

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего  
образования



НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра «Вычислительные системы и технологии»

Сети и телекоммуникации

Отчет

по лабораторной работе №3

ARP протокол

ПРОВЕРИЛ:

\_\_\_\_\_

Гай В.Е.

СТУДЕНТ:

\_\_\_\_\_

Козменкова Е.П.  
18 В-2

Нижний Новгород

2020 г.

### Задание:

Для экспериментов использовать схему из первой лабораторной работы. Все IP-адреса (или маски) необходимо поменять так, чтобы адрес сети у всех компьютеров был один. Все действия должны быть выполнены в симуляторе сетей CORE.

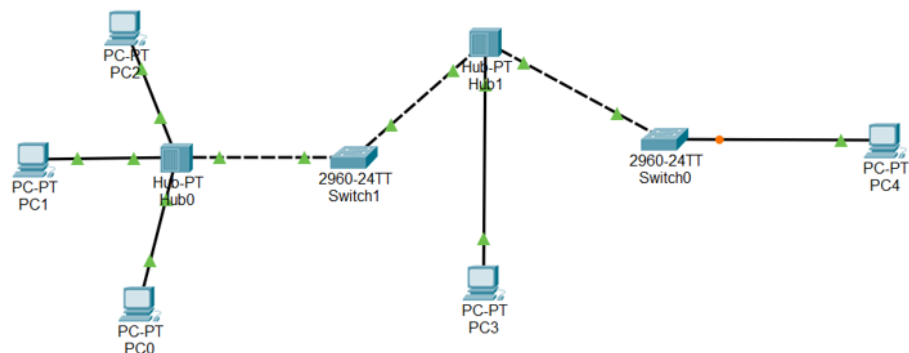
#### Часть 1. Формирование запроса и получение ответа

1. Начать захват пакетов при помощи Wireshark.
2. Сформировать кадр ARP-запроса с помощью утилиты PackETH и отправить его в сеть (компьютеры выбрать самостоятельно).
3. Убедиться, что был получен кадр ARP-ответа, соответствующий посланному запросу. Захваченные пакеты сохранить для отчета. Вывести arp таблицу (команда «arp»).
4. Прекратить захват пакетов.

#### Часть 2. ARP-спуфинг

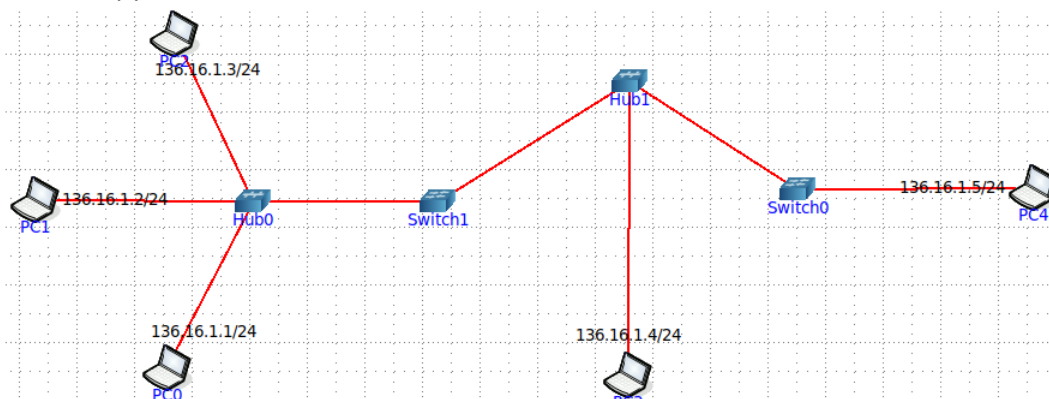
1. Выделить на схеме и обозначить три компьютера: А, В, Сервер.
2. Подготовить кадр ARP-ответа, направляемый Сервером хосту А с помощью программы PackETH. Кадр должен быть составлен так, чтобы MAC-адресу Сервера соответствовал IP-адрес хоста В. Вывести arp таблицу на хосте А. Отправить сформированный пакет от Сервера хосту А.
3. Организовать чат между узлами с помощью netcat.
4. Начать захват пакетов при помощи Wireshark на Сервере.
5. Попытаться установить соединение между хостом А и хостом В с помощью программы netcat (А отправляет сообщения В). Убедиться, что запросы от хоста А, направленные хосту В поступают на Сервер.
6. Прекратить захват пакетов.
7. Сохранить для отчета отправленный кадр ARP-ответа и несколько перехваченных пакетов, переданных на Сервер, arp таблицу хоста А.

### 2 Вариант



### Ход работы:

Смоделирую сеть из задания:



В задании также указано поменять все адреса так, чтобы компьютеры оказались в одной подсети.

