

ГУАП

КАФЕДРА № 42

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ  
ПРЕПОДАВАТЕЛЬ

ассистент

должность, уч. степень, звание

подпись, дата

Д. Д. Савельева

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 2

АНАЛИЗ ТРАФИКА КОМПЬЮТЕРНЫХ СЕТЕЙ УТИЛИТОЙ WIRESHARK

по курсу:

ИНФОКОММУНИКАЦИОННЫЕ СИСТЕМЫ И СЕТИ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. №

4326

подпись, дата

Г. С. Томчук

инициалы, фамилия

Санкт-Петербург 2025

## **1 Цель работы**

Цель работы: изучить структуру протокольных блоков данных, анализируя реальный трафик с помощью утилиты Wireshark.

## **2 Задание**

В процессе выполнения лабораторной работы выполняются наблюдения за передаваемым трафиком с компьютера пользователя в Интернет и в обратном направлении. Применение специализированной утилиты Wireshark позволяет наблюдать структуру передаваемых кадров, пакетов и сегментов данных различных сетевых протоколов. Требуется анализировать последовательности команд и назначение служебных данных, используемых для организации обмена данными в следующих протоколах: ARP, DNS, FTP, HTTP.

В качестве адреса сайта в заданиях следует использовать один из URL из первой лабораторной работы, свою страницу социальной сети или свой личный сайт. В таблице 1 представлены адреса сайтов, использованные в лабораторной работе № 1.

Таблица 1 — Вариант работы

№ Варианта	Исследуемые узлы
17	www.accounts.google.com www.coub.com www.fstec.ru

### 3 Ход выполнения работы

#### 3.1 Анализ трафика утилиты ping

Был отслежен и проанализирован утилитой Wireshark трафик, создаваемый утилитой ping при запуске из терминала. Для размера пакета были поочередно использованы значения от 100 до 8000. Не удалось выполнить команду со значением, превышающим 8000.

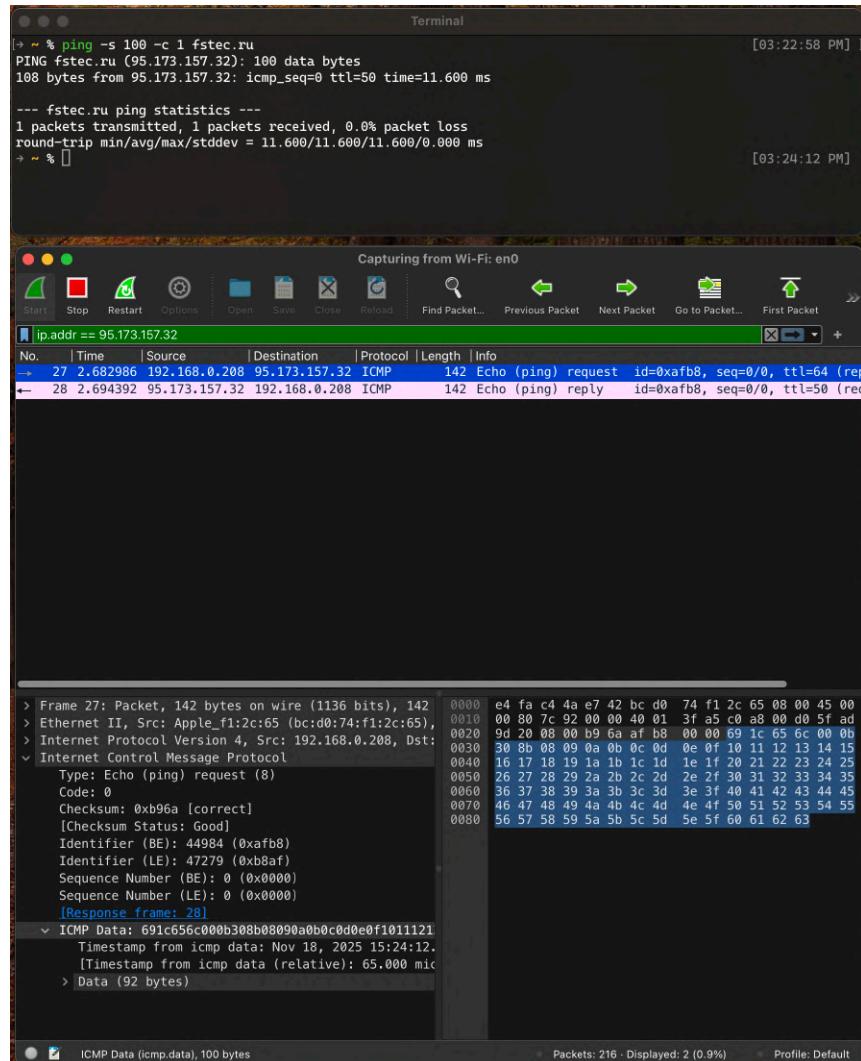


Рисунок 1 — ping с размером пакета 100

```

% ping -s 500 -c 1 fstec.ru
PING fstec.ru (95.173.157.32): 500 data bytes
508 bytes from 95.173.157.32: icmp_seq=0 ttl=50 time=13.819 ms
--- fstec.ru ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 13.819/13.819/13.819/nan ms

```

Capturing from Wi-Fi: en0

No.	Time	Source	Destination	Protocol	Length	Info
→ 1	6.000000	192.168.0.208	95.173.157.32	ICMP	542	Echo (ping) request id=0xe9b8
← 2	6.013560	95.173.157.32	192.168.0.208	ICMP	542	Echo (ping) reply id=0xe9b8

Frame 1: Packet, 542 bytes on wire (4336 bits), 542 b  
> Ethernet II, Src: Apple\_f1:2:c65 (bc:0:74:f1:2:c65), Dst: Internet Protocol Version 4, Src: 192.168.0.208, Dst: Internet Control Message Protocol  
Type: Echo (ping) request (8)  
Code: 0  
Checksum: 0x8149 [correct]  
[Checksum Status: Good]  
Identifier (BE): 59832 (0xe9b8)  
Identifier (LE): 47337 (0xb918)  
Sequence Number (BE): 0 (0x0000)  
Sequence Number (LE): 0 (0x0000)  
[Response frame: 2]  
ICMP Data [...] : 691c65c000b258208090a0b0c0d0e0f101  
Timestamp from icmp data: Nov 18, 2025 15:25:42.  
[Timestamp from icmp data (relative): 132.000 ms]  
> Data (492 bytes)

Frame 2: Packet, 542 bytes on wire (4336 bits), 542 b  
> Ethernet II, Src: Apple\_f1:2:c65 (bc:0:74:f1:2:c65), Dst: Internet Protocol Version 4, Src: 95.173.157.32, Dst: Internet Control Message Protocol  
Type: Echo (ping) reply (8)  
Code: 0  
Checksum: 0x8149 [correct]  
[Checksum Status: Good]  
Identifier (BE): 6329 (0x18b9)  
Identifier (LE): 47384 (0xb918)  
Sequence Number (BE): 0 (0x0000)  
Sequence Number (LE): 0 (0x0000)  
[Response frame: 2]  
ICMP Data [...] : 691c6607000748bf08090a0b0c0d0e0f101  
Timestamp from icmp data: Nov 18, 2025 15:26:47.  
[Timestamp from icmp data (relative): 130.000 ms]  
> Data (992 bytes)

