

# Лабораторная работа №3

## Сетевые технологии

---

Иванов Сергей Владимирович, НПИбд-01-23

4 октября 2025

Российский университет дружбы народов, Москва, Россия

## MAC-адресация

С помощью команды ipconfig для ОС типа Windows выведем информацию о текущем сетевом соединении. (рис. 1).

# MAC-адресация

1. Адаптер Ethernet Ethernet 3. (VirtualBox)
2. Адаптер беспроводной локальной сети Подключение по локальной сети\*  
9. Звёздочка указывает на виртуальный или вспомогательный адаптер.  
Состояние среды: Среда передачи недоступна. Этот адаптер неактивен.
3. Адаптер беспроводной локальной сети Подключение по локальной сети\*  
10. Аналогичен предыдущему. Этот адаптер тоже неактивен.
4. Адаптер Ethernet outline-tap0. Название говорит, что это виртуальный адаптер, созданный VPN (Outline VPN). Состояние среды: Среда передачи недоступна. Адаптер отключён, так как VPN выключен.

# MAC-адресация

Теперь используем опцию /all команды ipconfig. Команда предоставляет расширенную информацию о конфигурации сетевых адаптеров.

```
Адаптер беспроводной локальной сети Беспроводная сеть:

DNS-суффикс подключения . . . . . : 
Описание. . . . . : MediaTek Wi-Fi 6 MT7921 Wireless LAN Card
Физический адрес. . . . . : 90-E8-68-FA-55-E3
DHCP включен. . . . . : Да
Автонастройка включена. . . . . : Да
Локальный IPv6-адрес канала . . . . . : fe80::2fef:1cd:c5e8:217a%14(Основной)
IPv4-адрес. . . . . : 172.16.94.216(Основной)
Маска подсети . . . . . : 255.255.254.0
Аренда получена. . . . . : 29 сентября 2025 г. 10:49:21
Срок аренды истекает. . . . . : 29 сентября 2025 г. 15:49:10
Основной шлюз. . . . . : 172.16.94.1
DHCP-сервер. . . . . : 192.168.80.59
IAID DHCPv6 . . . . . : 496035944
DUID клиента DHCPv6 . . . . . : 00-01-00-01-2D-00-EA-E7-58-11-22-88-F5-69
DNS-серверы. . . . . : 37.18.92.5
                                         193.232.218.194
NetBIOS через TCP/IP. . . . . : Включен
```

```
Адаптер Ethernet Сетевое подключение Bluetooth:
```

```
Состояние среды. . . . . : Среда передачи недоступна.
DNS-суффикс подключения . . . . . : 
Описание. . . . . : Bluetooth Device (Personal Area Network)
Физический адрес. . . . . : 90-E8-68-FA-55-E2
DHCP включен. . . . . : Да
Автонастройка включена. . . . . : Да
```

## MAC-адресация

**Рис. 3:** Команда ipconfig /all

## MAC-адресация

DNS-суффикс подключения: (пусто). Это может быть связано с динамической настройкой через DHCP.

Описание: MediaTek Wi-Fi 6 MT7921 Wireless LAN Card.

Физический адрес: 90-E8-68-FA-55-E3.

DHCP включён: Да. DHCP включён, IP-адрес и другие параметры сети автоматически выдаются сервером.

Локальный IPv6-адрес канала: fe80::2fef:1cdc:5e8:217a%14. используется для коммуникации внутри локального сегмента сети без маршрутизации.

IPv4-адрес: 172.16.4.1.

## MAC-адресация

Маска подсети: 255.255.255.0. Маска подсети /24 определяет, что подсеть содержит до 254 хостов.

Аренда получена: 29 сентября 2025 г. 10:49:21. Срок аренды истекает: 29 сентября 2025 г. 15:49:10.

Основной шлюх: 172.16.4.59. Шлюз, через который трафик направляется за пределы локальной сети.

DHCP-сервер: 192.168.80.59. IP-адрес сервера DHCP, который выдал конфигурацию.

DNS-серверы: 37.18.92.5 193.232.218.194. Два DNS-сервера.

## MAC-адресация

Определим MAC-адрес интерфеса Беспроводное соединение (Wi-Fi). Он находится в поле физический адрес. Имеет длину 6 байт. Первые 3 байта – идентификатор организации, который определяет производителя оборудования. Последние 3 байта – серийный номер, назначаемый производителем.

Рис. 4: MAC-адрес

## MAC-адресация

90-E8-68 принадлежит компании AzureTechnology (MediaTek Inc.). Это совпадает с описанием устройства в выводе команды. Проанализируем адрес. Для этого нужно посмотреть на значение первого первого байта адреса в двоичном формате.