**IP адреса:**

PC0 = 136.16.1.1/24

PC1 = 136.16.1.2/24

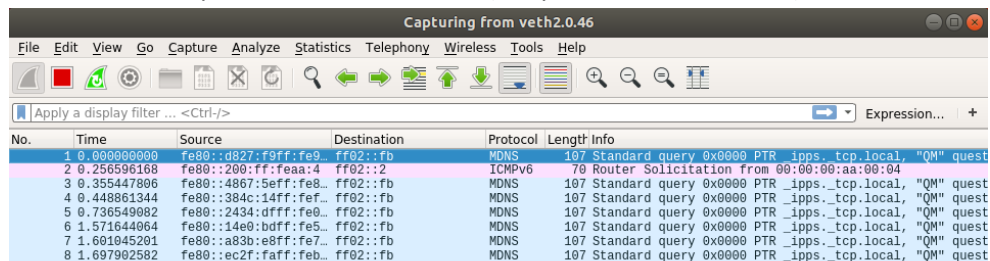
PC2 = 136.16.1.3/24

PC3 = 136.16.1.4/24

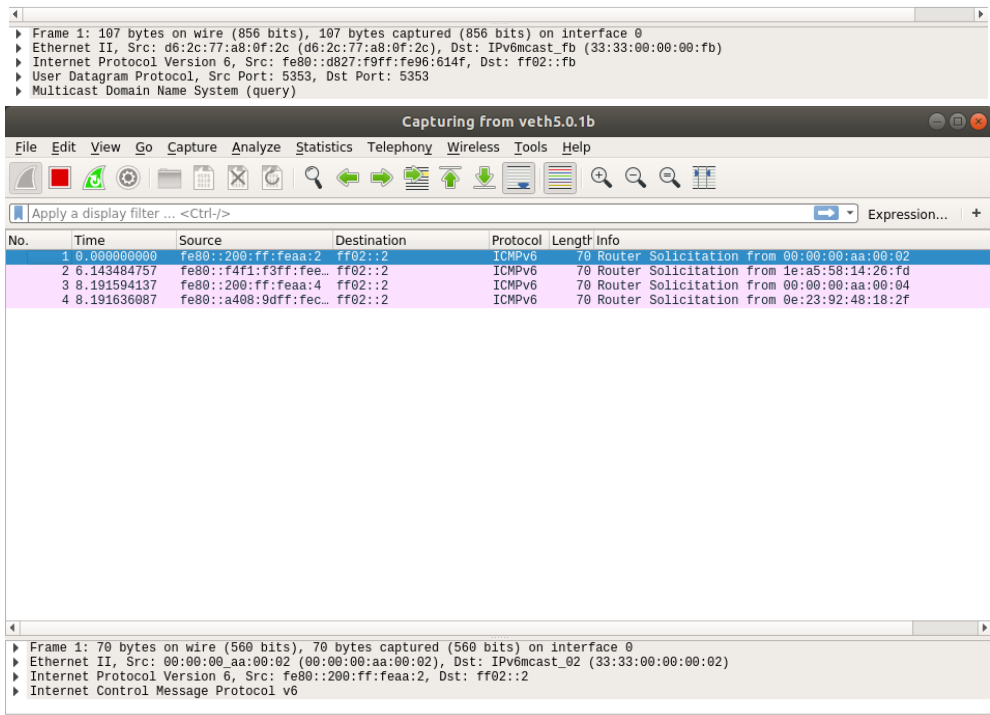
PC4 = 136.16.1.5/24

## Часть 1. Формирование запроса и получение ответа

Начинаю захват пакетов при помощи Wireshark. (Запускаю для PC1 и PC4)



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::d827:f9ff:fe9... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest
2	0.256596168	fe80::200:ff:feaa:4	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:04
3	0.355447806	fe80::4867:5eff:fe8... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest
4	0.448861344	fe80::384c:14ff:fe7... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest
5	0.736549882	fe80::2434:dfff:fe0... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest
6	1.571644064	fe80::14e0:bdff:fe5... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest
7	1.601845201	fe80::a83b:e8ff:fe7... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest
8	1.697902582	fe80::ec2f:fa9f:feb... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" quest



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::200:ff:feaa:2	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:02
2	6.143484757	fe80::f4f1:f3ff:fee... ff02::2		ICMPv6	70	Router Solicitation from 1e:a5:58:14:26:fd
3	8.191594137	fe80::200:ff:feaa:4	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:04
4	8.191636087	fe80::a408:9dff:fec... ff02::2		ICMPv6	70	Router Solicitation from 0e:23:92:48:18:2f

Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0  
Ethernet II, Src: 00:00:00:aa:00:02 (00:00:00:aa:00:02), Dst: IPv6mcast\_02 (33:33:00:00:00:02)  
Internet Protocol Version 6, Src: fe80::200:ff:feaa:2, Dst: ff02::2  
Internet Control Message Protocol v6