Рисунок 2 — ping с размером пакета 500

```

% ping -s 1000 -c 1 fstec.ru
PING fstec.ru (95.173.157.32): 1000 data bytes
1008 bytes from 95.173.157.32: icmp_seq=0 ttl=50 time=14.332 ms
--- fstec.ru ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 14.332/14.332/14.332/nan ms

```

Capturing from Wi-Fi: en0

No.	Time	Source	Destination	Protocol	Length	Info
→ 1	6.000000	192.168.0.208	95.173.157.32	ICMP	1042	Echo (ping) request id=0x18b9
← 2	6.014063	95.173.157.32	192.168.0.208	ICMP	1042	Echo (ping) reply id=0x18b9

Frame 1: Packet, 1042 bytes on wire (8336 bits), 1042 b  
> Ethernet II, Src: Apple\_f1:2:c65 (bc:0:74:f1:2:c65), Dst: Internet Protocol Version 4, Src: 192.168.0.208, Dst: Internet Control Message Protocol  
Type: Echo (ping) request (8)  
Code: 0  
Checksum: 0x41e9 [correct]  
[Checksum Status: Good]  
Identifier (BE): 6329 (0x18b9)  
Identifier (LE): 47384 (0xb918)  
Sequence Number (BE): 0 (0x0000)  
Sequence Number (LE): 0 (0x0000)  
[Response frame: 2]  
ICMP Data [...] : 691c6607000748bf08090a0b0c0d0e0f101  
Timestamp from icmp data: Nov 18, 2025 15:26:47.  
[Timestamp from icmp data (relative): 130.000 ms]  
> Data (992 bytes)

Frame 2: Packet, 1042 bytes on wire (8336 bits), 1042 b  
> Ethernet II, Src: Apple\_f1:2:c65 (bc:0:74:f1:2:c65), Dst: Internet Protocol Version 4, Src: 95.173.157.32, Dst: Internet Control Message Protocol  
Type: Echo (ping) reply (8)  
Code: 0  
Checksum: 0x41e9 [correct]  
[Checksum Status: Good]  
Identifier (BE): 6329 (0x18b9)  
Identifier (LE): 47384 (0xb918)  
Sequence Number (BE): 0 (0x0000)  
Sequence Number (LE): 0 (0x0000)  
[Response frame: 2]  
ICMP Data [...] : 691c6607000748bf08090a0b0c0d0e0f101  
Timestamp from icmp data: Nov 18, 2025 15:27:04.  
[Timestamp from icmp data (relative): 167.000 ms]  
> Data (992 bytes)

Рисунок 3 — ping с размером пакета 1000

```

% ping -s 2000 -c 1 fstec.ru
PING fstec.ru (95.173.157.32): 2000 data bytes
4008 bytes from 95.173.157.32: icmp_seq=0 ttl=50 time=15.807 ms
--- fstec.ru ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 15.807/15.807/15.807/0.000 ms

```

Capturing from Wi-Fi: en0

No.	Time	Source	Destination	Protocol	Length	Info
32	5.158460	192.168.0.208	95.173.157.32	IPv4	1514	Fragmented IP protocol (proto=)
33	5.158517	192.168.0.208	95.173.157.32	ICMP	562	Echo (ping) request id=0x2eb9
34	5.173375	95.173.157.32	192.168.0.208	IPv4	1514	Fragmented IP protocol (proto=)
35	5.173396	95.173.157.32	192.168.0.208	ICMP	562	Echo (ping) reply id=0x2eb9

> Frame 32: Packet, 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0 at 00:00:00:00:00:00 [eth0] (Ethernet II, Src: Apple\_f1:2c:65 (bc:00:74:1f:2c:65), Dst: 95.173.157.32 [95:17:3:15:7:32]) on wire (12112 bits)
> Ethernet II, Src: Apple\_f1:2c:65 (bc:00:74:1f:2c:65), Dst: 95.173.157.32 (95:17:3:15:7:32)
> Internet Protocol Version 4, Src: 192.168.0.208, Dst: 95.173.157.32 (95:17:3:15:7:32)
> Data (1480 bytes)

0000 e4 fa c4 4a e7 42 bc d0 74 f1 2c 65 08 00 45 00
0010 05 f0 16 00 00 00 00 00 00 00 00 00 00 00 00 ad
0020 9d 28 08 00 52 2d 2e b9 00 00 69 1c 66 23 00 0b
0030 6f b6 08 00 00 00 00 0c 0d 0e 0f 10 11 12 13 14 15
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45
0070 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55
0080 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64 65
0090 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75
00a0 70 71 78 79 7a 7b 7c 7d 7e 7f 80 81 82 83 84 85
00b0 80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f
00c0 88 89 8a 8b 8c 8d 8e 8f 80 81 82 83 84 85
00d0 86 87 98 99 9a 9b 9c 9d 9e 9f 80 81 82 83 84 85
00e0 a6 a7 a8 a9 aa ab ac ad ae af b0 b1 b2 b3 b4 b5
00f0 b6 b7 b8 b9 ba bb bc bd be bf c0 c1 c2 c3 c4 c5
0100 c6 c7 c8 c9 ca cb cc cd ce cf d0 d1 d2 d3 d4 d5
0110 d6 d7 d8 d9 da db dc dd de df e0 e1 e2 e3 e4 e5
0120 e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 e0 e1 e2 e3 e4 e5
0130 f6 f7 f8 f9 f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 f0 f1
0140 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
0150 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35
0160 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45

Wi-Fi: en0: <live capture in progress>

Packets: 71 · Displayed: 4 (6.6%) · Profile: Default

Рисунок 4 — ping с размером пакета 2000

```

% ping -s 4000 -c 1 fstec.ru
PING fstec.ru (95.173.157.32): 4000 data bytes
4008 bytes from 95.173.157.32: icmp_seq=0 ttl=50 time=16.084 ms
--- fstec.ru ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 16.084/16.084/16.084/nan ms

```

Capturing from Wi-Fi: en0

No.	Time	Source	Destination	Protocol	Length	Info
52	5.049847	192.168.0.208	95.173.157.32	IPv4	1514	Fragmented IP protocol (proto=)
53	5.049907	192.168.0.208	95.173.157.32	IPv4	1514	Fragmented IP protocol (proto=)
54	5.049918	192.168.0.208	95.173.157.32	ICMP	1082	Echo (ping) request id=0x47b9
55	5.064615	95.173.157.32	192.168.0.208	IPv4	1514	Fragmented IP protocol (proto=)
56	5.064768	95.173.157.32	192.168.0.208	IPv4	1514	Fragmented IP protocol (proto=)
57	5.064770	95.173.157.32	192.168.0.208	ICMP	1082	Echo (ping) reply id=0x47b9

> Frame 52: Packet, 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0 at 00:00:00:00:00:00 [eth0] (Ethernet II, Src: Apple\_f1:2c:65 (bc:00:74:1f:2c:65), Dst: 95.173.157.32 [95:17:3:15:7:32]) on wire (12112 bits)
> Ethernet II, Src: Apple\_f1:2c:65 (bc:00:74:1f:2c:65), Dst: 95.173.157.32 (95:17:3:15:7:32)
> Internet Protocol Version 4, Src: 192.168.0.208, Dst: 95.173.157.32 (95:17:3:15:7:32)
> Data (1480 bytes)

