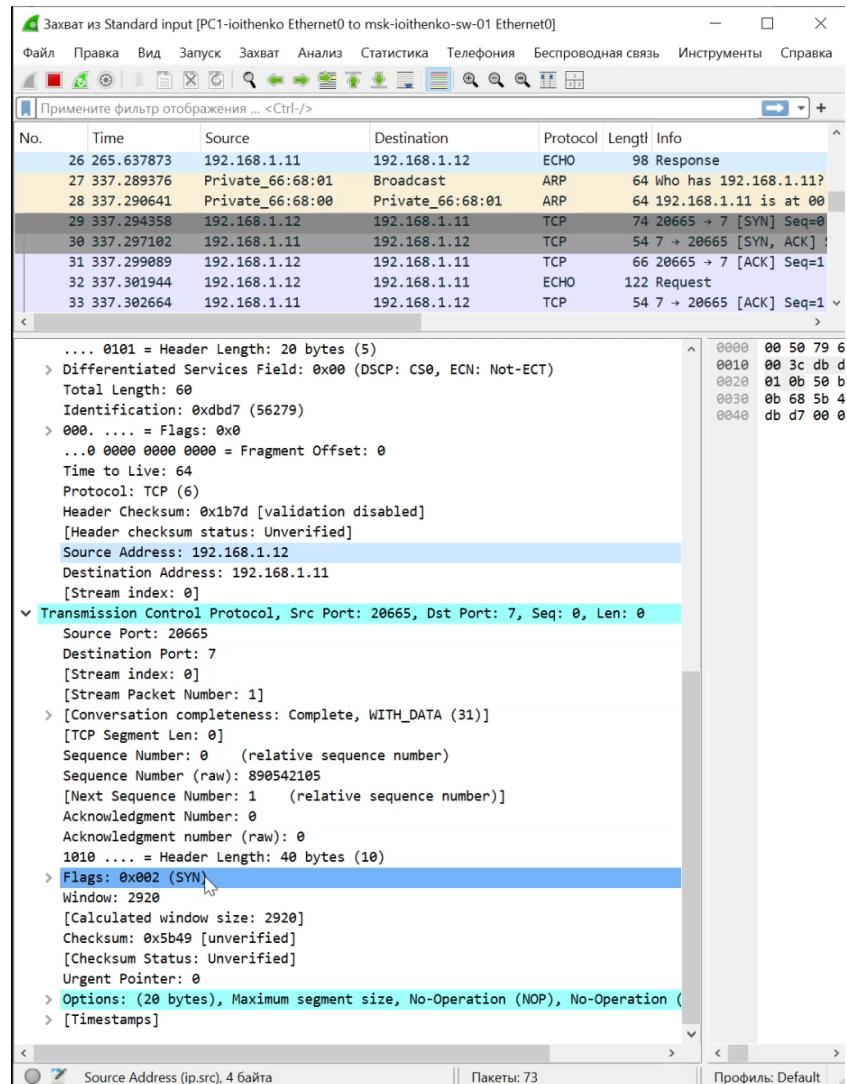


Рис. 2.15: Полученная информация по эхо-запросу в TCP-моде к узлу РС-1

На втором шаге установлены флаги SYN и ACK (рис. fig. 2.16).