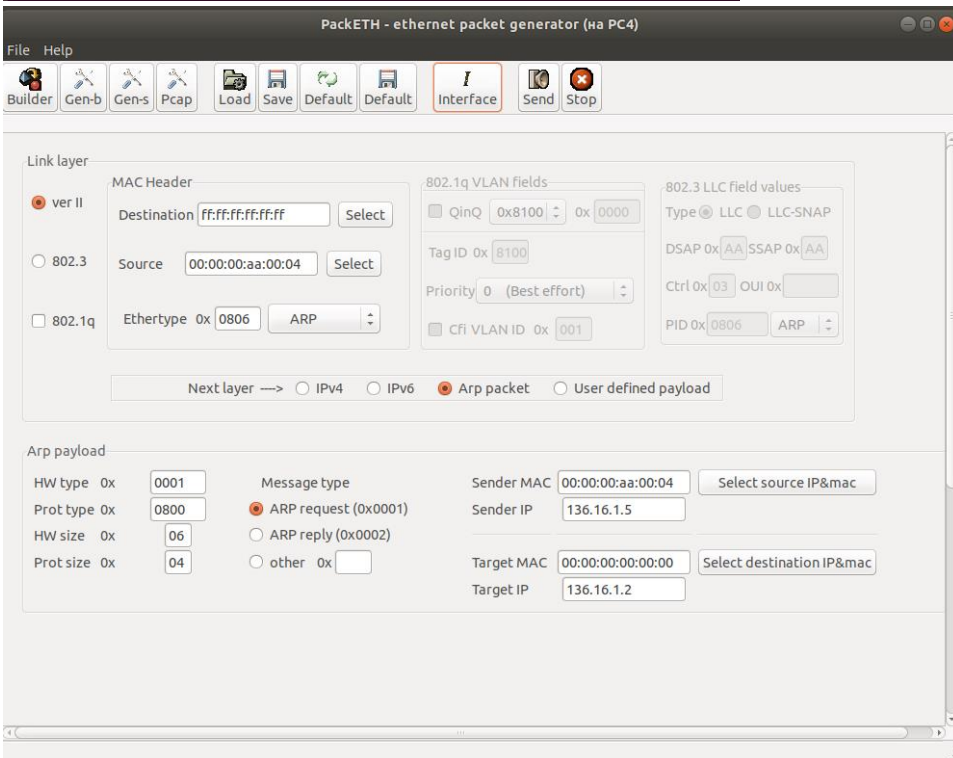
Сформирую кадр ARP-запроса с помощью утилиты RaskETH и отправлю его в сеть (Отправляю с компьютера PC4 на компьютер PC1). MAC-адрес отправителя узнаю из интерфейсов (команда ifconfig).

```

root@PC4:/tmp/pycore.40581/PC4.conf# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 136.16.1.5 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 fe80::200:ff:feaa:4 prefixlen 64 scopeid 0x20<link>
    ether 00:00:00:aa:00:04 txqueuelen 1000 (Ethernet)
    RX packets 205 bytes 22306 (22.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 11 bytes 866 (866.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Локальная петля (Loopback))
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```



Проверю, был ли получен кадр ARP-ответа, соответствующий посланному запросу.

PC1:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::2cec:86ff:feb...	ff02::2	ICMPv6	70	Router Solicitation from 2e:ec:86:b2:13:b0
2	4.052589580	00:00:00_aa:00:04	Broadcast	ARP	60	Who has 136.16.1.2? Tell 136.16.1.5
3	4.052626972	00:00:00_aa:00:01	00:00:00_aa:00:04	ARP	42	136.16.1.2 is at 00:00:00_aa:00:01
4	14.336438613	fe80::b4e7:f5ff:fee...	ff02::2	ICMPv6	70	Router Solicitation from 1a:fb:a0:64:5d:9e

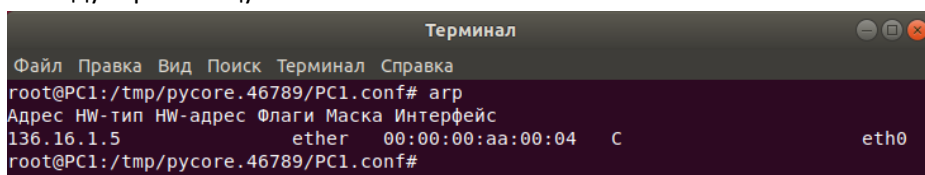
  

<b>Address Resolution Protocol (request)</b> Hardware type: Ethernet (1) Protocol type: IPv4 (0x0800) Hardware size: 6 Protocol size: 4 Opcode: request (1) Sender MAC address: 00:00:00_aa:00:04 (00:00:00_aa:00:04) Sender IP address: 136.16.1.5 Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00) Target IP address: 136.16.1.2	<b>Address Resolution Protocol (reply)</b> Hardware type: Ethernet (1) Protocol type: IPv4 (0x0800) Hardware size: 6 Protocol size: 4 Opcode: reply (2) Sender MAC address: 00:00:00_aa:00:01 (00:00:00_aa:00:01) Sender IP address: 136.16.1.2 Target MAC address: 00:00:00_aa:00:04 (00:00:00_aa:00:04) Target IP address: 136.16.1.5
--	--

PC4:

15	76.803094090	fe80::5866:52ff:feb...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" ques
16	77.380361639	fe80::ecbe:44ff:feb...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" ques
17	81.919220312	fe80::5866:52ff:feb...	ff02::2	ICMPv6	70	Router Solicitation from 5a:66:52:b4:8d:3b
18	83.800434925	00:00:00_aa:00:04	Broadcast	ARP	60	Who has 136.16.1.2? Tell 136.16.1.5
19	83.800486412	00:00:00_aa:00:01	00:00:00_aa:00:04	ARP	42	136.16.1.2 is at 00:00:00_aa:00:01

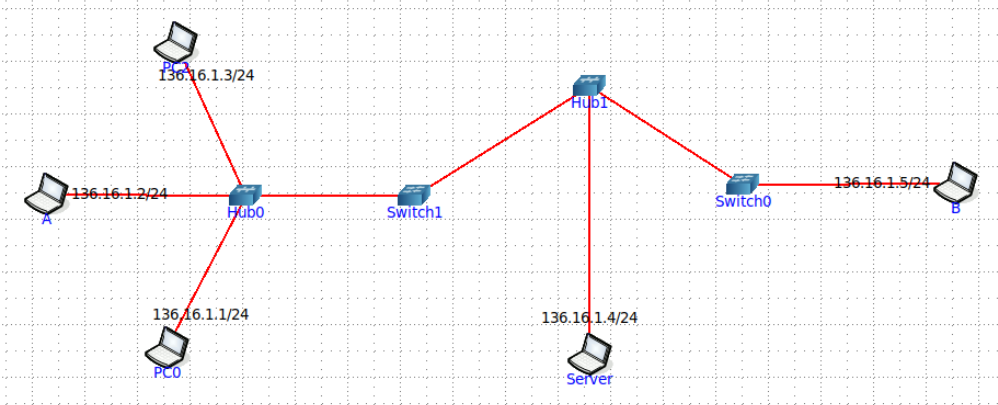