0000 e4 fa c4 4a e7 42 bc d0 74 f1 2c 65 08 00 45 00
0010 05 f0 16 00 00 00 00 00 00 00 00 00 00 00 00 ad
0020 9d 28 08 00 52 2d 2e b9 00 00 69 1c 66 23 00 0b
0030 6f b6 08 00 00 00 00 0c 0d 0e 0f 10 11 12 13 14 15
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45
0070 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55
0080 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64 65
0090 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75
00a0 70 71 78 79 7a 7b 7c 7d 7e 7f 80 81 82 83 84 85
00b0 80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f
00c0 88 89 8a 8b 8c 8d 8e 8f 80 81 82 83 84 85
00d0 86 87 98 99 9a 9b 9c 9d 9e 9f 80 81 82 83 84 85
00e0 a6 a7 a8 a9 aa ab ac ad ae af b0 b1 b2 b3 b4 b5
00f0 b6 b7 b8 b9 ba bb bc bd be bf c0 c1 c2 c3 c4 c5
0100 c6 c7 c8 c9 ca cb cc cd ce cf d0 d1 d2 d3 d4 d5
0110 d6 d7 d8 d9 da db dc dd de df e0 e1 e2 e3 e4 e5
0120 e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 e0 e1 e2 e3 e4 e5
0130 f6 f7 f8 f9 f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 f0 f1
0140 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
0150 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35
0160 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45

Wi-Fi: en0: <live capture in progress>

Packets: 63 · Displayed: 6 (9.5%) · Profile: Default

Рисунок 5 — ping с размером пакета 4000

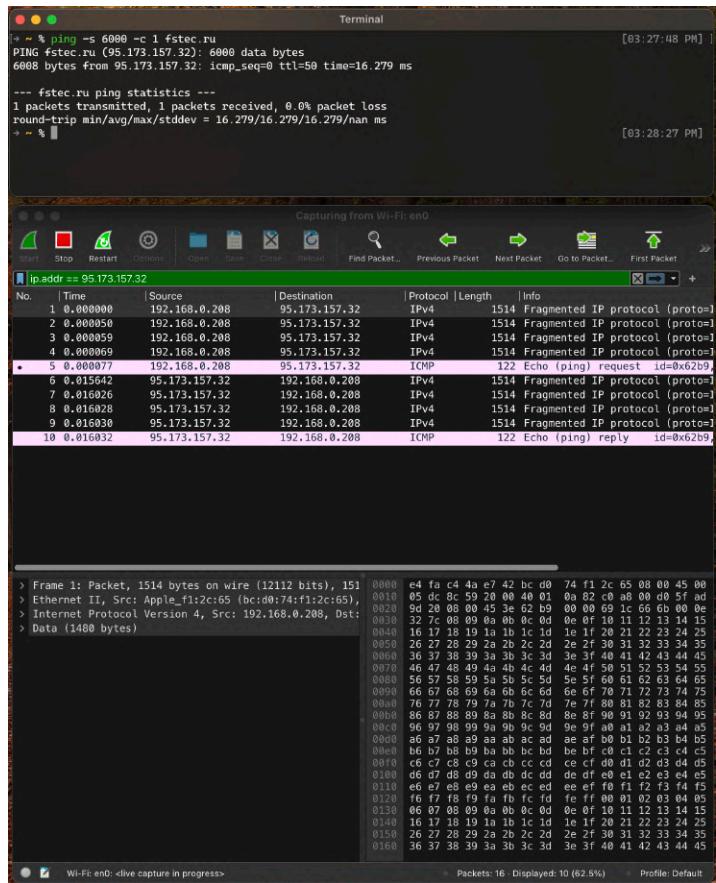


Рисунок 6 — ping с размером пакета 6000

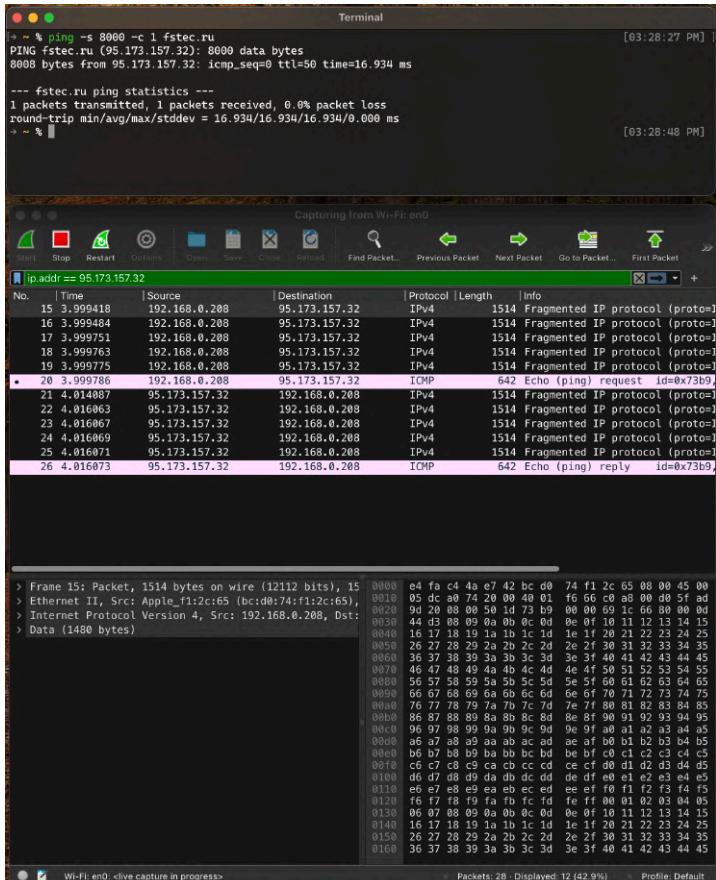


Рисунок 7 — ping с размером пакета 8000

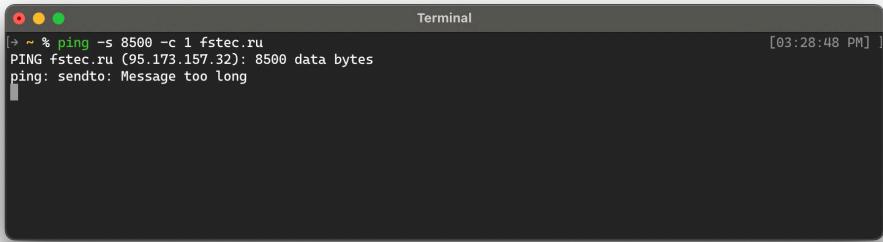


Рисунок 8 — ping с размером пакета 8500, выдающий ошибку

Структура перехваченных PDU (Protocol Data Unit):

1. Канальный уровень: Ethernet Frame с MAC-адресами;
2. Сетевой уровень: IP-пакет с IP-адресами;
3. Транспортный уровень: протокол ICMP;
4. Прикладной уровень: специфичные сообщения (ICMP-запросы).

При фрагментации каждый фрагмент — самостоятельный IP-пакет с данными транспортного уровня.

1. Имеет ли место фрагментация исходного пакета, какое поле на это указывает? — Да, в случае если размер пакета превышает MTU. MTU (Maximum Transmission Unit) — максимальный размер блока данных, который может быть передан за один раз без фрагментации. Фрагментация видна по флагу “More fragments” и полю Fragment offset в заголовке IP.