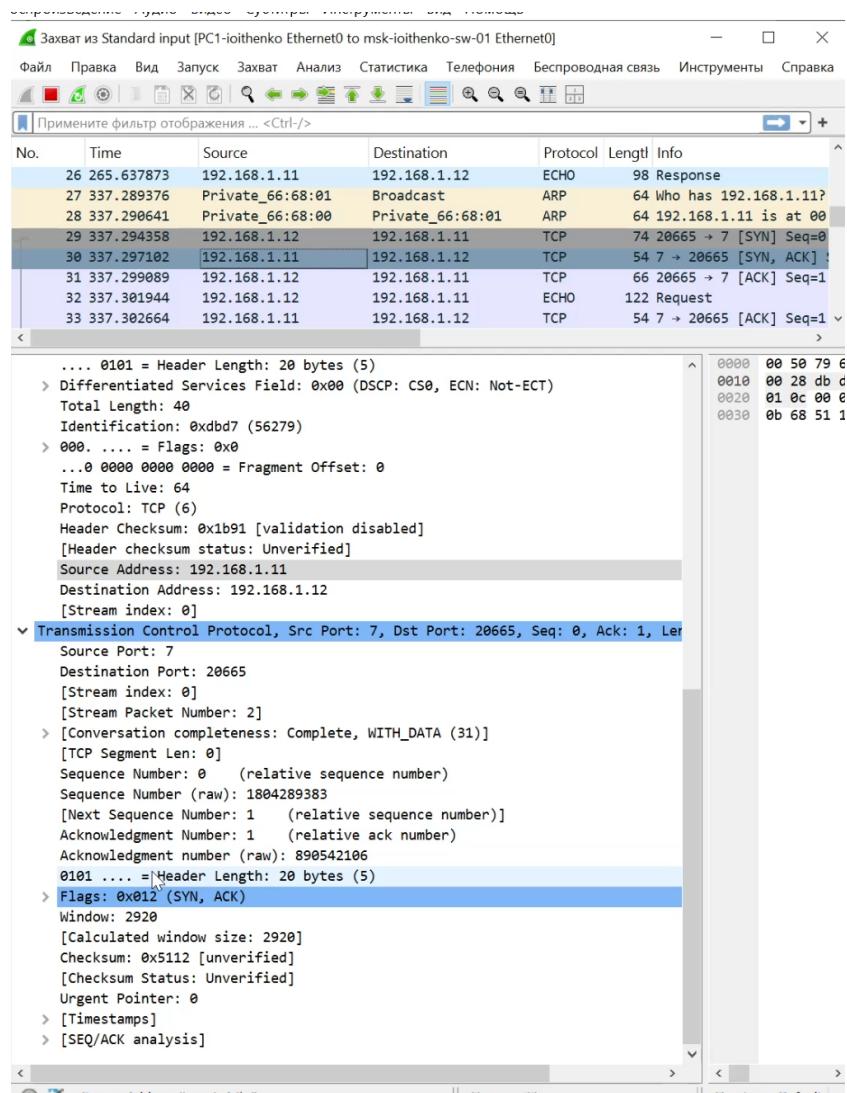


Рис. 2.16: Полученная информация по эхо-запросу в TCP-моде к узлу РС-1

На третьем шаге установлен флаг ACK (рис. fig. 2.17).