Выведу arp таблицу:



Прекращу захват пакетов.

Часть 2. ARP-спуфинг

Выделю на схеме три компьютера: А, В, Сервер.



Так как они соединены через hub, сервер так же будет получать пакеты, пересылаемые между А и В.

Файл Правка Вид Поиск Терминал Справка

root@A:/tmp/pycore.46789/A.conf# nc -l -p 9000  
123

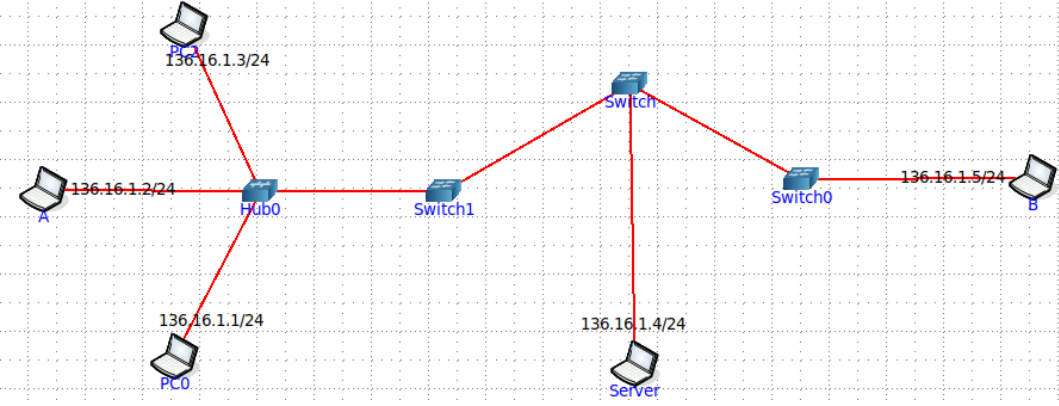
Файл Правка Вид Поиск Терминал Справка

root@B:/tmp/pycore.46789/B.conf# nc 136.16.1.2 9000  
123

19	14.335957552	fe80::38e8:d6ff:fec...	ff02::2	ICMPv6	70 Router Solicitation from 3a:e8:d6:c2:ae:dc
20	20.654211135	00:00:00_aa:00:03	Broadcast	ARP	42 Who has 136.16.1.2? Tell 136.16.1.5
21	20.654244728	00:00:00_aa:00:01	00:00:00_aa:00:03	ARP	42 136.16.1.2 is at 00:00:00_aa:00:01
22	20.654252678	136.16.1.5	136.16.1.2	TCP	74 48896 → 9000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SA
23	20.654269951	136.16.1.2	136.16.1.5	TCP	74 9000 → 48896 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0
24	20.654283211	136.16.1.5	136.16.1.2	TCP	66 48896 → 9000 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval
25	24.760797903	136.16.1.5	136.16.1.2	TCP	70 48896 → 9000 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=4
26	24.760826603	136.16.1.2	136.16.1.5	TCP	66 9000 → 48896 [ACK] Seq=1 Ack=5 Win=65280 Len=0 TSval

Frame 25: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0  
Ethernet II, Src: 00:00:00\_aa:00:03 (00:00:00\_aa:00:03), Dst: 00:00:00\_aa:00:01 (00:00:00\_aa:00:01)  
Internet Protocol Version 4, Src: 136.16.1.5, Dst: 136.16.1.2  
Transmission Control Protocol, Src Port: 48896, Dst Port: 9000, Seq: 1, Ack: 1, Len: 4  
Data (4 bytes)  
0000 00 00 00 aa 00 01 00 00 00 aa 00 03 08 00 45 00 .....E  
0010 00 38 6f bf 40 00 40 06 b8 d9 88 10 01 05 88 10 ..8o @ @ @  
0020 01 02 bf 00 23 28 b4 58 27 f2 fe ea 1a 01 80 18 ....-#(X'n  
0030 01 f6 d7 01 00 00 01 01 00 0a 11 fc 2c 90 51 15 .....Q  
0040 50 55 31 32 33 0a PU123

Изменяю hub на switch.



Теперь пакеты, проходящие между А и В, на сервер не попадают:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::ac45:50ff:fe7...	ff02::2	ICMPv6	70	Router Solicitation from 56:0d:f3:59:cd:34
2	2.047879252	fe80::200:ff:feaa:4	ff02::2	ICMPv6	70	Router Solicitation from 09:00:00:aa:00:04
3	2.047952418	fe80::804:b1ff:fe0...	ff02::2	ICMPv6	70	Router Solicitation from 0a:04:b1:80:eb:64
4	2.047962905	fe80::e41d:77ff:fe...	ff02::2	ICMPv6	70	Router Solicitation from 9a:19:9d:20:6d:8e