2. Какая информация указывает, является ли фрагмент пакета последним или промежуточным? — Когда бит "MF" установлен в 1, это свидетельствует о том, что текущий фрагмент является промежуточным, и в последующих фрагментах данного IP-пакета будут содержаться дополнительные фрагменты данных. Установка бита "MF" в 0 указывает на то, что данный фрагмент представляет собой завершающий фрагмент в последовательности фрагментов данного IP-пакета. Если бит "DF" установлен в 1, это означает, что IP-пакет не может быть фрагментирован в процессе передачи по сети. Если бит "DF" равен 0, то IP-пакет может быть фрагментирован, если это необходимо в процессе передачи.

```

> Frame 4: Packet, 1514 bytes on wire (12112 bits), 151 0010 05 dc f6 b9 20 b9 40 01 9f 68 c0 a8 00 d0 5f ad
> Ethernet II, Src: Apple_f1:2c:65 (bc:0:74:f1:2c:65), Dst: 0020 9d 20 c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd
< Internet Protocol Version 4, Src: 192.168.0.208, Dst: 0030 ce cf d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd
    0100 ... = Version: 4 0040 de df e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed
    .... 0101 = Header Length: 20 bytes (5) 0050 ee ef f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECI 0060 fe ff 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d
    Total Length: 1500 0070 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d
    Identification: 0xf6b9 (63161) 0080 1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d
    < 001. .... = Flags: 0x1, More fragments 0090 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d
    0... .... = Reserved bit: Not set 00a0 3e 40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d
    .0. .... = Reserved fragment: Not set 00b0 4e 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d
    ...1. .... = More fragments: Set 00c0 5e 5f 60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d
    ...0 0000 1011 1001 = Fragment Offset: 1480 00d0 7e 7f 80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d
    Time to Live: 64 00e0 8e 8f 90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d
    Protocol: ICMP (1) 00f0 9e 9f a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad
    Header Checksum: 0x9f68 [Validation disabled] 0100 ae af bb b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd
    [Header checksum status: Unverified] 0110 be bf c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd
    Source Address: 192.168.0.208 0120 ce cf d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd
    Destination Address: 95.173.157.32 0130 ef e0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd
    [Reassembled IPv4 in frame: 5] 0140 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d
    Packets: 1718 - Displayed: 6 (0.3%) Profile: Default

```

Рисунок 9 — MF=1 для второго фрагмента из трех

3. Чему равно количество фрагментов при передаче ping-пакетов? —

В случае, если размер пакета превышает MTU и DF не равен 1, пакет будет поделен на фрагменты равные размеру установленного MTU (в данном случае 1500).

4. Построить график, в котором на оси абсцисс находится размер\_пакета, а по оси ординат – количество фрагментов, на которое был разделён каждый ping-пакет. — На рисунке 10 изображен данный график.

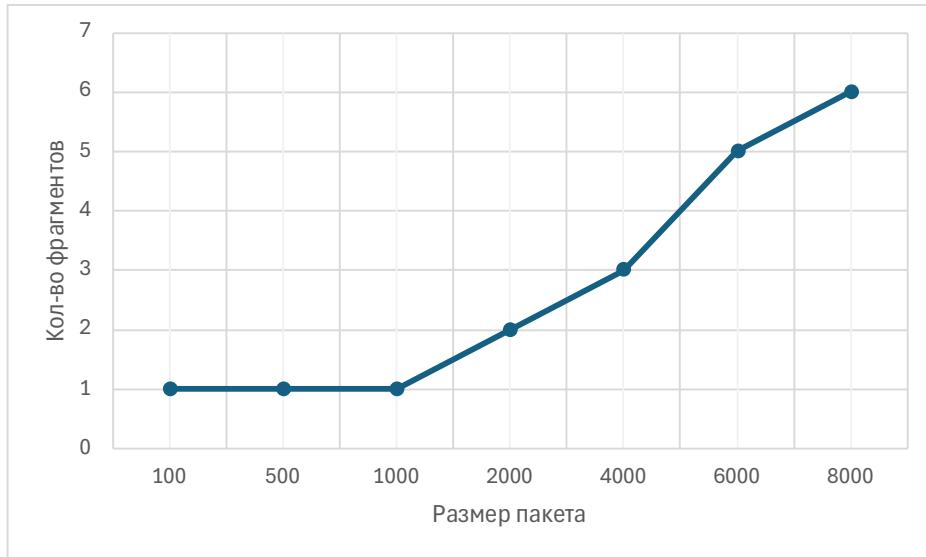


Рисунок 10 — График зависимости количества фрагментов от размера пакета

5. Как изменить поле TTL с помощью утилиты ping? — Нужно добавить аргумент -m (на macOS), значением которого будет TTL в миллисекундах.

6. Что содержится в поле данных ping-пакета? — Содержимое поля данных в ping-пакете представляет собой последовательность букв английского алфавита, которая повторяется до достижения заданного

размера пакета (на macOS).

### 3.2 Анализ трафика утилиты traceroute (traceroute)

Был отслежен и проанализирован трафик, создаваемый утилитой traceroute, при её запуске из терминала.