Первый байт: 90 (в шестнадцатеричной системе) = 1001 0000 (в двоичной системе).

Нас интересуют два младших бита:

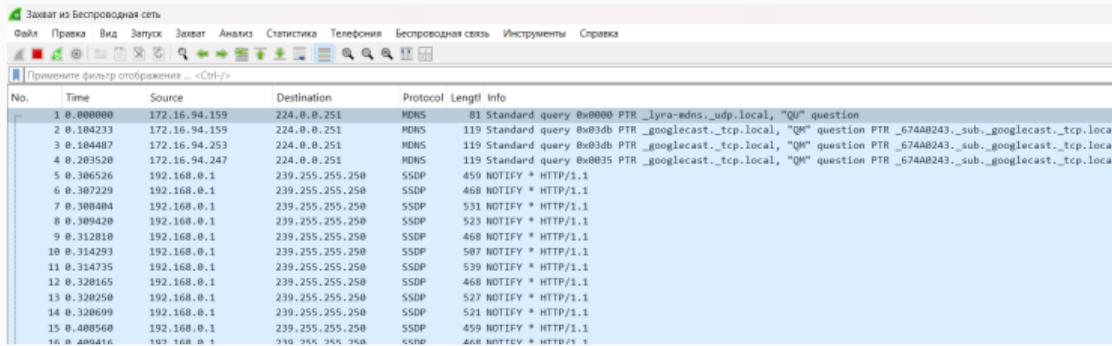
b0 (самый младший бит): Управляет типом адресации. 0 = Индивидуальный (Unicast). 1 = Групповой (Multicast).

b1 (второй бит): Управляет способом администрирования. 0 = Глобально управляемый. 1 = Локально управляемый.

Значит, наш адрес индивидуальный и глобально администрируемый.

# Анализ кадров канального уровня в Wireshark

Запустим Wireshark. Выберем активный на устройстве сетевой интерфейс и убедимся, что начался процесс захвата трафика. (рис. 5)



The screenshot shows the Wireshark interface with a list of captured network frames. The window title is "Захват из Беспроводная сеть". The menu bar includes: Файл, Прекрати, Вид, Запуск, Захват, Анализ, Статистика, Телефония, Беспроводная связь, Инструменты, Справка. A search bar at the top says "Примените фильтр отображения ...<Ctrl-/>". The main pane displays 16 rows of captured data, each with columns: No., Time, Source, Destination, Protocol, Length, and Info. The "Info" column contains detailed descriptions of the traffic, such as "Standard query 0x0000 PTR \_lyra-mdns.\_udp.local, \"Q\" question" and "NOTIFY \* HTTP/1.1".

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.94.159	224.0.0.251	MDNS	81	Standard query 0x0000 PTR _lyra-mdns._udp.local, "Q" question
2	0.104233	172.16.94.159	224.0.0.251	MDNS	119	Standard query 0x03db PTR _googlecast._tcp.local, "Q" question PTR _674AB243._sub._googlecast._tcp.local,
3	0.104487	172.16.94.253	224.0.0.251	MDNS	119	Standard query 0x03db PTR _googlecast._tcp.local, "Q" question PTR _674AB243._sub._googlecast._tcp.local,
4	0.203528	172.16.94.247	224.0.0.251	MDNS	119	Standard query 0x03db PTR _googlecast._tcp.local, "Q" question PTR _674AB243._sub._googlecast._tcp.local,
5	0.306526	192.168.0.1	239.255.255.250	SSDP	459	NOTIFY * HTTP/1.1
6	0.307229	192.168.0.1	239.255.255.250	SSDP	468	NOTIFY * HTTP/1.1
7	0.308404	192.168.0.1	239.255.255.250	SSDP	531	NOTIFY * HTTP/1.1
8	0.309428	192.168.0.1	239.255.255.250	SSDP	523	NOTIFY * HTTP/1.1
9	0.312818	192.168.0.1	239.255.255.250	SSDP	468	NOTIFY * HTTP/1.1
10	0.314293	192.168.0.1	239.255.255.250	SSDP	507	NOTIFY * HTTP/1.1
11	0.314735	192.168.0.1	239.255.255.250	SSDP	539	NOTIFY * HTTP/1.1
12	0.320165	192.168.0.1	239.255.255.250	SSDP	468	NOTIFY * HTTP/1.1
13	0.320258	192.168.0.1	239.255.255.250	SSDP	527	NOTIFY * HTTP/1.1
14	0.320699	192.168.0.1	239.255.255.250	SSDP	521	NOTIFY * HTTP/1.1
15	0.408568	192.168.0.1	239.255.255.250	SSDP	459	NOTIFY * HTTP/1.1
16	0.409416	192.168.0.1	239.255.255.250	SSDP	458	NOTIFY * HTTP/1.1