5	4.096229942	fe80::9819:9dff:fe2...	ff02::2	ICMPv6	70	Router Solicitation from 9a:19:9d:20:6d:8e
6	4.096239798	fe80::200:ff:feaa:0	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:00
7	6.144114878	fe80::58a0:10ff:fec...	ff02::2	ICMPv6	70	Router Solicitation from 0a:04:b1:80:eb:64
8	6.144195333	fe80::200:ff:feaa:2	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:02
9	6.528368985	fe80::036:d0ff:fe0...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" que
10	6.754119796	fe80::58a0:10ff:fec...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" que
11	7.714502643	fe80::88c1:7ff:fed8...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" que
12	8.192001639	fe80::88c1:7ff:fed8...	ff02::2	ICMPv6	70	Router Solicitation from 8a:c1:07:d8:fd:cb
13	8.192024171	fe80::200:ff:feaa:1	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:01
14	8.292155458	fe80::804:b1ff:fe0...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" que
15	8.494487532	fe80::9819:9dff:fe2...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" que
16	8.800425316	fe80::801c:deff:fe2...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" que
17	10.240021943	fe80::200:ff:feaa:3	ff02::2	ICMPv6	70	Router Solicitation from 00:00:00:aa:00:03
18	12.287895237	fe80::801c:deff:fe2...	ff02::2	ICMPv6	70	Router Solicitation from 82:1c:de:29:10:8a
19	12.287910019	fe80::036:d0ff:fe0...	ff02::2	ICMPv6	70	Router Solicitation from 82:1c:de:29:10:8a

Подготавливаю кадр ARP-ответа, направляемый Сервером хосту А с помощью программы PackETH. Составлю его так, чтобы MAC-адресу Сервера соответствовал IP-адрес хоста В.

File Help

Builder Gen-b Gen-s Pcap Load Save Default Default Interface Send Stop

Link layer

☒ ver II

☐ 802.3

☐ 802.1q

MAC Header

Destination 00:00:00:aa:00:01 Select

Source 00:00:00:aa:00:04 Select

Ethertype 0x0806 ARP

802.1q VLAN fields

☐ QinQ 0x8100 0x0000

Tag ID 0x8100

Priority 0 (Best effort)

☐ Cfi VLAN ID 0x001

802.3 LLC field values

Type ☒ LLC ☐ LLC-SNAP

DSAP 0xAA SSAP 0xAA

Ctrl 0x03 OUI 0x

PID 0x0806 ARP

Next layer --> ☐ IPv4 ☐ IPv6 ☒ Arp packet ☐ User defined payload

Arp payload

HW type 0x0001

Prot type 0x0800

HW size 0x06

Prot size 0x04

Message type

☐ ARP request (0x0001)

☒ ARP reply (0x0002)

☐ other 0x

Sender MAC 00:00:00:aa:00:04 Select source IP&mac

Sender IP 136.16.1.5

Target MAC 00:00:00:aa:00:01 Select destination IP&mac

Target IP 136.16.1.2

Выведу arp таблицу на хосте А. Сейчас она содержит MAC-адрес компьютера В.