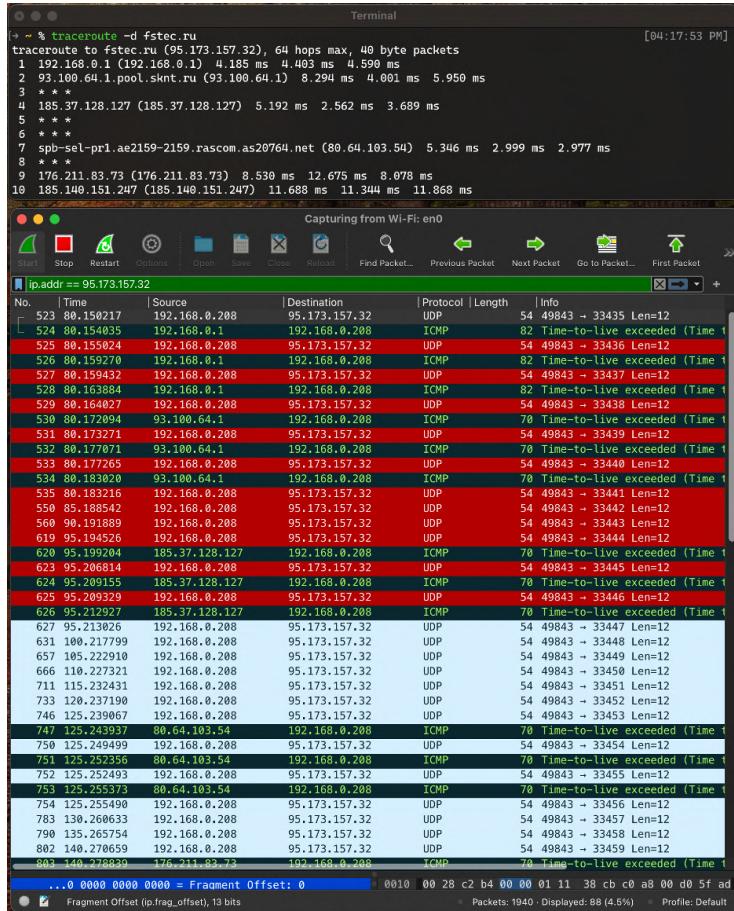


Рисунок 11 — traceroute -d fstec.ru

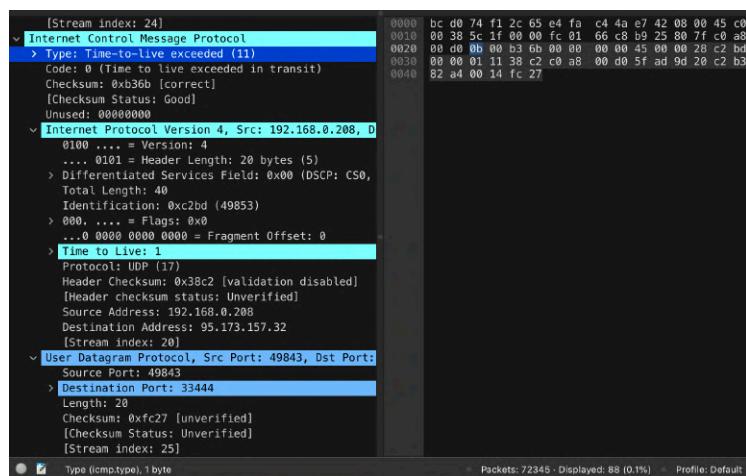


Рисунок 12 — Структура PDU сообщения “Time-to-live exceeded”

## Структура перехваченных PDU:

В ICMP-сообщении "Time-to-live exceeded" структура уровней 2–3 аналогична обычным ICMP-пакетам. На транспортном уровне содержится ICMP с данным сообщением об ошибке, которое включает вложенный IP-заголовок исходного пакета. Это позволяет идентифицировать, на каком маршрутизаторе истёк TTL и какой пакет был отправлен.

1. Сколько байт содержится в заголовке IP? Сколько байт содержится в поле данных? — Заголовок IP (Internet Protocol) имеет фиксированный размер и состоит из 20 байт (160 бит) для версии IPv4. Поле данных имеет переменную длину, но для ICMP обычно содержит 32 байта.

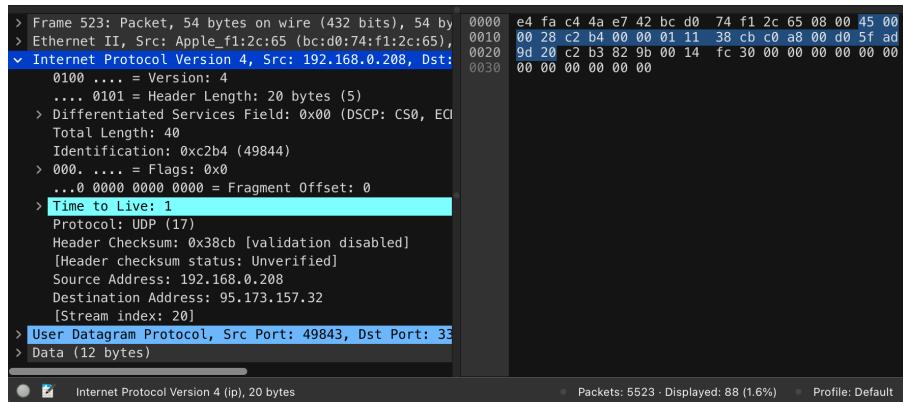


Рисунок 13 — Содержание пакета. Выделен заголовок IP

2. Как и почему изменяется поле TTL в следующих друг за другом ICMP-пакетах tracert? Для ответа на этот вопрос нужно проследить изменение TTL при передаче по маршруту, состоящему из более чем двух хопов. — Поле TTL изменяется для того, чтобы пакеты могли "прыгнуть" от одного маршрутизатора к другому, и изменение TTL позволяет определить, сколько хопов включено в маршрут между исходным компьютером и конечным пунктом назначения. На рисунке 9 отображена эта логика на перехваченном пакете.

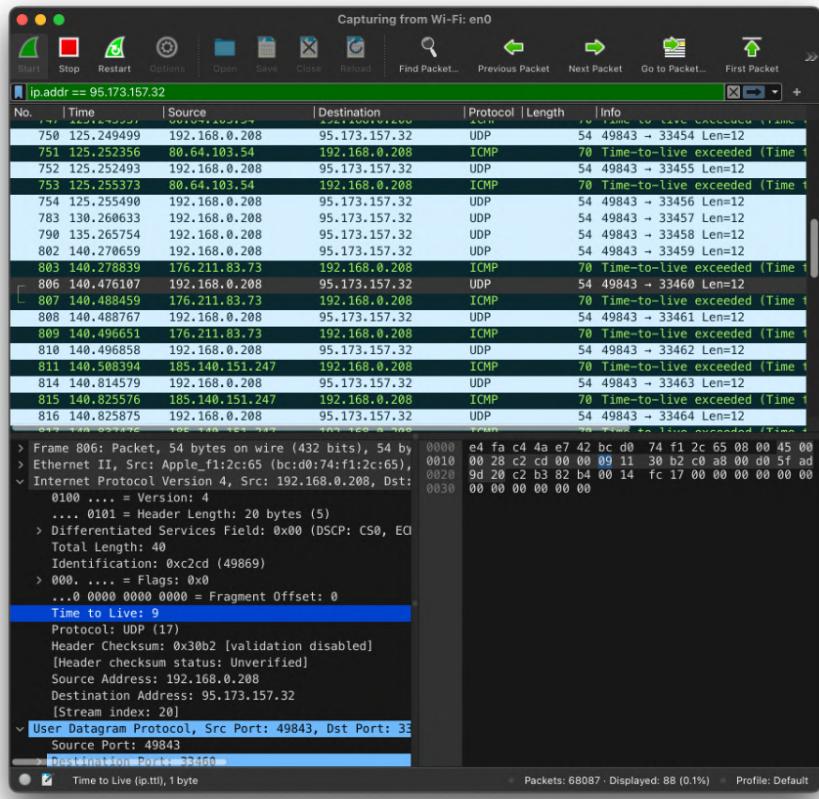


Рисунок 14 — TTL равен 9 для исследования 9-го хопа

3. Чем отличаются ICMP-пакеты, генерируемые утилитой tracert, от ICMP-пакетов, генерируемых утилитой ping (см. предыдущее задание). — Утилита ping использует ICMP Echo Request и Echo Reply для проверки доступности узла и измерения времени отклика. В отличие от этого, tracert отправляет пакеты с постепенно увеличивающимся TTL, чтобы получить от каждого промежуточного маршрутизатора ICMP-сообщение "Time Exceeded". Таким образом, ping работает по принципу запрос-ответ для проверки связи, а tracert использует сообщения об ошибках для построения маршрута следования пакетов.

4. Чем отличаются полученные пакеты «ICMP reply» от «ICMP error» и зачем нужны оба этих типа ответов? — Пакеты ICMP Reply являются нормальными ответами на корректные запросы (например, Echo Reply на Echo Request в ping), подтверждая доступность узла. ICMP Error сообщения генерируются при проблемах обработки пакета (недоступность порта, превышение TTL) и служат для диагностики сетевых неисправностей.

Таким образом, ICMP Reply обеспечивает проверку связности, а ICMP Error — диагностику ошибок передачи данных, что в совокупности позволяет эффективно анализировать работу сети.

5. Что изменится в работе tracert, если убрать ключ “-d”? Какой дополнительный трафик при этом будет генерироваться? — Если убрать ключ «-d», то traceroute будет пытаться разрешать имена хостов для промежуточных узлов в пути маршрутизации, что может добавить лишний сетевой трафик и замедлить выполнение команды.