Рис. 5: Захват трафика

## Анализ кадров канального уровня в Wireshark

На устройстве в консоли определим с помощью команды ipconfig IP-адрес устройства и шлюз по умолчанию. IP: 172.16.94.216, Шлюз: 172.16.94.1 (рис. 6)

**Рис. 6:** Определение IP и шлюза

# Анализ кадров канального уровня в Wireshark

В консоли с помощью команды ping адрес\_шлюза пропингуем шлюз по умолчанию. (рис. 7)

```
C:\Users\1serg>ping 172.16.94.1

Обмен пакетами с 172.16.94.1 по с 32 байтами данных:
Ответ от 172.16.94.1: число байт=32 время=3мс TTL=254
Ответ от 172.16.94.1: число байт=32 время=2мс TTL=254
Ответ от 172.16.94.1: число байт=32 время=4мс TTL=254
Ответ от 172.16.94.1: число байт=32 время=4мс TTL=254

Статистика Ping для 172.16.94.1:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
              (0% потеря)
Приблизительное время приема-передачи в мс:
    Минимальное = 2мсек, Максимальное = 4 мсек, Среднее = 3 мсек

C:\Users\1serg>
```

Рис. 7: ping шлюза

# Анализ кадров канального уровня в Wireshark

Остановим захват трафика. В строке фильтра пропишем фильтр arp or icmp. Убедимся, что в списке пакетов отобразятся только пакеты ARP или ICMP. (рис. 8)

The screenshot shows the Wireshark interface with the following details:

- Menu Bar:** Файл, Правка, Вид, Запуск, Захват, Анализ, Статистика, Телефония, Беспроводная связь, Инструменты, Справка.
- Toolbar:** Standard icons for opening files, saving, zooming, and analysis.
- Filter Bar:** arp or icmp
- List View Headers:** Список пакетов, Фильтр отображения, Введите фильтр отображения ...
- Options:** Обычные и многобайтовые, Чувствительность к регистру, Назад, Множественные случаи.
- Table Headers:** No., Time, Source, Destination, Protocol, Length, Info
- Data Rows:** A list of network packets, mostly ARP requests and ICMP echo requests, from various sources like Broadcast, Apple\_d3:3d:43, and Cisco\_60:9c:e6, directed to 172.16.94.1 or 172.16.94.216.

No.	Time	Source	Destination	Protocol	Length	Info
1767	121.550542	c2:85:6e:43:31:17	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.204
2207	127.896711	c6:8d:74:01:b0:67	Broadcast	ARP	60	Who has 172.16.94.85? Tell 172.16.94.101
2208	127.897691	Apple_d3:3d:43	Broadcast	ARP	60	Who has 172.16.94.85? Tell 172.16.94.74
2296	138.443866	e2:af:c3:73:08:00	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.87
2307	140.184151	6e:b9:26:3a:d0:69	Broadcast	ARP	60	ARP Announcement for 172.16.94.111
2308	140.286357	6e:b9:26:3a:d0:69	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.111
2323	141.720039	6e:b9:26:3a:d0:69	Broadcast	ARP	60	ARP Announcement for 172.16.94.111
2324	141.720069	6e:b9:26:3a:d0:69	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.111
2325	142.027253	6e:b9:26:3a:d0:69	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.111
2364	146.547978	172.16.94.216	172.16.94.1	ICMP	74	Echo (ping) request id=0x0001, seq=1/256, ttl=128 (reply in 2365)
2365	146.551747	172.16.94.1	172.16.94.216	ICMP	74	Echo (ping) reply id=0x0001, seq=1/256, ttl=254 (request in 2364)
2367	147.557766	172.16.94.216	172.16.94.1	ICMP	74	Echo (ping) request id=0x0001, seq=2/512, ttl=128 (reply in 2368)
2368	147.559846	172.16.94.1	172.16.94.216	ICMP	74	Echo (ping) reply id=0x0001, seq=2/512, ttl=254 (request in 2367)
2369	148.570093	172.16.94.216	172.16.94.1	ICMP	74	Echo (ping) request id=0x0001, seq=3/768, ttl=128 (reply in 2370)
2370	148.574770	172.16.94.1	172.16.94.216	ICMP	74	Echo (ping) reply id=0x0001, seq=3/768, ttl=254 (request in 2369)
2371	149.586784	172.16.94.216	172.16.94.1	ICMP	74	Echo (ping) request id=0x0001, seq=4/1024, ttl=128 (reply in 2372)
2372	149.591655	172.16.94.1	172.16.94.216	ICMP	74	Echo (ping) reply id=0x0001, seq=4/1024, ttl=254 (request in 2371)
2373	149.912138	92:1b:50:c1:b9:e3	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.125
2424	161.073515	Cisco_60:9c:e6	Broadcast	ARP	60	Who has 172.16.94.42? Tell 172.16.94.1