```
root@A:/tmp/pycore.46789/A.conf# arp
Адрес HW-тип HW-адрес Флаги Маска Интерфейс
136.16.1.5 ether 00:00:00:aa:00:03 C eth0
root@A:/tmp/pycore.46789/A.conf#
```

Отправлю сформированный пакет от Сервера хосту А и проверю arp таблицу. Теперь в ней MAC-адрес Сервера.

```
root@A:/tmp/pycore.46789/A.conf# arp
Адрес HW-тип HW-адрес Флаги Маска Интерфейс
136.16.1.5 ether 00:00:00:aa:00:04 C eth0
root@A:/tmp/pycore.46789/A.conf#
```

Попробую проверить, проходит ли ping между компьютерами А и В.

```
root@A:/tmp/pycore.46789/A.conf# ping 136.16.1.5
PING 136.16.1.5 (136.16.1.5) 56(84) bytes of data.
^C
--- 136.16.1.5 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2038ms
```

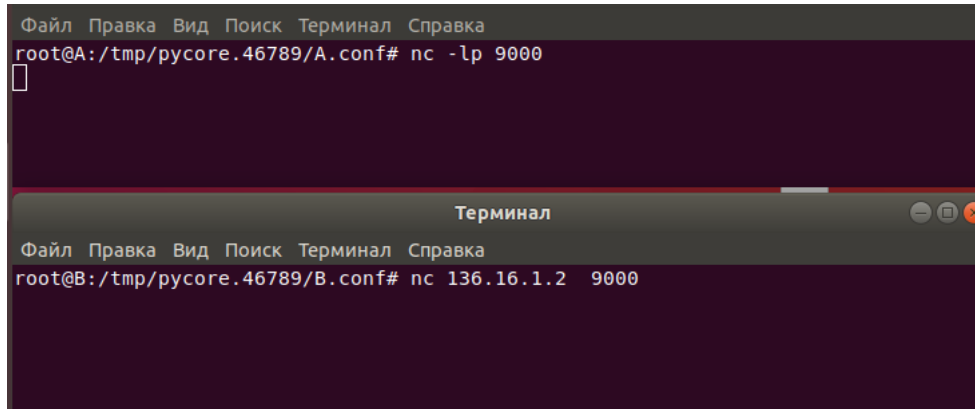
Не проходит. Это происходит из-за того, что компьютер А не получает ответ от компьютера В (из-за перехвата пакетов). Но если немного подождать, то можно увидеть, что рано или поздно пакеты все-таки дойдут (хоть и с потерей). Проверю arp-таблицу. В ней видно, что теперь у компьютера В стоит правильный MAC-адрес.