### 3.3 Анализ HTTP-трафика

Был отслежен и проанализирован HTTP-трафик, создаваемый браузером при посещении Интернет-сайта <http://185.202.207.59/> (личный сайт). На рисунках 15–19 представлены полученные данные в результате анализа трафика.

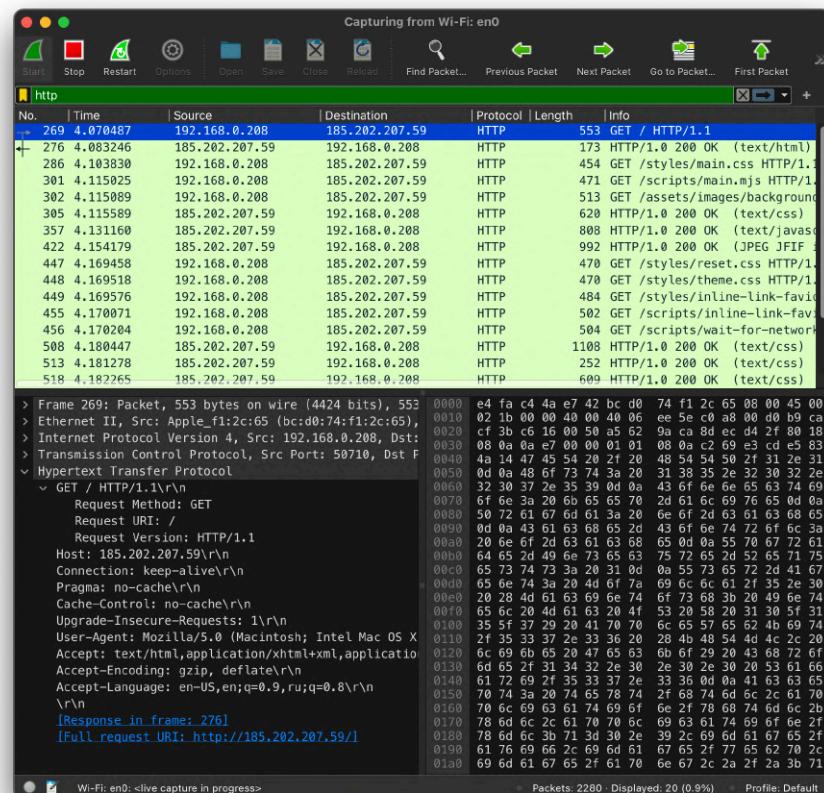


Рисунок 15 — Перехваченные GET-запросы и ответы на них

```

> Frame 269: Packet, 553 bytes on wire (4424 bits), 553
> Ethernet II, Src: Apple_f1:2c:65 (bc:d0:74:f1:2c:65),
> Internet Protocol Version 4, Src: 192.168.0.208, Dst:
> Transmission Control Protocol, Src Port: 50710, Dst F
  Hypertext Transfer Protocol
    GET / HTTP/1.1\r\n
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
      Host: 185.202.207.59\r\n
      Connection: keep-alive\r\n
      Pragma: no-cache\r\n
      Cache-Control: no-cache\r\n
      Upgrade-Insecure-Requests: 1\r\n
      User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
      Accept: text/html,application/xhtml+xml,application/
      Accept-Encoding: gzip, deflate\r\n
      Accept-Language: en-US,en;q=0.9,ru;q=0.8\r\n
      \r\n
      [Response in frame: 276]
      [Full request URI: http://185.202.207.59/]

0040 4a 14 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31
0050 0d 0a 48 6f 73 74 3a 20 31 38 35 2e 32 30 32 2e
0060 32 30 37 2e 35 39 0d 0a 43 6f 6e 65 63 74 69
0070 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a
0080 50 72 61 67 6d 61 3a 20 6e 6f 2d 63 61 63 68 65
0090 0d 0a 43 61 63 68 65 2d 43 6f 6e 74 72 6f 6c 3a
00a0 20 6e 6f 2d 63 63 68 65 0d 0a 55 70 67 72 61
00b0 64 65 2d 49 6e 73 65 63 72 65 2d 52 65 71 75
00c0 65 73 74 73 3a 20 31 0d 0a 55 73 65 72 2d 41 67
00d0 65 6e 74 3a 20 4d 6f 7a 69 6c 61 2f 35 2e 30
00e0 20 28 4d 61 63 69 6e 74 6f 73 68 3b 20 49 6e 74
00f0 65 6c 20 4d 61 63 20 4f 53 20 58 20 31 39 5f 31
0100 35 5f 37 29 20 41 70 70 6c 65 57 65 62 4b 69 74
0110 2f 35 33 37 2e 33 36 20 28 4b 48 54 4d 4c 2c 20
0120 61 69 66 20 47 65 63 6b 6f 29 20 43 68 72 6f
0130 6d 65 2f 31 34 2e 3e 30 2e 30 2e 30 20 53 61 66
0140 61 72 69 2f 35 33 37 2e 33 36 0d 0a 41 63 63 65
0150 70 74 3a 20 74 65 78 74 2f 68 74 6d 6c 2c 61 70
0160 70 6c 69 63 61 74 69 6f 6e 2f 78 68 74 6d 6c 2b
0170 78 6d 6c 61 70 70 6c 69 63 61 74 69 6f 6e 2f
0180 78 6d 6c 3b 71 3d 30 2e 39 2c 69 6d 61 67 65 2f
0190 61 76 69 66 2c 69 6d 61 67 65 2f 77 65 62 70 2c
01a0 69 6d 61 67 65 2f 61 70 6e 67 2c 2a 2f 2a 3b 71
01b0 3d 30 2e 38 2c 6f 70 70 6c 69 63 61 74 69 6f 6e
01c0 2f 73 69 67 6e 65 64 2d 65 78 63 68 61 6e 67 65
01d0 3b 76 3d 62 33 3b 71 3d 30 2e 37 0d 0a 41 63 63
01e0 65 70 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a

```

Пакеты: 2476 · Displayed: 20 (0.8%) · Profile: Default

Рисунок 16 — Заголовки GET-запроса

```

> Internet Protocol Version 4, Src: 185.202.207.39, Dst:
> Transmission Control Protocol, Src Port: 80, Dst Port:
> [5 Reassembled TCP Segments (4639 bytes): #271(188), #272(188), #273(188), #274(188), #275(188)]
  Hypertext Transfer Protocol
    HTTP/1.0 200 OK\r\n
      Response Version: HTTP/1.0
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Server: SimpleHTTP/0.6 Python/3.10.12\r\n
      Date: Tue, 18 Nov 2025 14:22:01 GMT\r\n
      Content-type: text/html\r\n
> Content-Length: 4451\r\n
Last-Modified: Wed, 05 Nov 2025 00:16:02 GMT\r\n
\r\n
      [Request in frame: 269]
      [Time since request: 12.759000 milliseconds]
      [Request URI: /]
      [Full request URI: http://185.202.207.59/]
      File Data: 4451 bytes
      Line-based text data: text/html (123 lines)
        <!DOCTYPE html>\r\n
        <html lang="en" class="sl-theme-dark">\n

```

Packet (173 bytes)	Reassembled TCP (4639 bytes)
--------------------	------------------------------