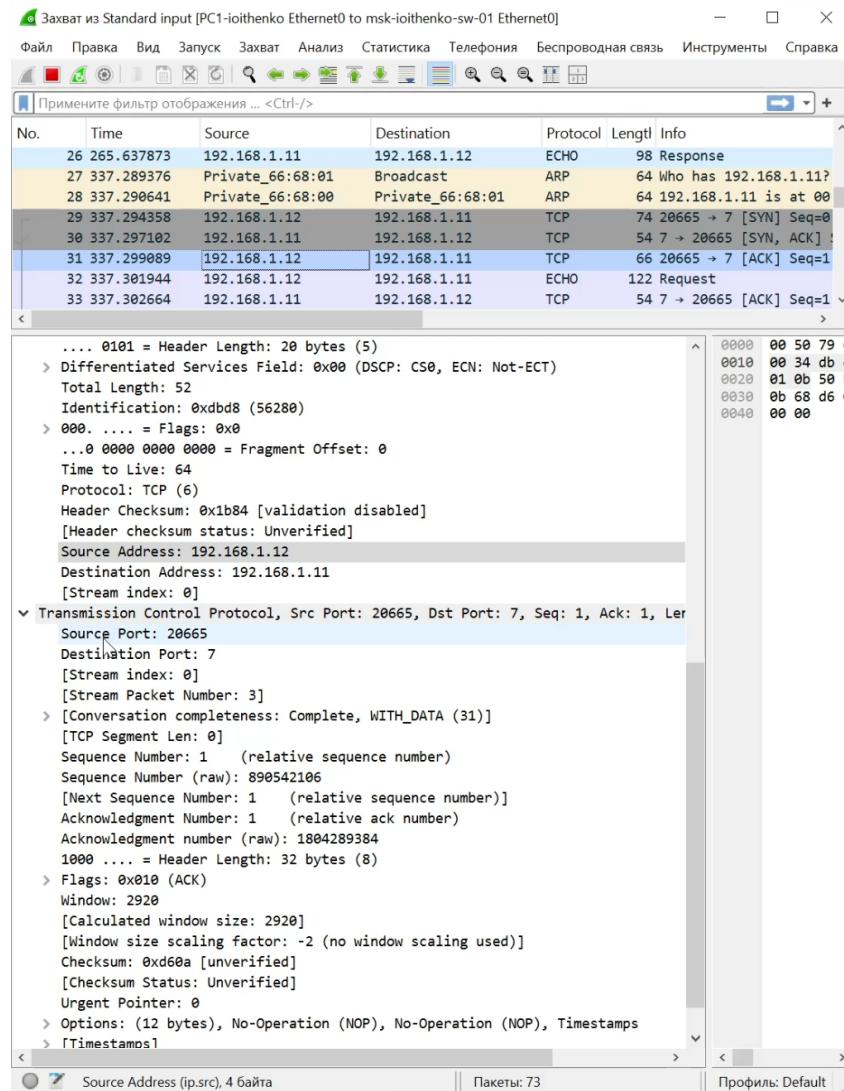


Рис. 2.17: Полученная информация по эхо-запросу в TCP-моде к узлу PC-1

Теперь можно остановить захват трафика в Wireshark.

Теперь нам нужно построить в GNS3 топологию сети (рис. fig. 2.18), состоящей из маршрутизатора FRR, коммутатора Ethernet и оконечного устройства.

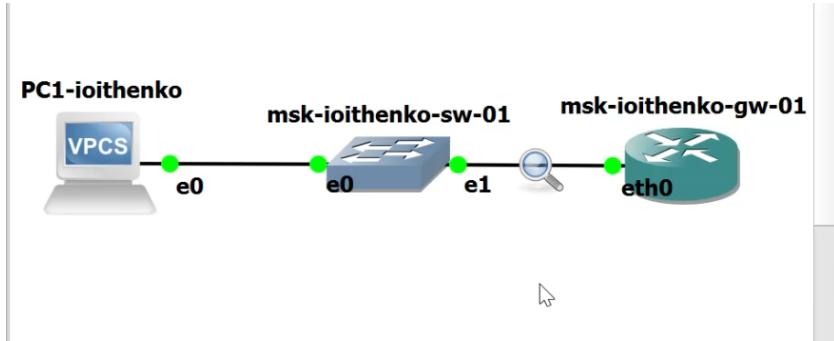


Рис. 2.18: Топология сети с маршрутизатором FRR

К сожалению, у меня возникают проблемы после запуска терминала маршрутизатора FRR, и нет возможности вводить команды (рис. fig. 2.19). Ошибка: kernel panic.

```
[ 2.115198]    kernel_init+0x11/0x120
[ 2.115198]    ret_from_fork+0x22/0x30
[ 2.115198] </TASK>
[ 2.115198] Modules linked in:
[ 2.115198] ---[ end trace b197cb4e10a7bbf0 ]---
[ 2.115198] RIP: 0010:native_write_msr+0x6/0x30
[ 2.115198] Code: b8 40 00 00 00 ff 00 d8 89 ef e8 a5 56 fe ff eb ca e8 4e 25
78 00 66 66 2e 0f 1f 84 00 00 00 00 0f 1f 00 89 f0 89 f9 0f 30 <66> 90 31 c0
89 c2 89 c1 89 c6 89 c7 c3 cc cc cc cc 48 c1 e2 20 48
[ 2.115198] RSP: 0018:fffffa91080013e30 EFLAGS: 00010056
[ 2.115198] RAX: 0000000000000040 RBX: 0000000000000000 RCX: 00000000000000830
[ 2.115198] RDX: 0000000000000000 RSI: 0000000000000040 RDI: 00000000000000830
[ 2.115198] RBP: 0000000000000040 R08: 0000000000000000 R09: 0000000000000000
[ 2.115198] R10: 0000000000000000 R11: 0000000000000000 R12: ffffffff8855f0e0
[ 2.115198] R13: 0000000000080000 R14: 00000000000151c0 R15: 0000000000000000
[ 2.115198] FS: 0000000000000000(0000) GS:fffff8bc4f200000(0000) knlGS:00000
000000000000
[ 2.115198] CS: 0010 DS: 0000 ES: 0000 CRO: 0000000080050033
[ 2.115198] CR2: fffff8bc447801000 CR3: 00000000700a0000 CR4: 00000000000006b0
[ 2.115198] Kernel panic - not syncing: Attempted to kill init! exitcode=0x000
00000b
[ 2.115198] ---[ end Kernel panic - not syncing: Attempted to kill init! exit
code=0x00000000b ]---
```