```
root@A:/tmp/pycore.46789/A.conf# ping 136.16.1.5
PING 136.16.1.5 (136.16.1.5) 56(84) bytes of data.
64 bytes from 136.16.1.5: icmp_seq=9 ttl=64 time=0.193 ms
64 bytes from 136.16.1.5: icmp_seq=10 ttl=64 time=0.123 ms
64 bytes from 136.16.1.5: icmp_seq=11 ttl=64 time=0.116 ms
64 bytes from 136.16.1.5: icmp_seq=12 ttl=64 time=0.121 ms
^C
--- 136.16.1.5 ping statistics ---
12 packets transmitted, 4 received, 66% packet loss, time 11254ms
rtt min/avg/max/mdev = 0.116/0.138/0.193/0.032 ms
root@A:/tmp/pycore.46789/A.conf# arp
Адрес HW-тип HW-адрес Флаги Маска Интерфейс
136.16.1.5 ether 00:00:00:aa:00:03 C eth0
root@A:/tmp/pycore.46789/A.conf#
```

Взглянем поближе, что же там все-таки происходит.

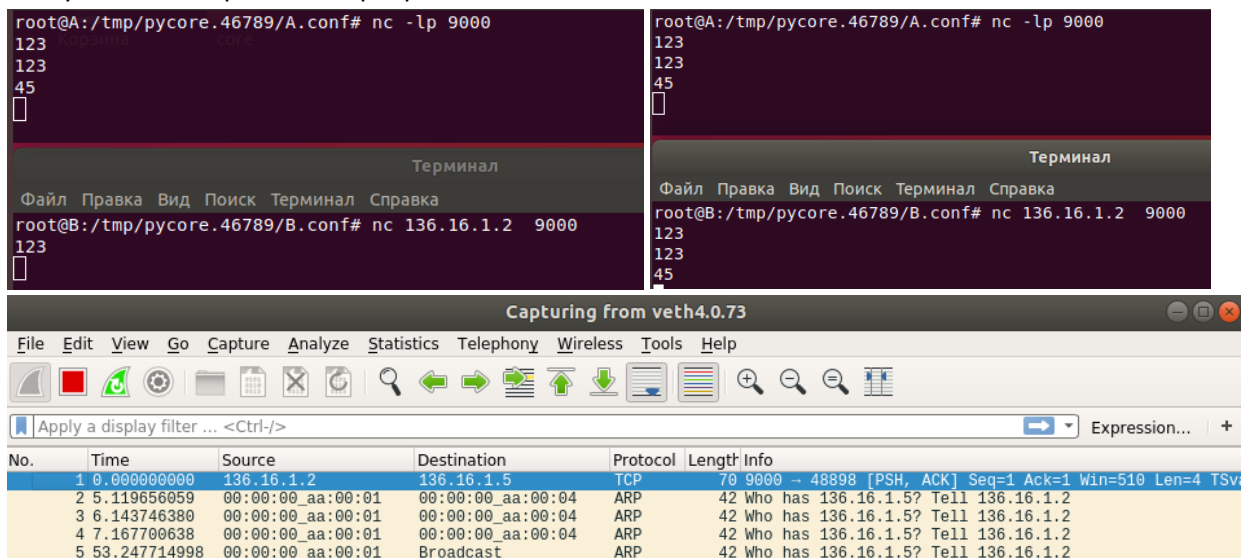


Организирую чат между узлами с помощью netcat.



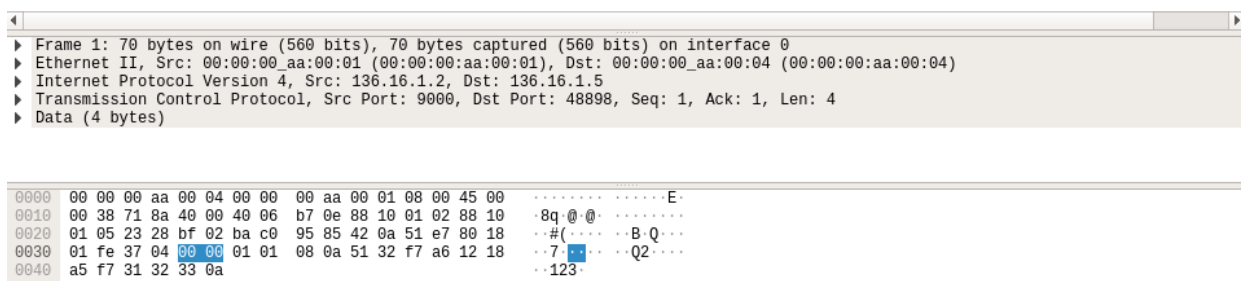
The image shows two terminal windows. The top window is titled 'Терминал' and shows a netcat listener on host A: `root@A:/tmp/pycore.46789/A.conf# nc -lp 9000`. The bottom window is also titled 'Терминал' and shows a netcat client on host B: `root@B:/tmp/pycore.46789/B.conf# nc 136.16.1.2 9000`.

Начну захват пакетов при помощи WireShark на Сервере. Попытаюсь установить соединение между хостом А и хостом В с помощью программы netcat (А отправляет сообщения В) и проверить, поступают ли запросы на Сервер.



The image shows a WireShark packet capture window titled 'Capturing from veth4.0.73'. It displays a list of captured packets. The first packet is a TCP SYN from 136.16.1.2 to 136.16.1.5. The second packet is an ARP request from 00:00:00:aa:00:01 to 00:00:00:aa:00:04. The third packet is an ARP request from 00:00:00:aa:00:01 to 00:00:00:aa:00:04. The fourth packet is an ARP request from 00:00:00:aa:00:01 to 00:00:00:aa:00:04. The fifth packet is an ARP request from 00:00:00:aa:00:01 to Broadcast. The sixth packet is a TCP SYN from 136.16.1.2 to 136.16.1.5.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	136.16.1.2	136.16.1.5	TCP	70	9000 -> 48898 [PSH, ACK] Seq=1 Ack=1 Win=510 Len=4 TSv
2	5.119656059	00:00:00:aa:00:01	00:00:00:aa:00:04	ARP	42	Who has 136.16.1.5? Tell 136.16.1.2
3	6.143746380	00:00:00:aa:00:01	00:00:00:aa:00:04	ARP	42	Who has 136.16.1.5? Tell 136.16.1.2
4	7.167709638	00:00:00:aa:00:01	00:00:00:aa:00:04	ARP	42	Who has 136.16.1.5? Tell 136.16.1.2
5	53.247714998	00:00:00:aa:00:01	Broadcast	ARP	42	Who has 136.16.1.5? Tell 136.16.1.2



The image shows the details of the first packet in the WireShark capture. It is a TCP SYN packet from 136.16.1.2 to 136.16.1.5. The details pane shows the following information:

- Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
- Ethernet II, Src: 00:00:00:aa:00:01 (00:00:00:aa:00:01), Dst: 00:00:00:aa:00:04 (00:00:00:aa:00:04)
- Internet Protocol Version 4, Src: 136.16.1.2, Dst: 136.16.1.5
- Transmission Control Protocol, Src Port: 9000, Dst Port: 48898, Seq: 1, Ack: 1, Len: 4
- Data (4 bytes)

The packet bytes pane shows the raw data of the packet, which is a TCP SYN packet.