Пакеты: 2571 · Displayed: 20 (0.8%) · Profile: Default

Рисунок 17 — Заголовки HTTP-ответа с кодом 200

```

> Frame 3170: Packet, 586 bytes on wire (4688 bits), 586
> Ethernet II, Src: Apple_f1:2c:65 (bc:d0:74:f1:2c:65),
> Internet Protocol Version 4, Src: 192.168.0.208, Dst:
> Transmission Control Protocol, Src Port: 50764, Dst F
  Hypertext Transfer Protocol
    GET / HTTP/1.1\r\n
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
      Host: 185.202.207.59\r\n
      Connection: keep-alive\r\n
      Cache-Control: max-age=0\r\n
      Upgrade-Insecure-Requests: 1\r\n
      User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
      Accept: text/html,application/xhtml+xml,application/
      Accept-Encoding: gzip, deflate\r\n
      Accept-Language: en-US,en;q=0.9,ru;q=0.8\r\n
      If-Modified-Since: Wed, 05 Nov 2025 00:16:02 GMT\r\n
      \r\n
      [Response in frame: 3179]
      [Full request URI: http://185.202.207.59/]

0040 4d ea 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31
0050 0d 0a 48 6f 73 74 3a 20 31 38 35 2e 32 30 32 2e
0060 32 30 37 2e 35 39 0d 0a 43 6f 6e 65 63 74 69
0070 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a
0080 43 61 63 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6d
0090 61 78 2d 61 67 65 3d 30 0d 0a 55 70 67 72 61 64
00a0 65 2d 49 6e 73 65 63 75 72 65 2d 52 65 71 75 65
00b0 73 74 73 3a 20 31 0d 0a 55 73 65 72 2d 41 67 65
00c0 6e 74 3a 20 4d 6f 7a 66 6c 61 2f 35 2e 30 20
00d0 28 4d 61 63 69 6e 74 6f 73 68 3b 20 49 6e 74 65
00e0 6c 20 4d 61 63 20 4f 53 20 58 20 31 39 5f 31 35
00f0 5f 37 29 20 41 70 70 6c 65 57 65 62 4b 69 74 2f
0100 35 33 37 2e 33 36 20 28 4b 48 54 4d 4c 2c 20 6c
0110 69 6b 65 20 47 65 63 6b 6f 29 20 43 68 72 6f 6d
0120 65 2f 31 34 32 2e 30 2e 30 2e 30 20 53 61 66 61
0130 72 69 2f 35 33 37 2e 33 36 0d 0a 41 63 63 65 70
0140 74 3a 20 74 65 78 74 2f 68 74 6d 6c 2c 61 70 70
0150 6c 69 63 61 74 69 6f 6e 2f 78 68 74 6d 6c 2b 78
0160 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78
0170 6d 6c 3b 71 3d 30 2e 39 2c 69 6d 61 67 65 2f 61
0180 76 69 66 2c 69 6d 61 67 65 2f 77 65 62 70 2c 69
0190 6d 67 65 2f 61 70 6e 67 2c 2a 2f 2a 3b 71 3d
01a0 30 2e 38 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f
01b0 73 69 67 6e 65 64 2d 65 78 63 68 61 6e 67 65 3b
01c0 76 3d 62 33 3b 71 3d 30 2e 37 0d 0a 41 63 63 65
01d0 70 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a 69
01e0 70 2c 20 64 65 66 6c 61 74 65 0d 0a 41 63 63 65

```

Пакеты: 4481 · Displayed: 37 (0.8%) · Profile: Default

Рисунок 18 — Заголовки условного GET-запроса

The screenshot shows a Wireshark capture of an Ethernet frame (Frame 3179) containing an HTTP response. The packet details pane shows the following:

```

> Frame 3179: Packet, 171 bytes on wire (1368 bits), 17
> Ethernet II, Src: TPLink_4ae7:42 (e4:fa:c4:e7:42)
> Internet Protocol Version 4, Src: 185.202.207.59, Dst
> Transmission Control Protocol, Src Port: 80, Dst Port
  Hypertext Transfer Protocol
    HTTP/1.0 304 Not Modified\r\n
      Response Version: HTTP/1.0
      Status Code: 304
      [Status Code Description: Not Modified]
      Response Phrase: Not Modified
      Server: SimpleHTTP/0.6 Python/3.10.12\r\n
      Date: Tue, 18 Nov 2025 14:24:03 GMT\r\n
      \r\n
      [Request in frame: 3170]
      [Time since request: 13.263000 milliseconds]
      [Request URI: /]
      [Full request URI: http://185.202.207.59/]

```

The packet bytes pane shows the raw hex and ASCII data of the response.

Рисунок 19 — Заголовки HTTP-ответа с кодом 304 (Not Modified)

Структура перехваченных PDU:

1. Канальный уровень: Ethernet Frame с MAC-адресами;
2. Сетевой уровень: IP-пакет с IP-адресами;
3. Транспортный уровень: TCP-сегменты с портами;
4. Прикладной уровень: HTTP-запросы/ответы с заголовками и данными.

Порядок действий при первичном посещении:

1. Установка TCP-соединения;
2. HTTP GET-запрос с заголовками;
3. Сервер формирует ответ (статус 200 (OK) и данные HTML);
4. Браузер отображает страницу.

При вторичном посещении порядок аналогичен первичному, но сервер может вернуть статус 304 (Not Modified) для использования кеша, если контент не изменился.

### 3.4 Анализ DNS-трафика

Был отслежен и проанализирован трафик протокола DNS, сгенерированный в результате посещения сайта fstec.ru. На рисунке 20 изображены перехваченные DNS-запросы и ответы.

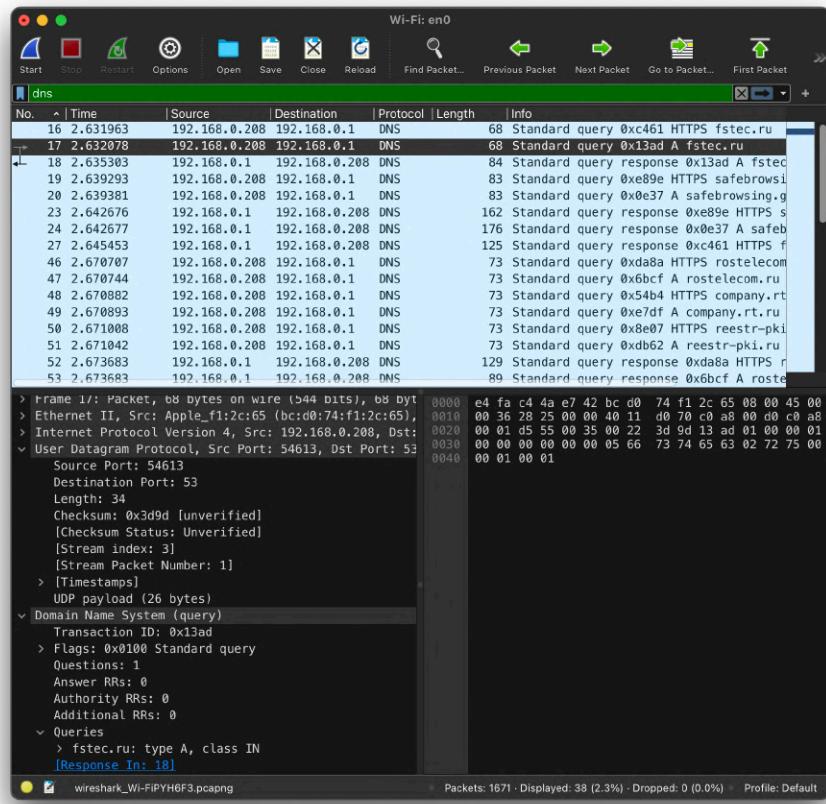


Рисунок 20 — Трафик DNS при заходе на сайт fstec.ru

Структура перехваченных PDU. DNS-запрос:

1. Канальный уровень: Ethernet Frame с MAC-адресами;
2. Сетевой уровень: IP-пакет с адресами отправителя и получателя;
3. Транспортный уровень: UDP-сегменты с портами;
4. Прикладной уровень: DNS-запрос с идентификатором, доменным именем и типом записи (A, AAAA и т.д.).