Рис. 2.19: Терминал маршрутизатора FRR

В рабочей области GNS3 разместим VPCS, коммутатор Ethernet и маршрутизатор VyOS. Изменим отображаемые названия устройств. Коммутатору присвоим название msk-ioithenko-sw-01, маршрутизатору – по принципу msk-ioithenko-gw-01, VPCS – по принципу PC1-ioithenko. Включим захват трафика на соединении между коммутатором и маршрутизатором (рис. fig. 2.20) (появится значок лупы). Запустим все устройства проекта и откроем консоль всех устройств проекта.



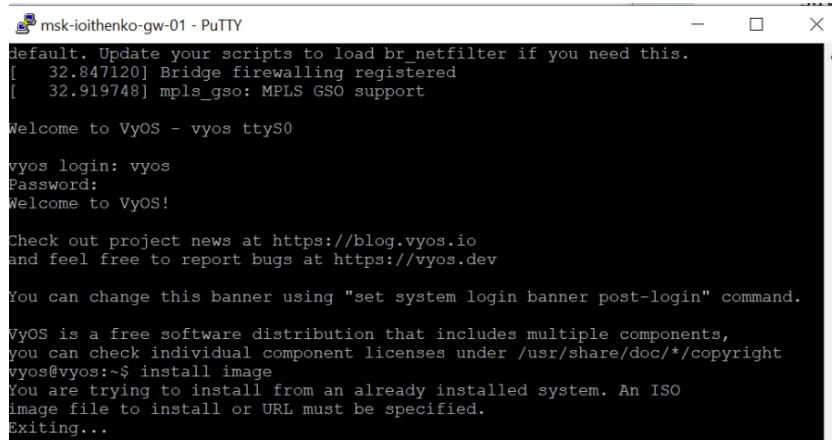
Рис. 2.20: Топология сети с маршрутизатором VyOS

Откроем окно терминала PC-1 и настроим IP-адресацию для интерфейса этого узла (рис. fig. 2.21).

```
PC1-ioithenko - PuTTY
Executing the startup file
Hostname is too long. (Maximum 12 characters)
VPCS> 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
VPCS> save
Saving startup configuration to startup.vpc
. done
VPCS> show ip
NAME      : VPCS[1]
IP/MASK   : 192.168.1.10/24
GATEWAY   : 192.168.1.1
DNS       : I
MAC       : 00:50:79:66:68:00
LPORT     : 10001
RHOST:PORT: 127.0.0.1:10002
MTU:      : 1500
VPCS>
```

Рис. 2.21: Настройка IP-адресации для интерфейса узла PC-1

Настроим маршрутизатор VyOS (рис. fig. 2.22): Во-первых, после загрузки нужно ввести логин vyos и пароль vyos. В рабочем режиме в командной строке отобразится символ \$. Далее надо установить систему на диск с помощью команды install image, но у меня эта система уже была.



```
msk-ioithenko-gw-01 - PuTTY
default. Update your scripts to load br_nf if you need this.
[ 32.847120] Bridge firewalling registered
[ 32.919748] mpls_gso: MPLS GSO support

Welcome to VyOS - vyos ttyS0

vyos login: vyos
Password:
Welcome to VyOS!

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

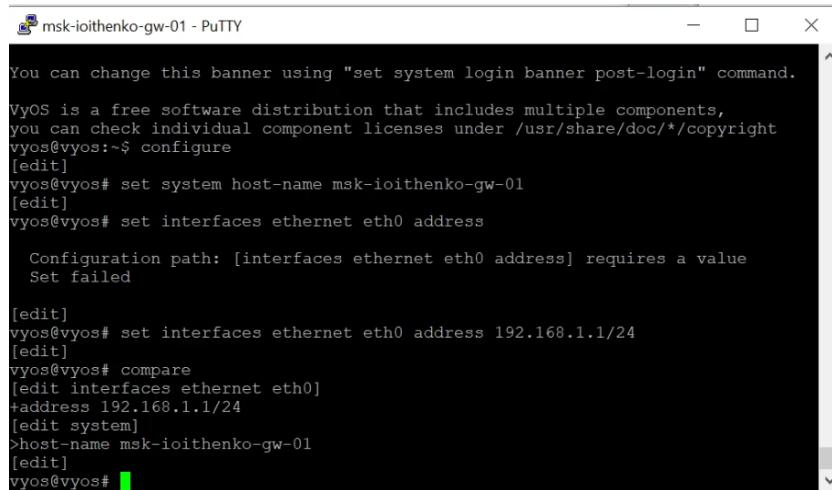
You can change this banner using "set system login banner post-login" command.

VyOS is a free software distribution that includes multiple components,
you can check individual component licenses under /usr/share/doc/*copyright
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...
```