Отсюда видно, что сначала нами был перехвачен пакет с числом 123 (вторым, первое дошло до компьютера В еще до запуска arp-ответа от Сервера). Потом компьютер А еще несколько раз пытается узнать MAC-адрес компьютера В, но запросы отправляет по MAC-адресу Сервера. После чего компьютер А отправляет уже широковещательный запрос и находит «настоящий» компьютер В. Как видно из второго скрина чата, недошедшая информация доходит до компьютера В. Прекращаю захват пакетов.

## Сохраню для отчета отправленный кадр ARP-ответа

19	591.666194080	00:00:00_aa:00:04	00:00:00_aa:00:01	ARP	60	136.16.1.5 is at 00:00:00:aa:00:04
▼ Address Resolution Protocol (reply)						
Hardware type: Ethernet (1)						
Protocol type: IPv4 (0x0800)						
Hardware size: 6						
Protocol size: 4						
Opcode: reply (2)						
Sender MAC address: 00:00:00_aa:00:04 (00:00:00:aa:00:04)						
Sender IP address: 136.16.1.5						
Target MAC address: 00:00:00_aa:00:01 (00:00:00:aa:00:01)						
Target IP address: 136.16.1.2						

## Перехваченный пакет, переданный на Сервер

1	0.000000000	136.16.1.2	136.16.1.5	TCP	70	9000 → 48898 [PSH, ACK] Seq=1 Ack=1 Win=510 Len=4 TS
2	5.119656059	00:00:00_aa:00:01	00:00:00_aa:00:04	ARP	42	Who has 136.16.1.5? Tell 136.16.1.2
3	6.143746380	00:00:00_aa:00:01	00:00:00_aa:00:04	ARP	42	Who has 136.16.1.5? Tell 136.16.1.2
▼ Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0						
▶ Ethernet II, Src: 00:00:00_aa:00:01 (00:00:00:aa:00:01), Dst: 00:00:00_aa:00:04 (00:00:00:aa:00:04)						
▶ Internet Protocol Version 4, Src: 136.16.1.2, Dst: 136.16.1.5						
▼ Transmission Control Protocol, Src Port: 9000, Dst Port: 48898, Seq: 1, Ack: 1, Len: 4						
Source Port: 9000						
Destination Port: 48898						
[Stream index: 0]						
[TCP Segment Len: 4]						
Sequence number: 1 (relative sequence number)						
[Next sequence number: 5 (relative sequence number)]						
Acknowledgment number: 1 (relative ack number)						
1000 .... = Header Length: 32 bytes (8)						
▶ Flags: 0x018 (PSH, ACK)						
Window size value: 510						
[Calculated window size: 510]						
[Window size scaling factor: -1 (unknown)]						
Checksum: 0x3704 [unverified]						
[Checksum Status: Unverified]						
Urgent pointer: 0						
▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps						
▶ [SEQ/ACK analysis]						
▶ [Timestamps]						
TCP payload (4 bytes)						
▼ Data (4 bytes)						
Data: 3132330a						
[Length: 4]						

Арп таблица хоста А до отправки с Сервера ARP-ответа, после отправки и после повторной отправки широковещательного запроса:

```
root@A:/tmp/pycore.46789/A.conf# arp
Адрес HW-тип HW-адрес Флаги Маска Интерфейс
136.16.1.5 ether 00:00:00:aa:00:03 C eth0
root@A:/tmp/pycore.46789/A.conf# arp
Адрес HW-тип HW-адрес Флаги Маска Интерфейс
136.16.1.5 ether 00:00:00:aa:00:04 C eth0
root@A:/tmp/pycore.46789/A.conf# arp
Адрес HW-тип HW-адрес Флаги Маска Интерфейс
136.16.1.5 ether 00:00:00:aa:00:03 C eth0
root@A:/tmp/pycore.46789/A.conf#
```