DNS-ответ имеет аналогичную структуру, но на прикладном уровне содержит запрошенные DNS-записи (IP-адреса или другие данные).

1. Почему адрес, на который отправлен DNS-запрос, не совпадает с адресом посещаемого сайта? — В таких случаях, как правило, имеют место распределенные DNS-системы, балансировка нагрузки, скрытие инфраструктуры.

2. Какие бывают типы DNS-запросов? — A (IPv4-адрес домена), AAAA (IPv6-адрес), MX (почтовые серверы домена), TXT (текстовая информация), NS (DNS-серверы зоны), PTR (обратное преобразование —

IP-адрес в доменное имя), SRV (нахождение сетевых сервисов).

3. В какой ситуации нужно выполнять независимые DNS-запросы для получения содержащихся на сайте изображений? — В случае, если изображения на сайте хранятся по другому адресу (например, CDN-система — Content Delivery Network, работающая на другой инфраструктуре).

### 3.5 Анализ ARP-трафика

Был отслежен и проанализирован трафик протокола ARP, сгенерированный в результате посещения сайта fstec.ru. На рисунке 21 изображены перехваченные ARP-пакеты.

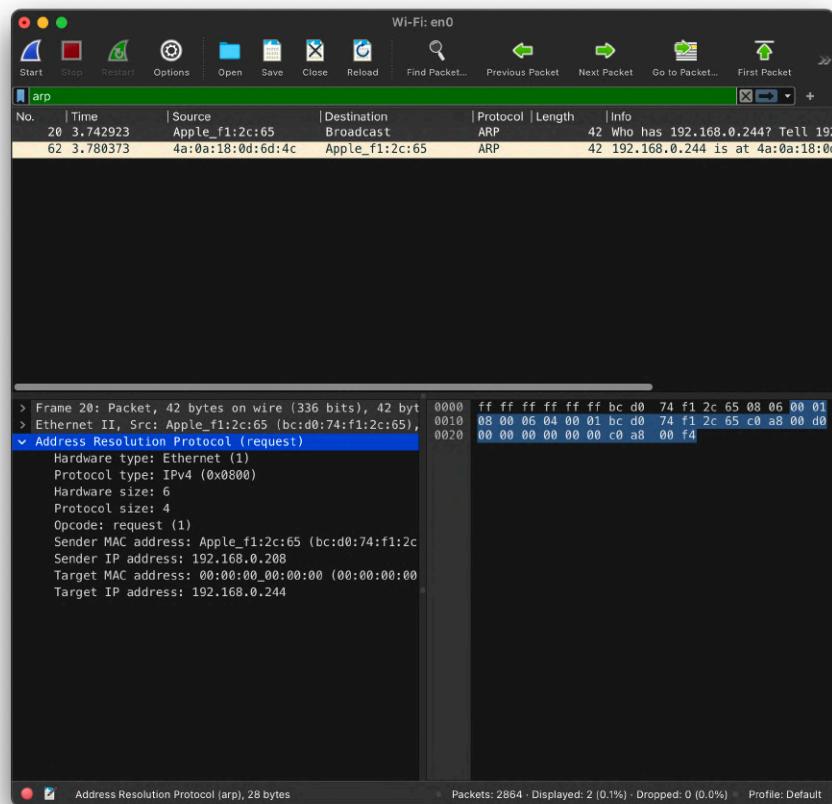


Рисунок 21 — Трафик ARP при заходе на сайт fstec.ru

Структура перехваченных PDU:

1. Канальный уровень: Ethernet Frame с MAC-адресами;
2. Сетевой уровень: информация о типе аппаратного обеспечения, типе протокола, MAC и IP-адреса отправителя и получателя, тип пакета (request 1 или reply 2);

1. Какие MAC-адреса присутствуют в захваченных пакетах ARP-протокола? Что означают эти адреса? Какие устройства они идентифицируют? — Sender MAC Address (MAC-адрес отправителя) — это устройство, которое выполнило ARP-запрос или ответ. Это может быть компьютер, маршрутизатор, сервер или любое другое сетевое устройство, которое участвует в процессе разрешения IP-адресов в MAC-адреса. Target MAC Address (MAC-адрес получателя) — в случае ARP-запроса это широковещательный адрес, что означает, что запрос отправляется всем устройствам в сети; а в случае ARP-ответа — это MAC-адрес устройства, которое запрашивало соответствующий IP-адрес.

2. Какие MAC-адреса присутствуют в захваченных HTTP-пакетах и что означают эти адреса? Какие устройства они идентифицируют? — В HTTP-пакетах присутствуют MAC-адреса, которые выполняют функцию идентификации отправителя и получателя данных пакетов. В данном контексте на обеих сторонах обмена можно наблюдать MAC-адреса маршрутизатора внутри локальной сети и MAC-адрес компьютера. Эти MAC-адреса указывают, от каких сетевых интерфейсов идут данные HTTP-пакеты и на какие сетевые устройства они направлены.

3. Для чего ARP-запрос содержит IP-адрес источника? — IP-адрес источника в ARP-запросе используется для определения MAC-адреса устройства, связанного с этим IP-адресом в локальной сети.

#### 4 Выводы

В ходе выполнения лабораторной работы была достигнута цель по изучению структуры протокольных блоков данных через анализ реального сетевого трафика с использованием Wireshark. На практике исследованы ключевые протоколы сетевого взаимодействия:

1. ICMP (ping/traceroute). Установлены различия в механизмах работы утилит ping (использующей запросы Echo Request/Reply) и traceroute (основанной на анализе ICMP-ошибок "Time Exceeded"). Определены условия фрагментации пакетов и зависимость количества фрагментов

от размера пакета.

2. HTTP. Проанализирована структура HTTP-запросов и ответов, включая процесс установки TCP-соединения и особенности кеширования при повторных обращениях к ресурсу.
3. DNS. Изучены типы DNS-запросов (A, AAAA, MX и т. д.) и их роль в преобразовании доменных имен в IP-адреса. Продемонстрирована важность DNS-кэширования для оптимизации трафика.
4. ARP. Исследован механизм разрешения IP-адресов в MAC-адреса в локальной сети, а также роль широковещательных запросов в этом процессе.

Работа подтвердила, что анализ сетевого трафика позволяет наглядно изучать архитектуру сетевых протоколов, диагностировать проблемы и оптимизировать сетевое взаимодействие. Полученные навыки использования Wireshark являются основой для дальнейшего освоения методик мониторинга и обеспечения безопасности компьютерных сетей.