Рис. 2.22: Логин, проверка установки системы на диск

Следующим шагом перейдем в режим конфигурирования с помощью команды `configure`. Изменим имя устройства командой `set system host-name msk-ioithenko-gw-01` Зададим IP-адрес на интерфейсе `eth0` командой `set interfaces ethernet eth0 address 192.168.1.1/24` Посмотрим внесённые в конфигурацию изменения с помощью команды `compare` (рис. fig. 2.23).



```
msk-ioithenko-gw-01 - PuTTY
You can change this banner using "set system login banner post-login" command.

VyOS is a free software distribution that includes multiple components,
you can check individual component licenses under /usr/share/doc/*copyright
vyos@vyos:~$ configure
[edit]
vyos@vyos# set system host-name msk-ioithenko-gw-01
[edit]
vyos@vyos# set interfaces ethernet eth0 address
Configuration path: [interfaces ethernet eth0 address] requires a value
Set failed
[edit]
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-ioithenko-gw-01
[edit]
vyos@vyos#
```

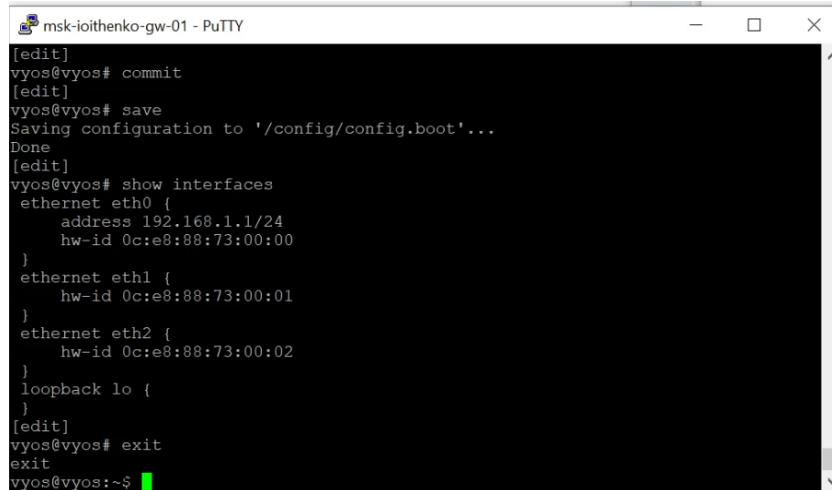
Рис. 2.23: Режим конфигурирования

Далее применим изменения в конфигурации и сохраним саму конфигурацию с помощью команд `commit` и `save` (рис. fig. 2.24).

```
vyos@vyos# commit
Can't configure both static IPv4 and DHCP address on the same interface
[[interfaces ethernet eth0]] failed
Commit failed
[edit]
vyos@vyos# delete interfaces ethernet eth0 address dhcp
[edit]
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
```

Рис. 2.24: Режим конфигурирования

Посмотрим информацию об интерфейсах маршрутизатора с помощью команды `show interfaces` (рис. fig. 2.25). Выйдем из режима конфигурирования, используя команду `exit`.



```
[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:e8:88:73:00:00
  }
  ethernet eth1 {
    hw-id 0c:e8:88:73:00:01
  }
  ethernet eth2 {
    hw-id 0c:e8:88:73:00:02
  }
  loopback lo {
  }
[edit]
vyos@vyos# exit
exit
vyos@vyos:~$
```

Рис. 2.25: Режим конфигурирования

Теперь проверим подключение. Узел PC1 должен успешно отправлять эхопротокол на адрес маршрутизатора 192.168.1.1. Для проверки пингуем маршрутизатор (рис. fig. 2.26). Получили эхо-ответ (4 пакета).

```
VPCS> save
Saving startup configuration to startup.vpc
. done

VPCS> show ip

NAME      : VPCS[1]
IP/MASK   : 192.168.1.10/24
GATEWAY   : 192.168.1.1
DNS       :
MAC       : 00:50:79:66:68:00
LPORT     : 10001
RHOST:PORT: 127.0.0.1:10002
MTU:      : 1500

VPCS> ping 192.168.1.1
84 bytes from 192.168.1.1 icmp_seq=1 ttl=64 time=1.705 ms
84 bytes from 192.168.1.1 icmp_seq=2 ttl=64 time=1.599 ms
84 bytes from 192.168.1.1 icmp_seq=3 ttl=64 time=1.494 ms
84 bytes from 192.168.1.1 icmp_seq=4 ttl=64 time=2.205 ms
84 bytes from 192.168.1.1 icmp_seq=5 ttl=64 time=1.041 ms

VPCS>
```

Рис. 2.26: Пингование маршрутизатора

В окне Wireshark проанализируем полученную информацию (рис. fig. 2.27). В поле канального уровня можем посмотреть мас-адреса источника и получателя. По нулевому и первому битам можем определить тип мас-адресов (получатель и источник - глобально администрируемые и одиночные, так как биты равны 0). В поле сетевого уровня указан протокол ICMP и IP-адреса источника (192.168.1.10, то есть PC-1) и получателя (192.168.1.1, то есть маршрутизатор VyOS).

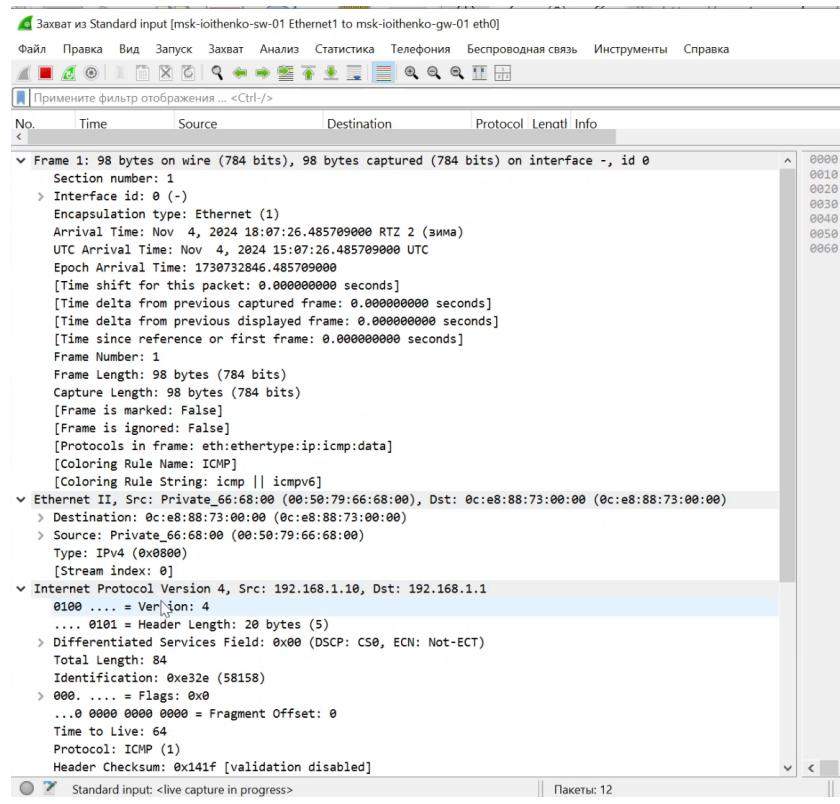


Рис. 2.27: Полученная информация в Wireshark по ICMP-сообщению

## **3 Выводы**

В процессе выполнения лабораторной работы я построила простейшие модели сети на базе коммутатора и маршрутизатора VyOS в GNS3, проанализировала трафик посредством Wireshark.