Рис. 8: фильтр arp or icmp

## Анализ кадров канального уровня в Wireshark

Изучим эхо-запрос и эхо-ответ ICMP в программе Wireshark:

Эхо-запрос: длина кадра - 74 байта, относится к типу Ethernet (1), MAC-адрес источника - 90:e8:68:fa:55:e3 (тип индивидуальный, глобально администрируемый), MAC-адрес шлюза - 70:18:a7:60:9c:e6 (тип индивидуальный, глобально администрируемый). (рис. 9)

# Анализ кадров канального уровня в Wireshark

+ 2364 146.547978 172.16.94.216 172.16.94.1 ICMP 74 Echo (ping) request id=0x0001, seq=1/256, ttl=128 (reply in 2365)
-< 2365 146.551747 172.16.94.1 172.16.94.216 ICMP 74 Echo (ping) reply id=0x0001, seq=1/256, ttl=254 (request in 2364)
2367 147.557766 172.16.94.216 172.16.94.1 ICMP 74 Echo (ping) request id=0x0001, seq=2/512, ttl=128 (reply in 2368)
2368 147.559846 172.16.94.1 172.16.94.216 ICMP 74 Echo (ping) reply id=0x0001, seq=2/512, ttl=254 (request in 2367)
2369 148.570093 172.16.94.216 172.16.94.1 ICMP 74 Echo (ping) request id=0x0001, seq=3/768, ttl=128 (reply in 2370)
2370 148.574770 172.16.94.1 172.16.94.216 ICMP 74 Echo (ping) reply id=0x0001, seq=3/768, ttl=254 (request in 2369)
2371 149.586784 172.16.94.216 172.16.94.1 ICMP 74 Echo (ping) request id=0x0001, seq=4/1024, ttl=128 (reply in 2372)
2372 149.591655 172.16.94.1 172.16.94.216 ICMP 74 Echo (ping) reply id=0x0001, seq=4/1024, ttl=254 (request in 2373)
2373 149.912138 92:1b:50:cf:b9:e3 Broadcast ARP 60 Who has 172.16.94.1? Tell 172.16.94.125
2424 161.073515 Cisco_60:9c:e6 Broadcast ARP 60 Who has 172.16.94.42? Tell 172.16.94.1

[Time shift for this packet: 0.000000000 seconds] [Time delta from previous captured frame: 0.932068000 seconds] [Time delta from previous displayed frame: 4.520725000 seconds] [Time since reference or first frame: 146.547978000 seconds] Frame Number: 2364 Frame Length: 74 bytes (592 bits) Capture Length: 74 bytes (592 bits) [Frame is marked: False] [Frame is ignored: False] [Protocols in frame: eth:ether:type:ip:icmp:data] [Coloring Rule Name: ICMP] [Coloring Rule String: icmp    icmpv6]	0000 70 18 a7 60 9c e6 90 e8 68 f4 0010 00 3c 97 cb 00 00 80 01 8d f7 0020 5e 01 08 00 4d 5a 00 01 00 01 0030 67 68 69 6a 6b 6c 6d 6e 6f 7e 0040 77 61 62 63 64 65 66 67 68 65
--	--

+ Ethernet II, Src: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3), Dst: Cisco_60:9c:e6 (70:18:a7:60:9c:e6) > Destination: Cisco_60:9c:e6 (70:18:a7:60:9c:e6) > Source: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3) Type: IPv4 (0x0800) [Stream index: 35]
---

Рис. 9: Эхо запрос

## Анализ кадров канального уровня в Wireshark

Эхо-ответ: длина кадра - 74 байта, относится к типу Ethernet (1), MAC-адрес источника - 70:18:a7:60:9c:e6 (тип индивидуальный, глобально администрируемый), MAC-адрес шлюза - 90:e8:68:fa:55:e3 (тип индивидуальный, глобально администрируемый) (рис. 10)

# Анализ кадров канального уровня в Wireshark

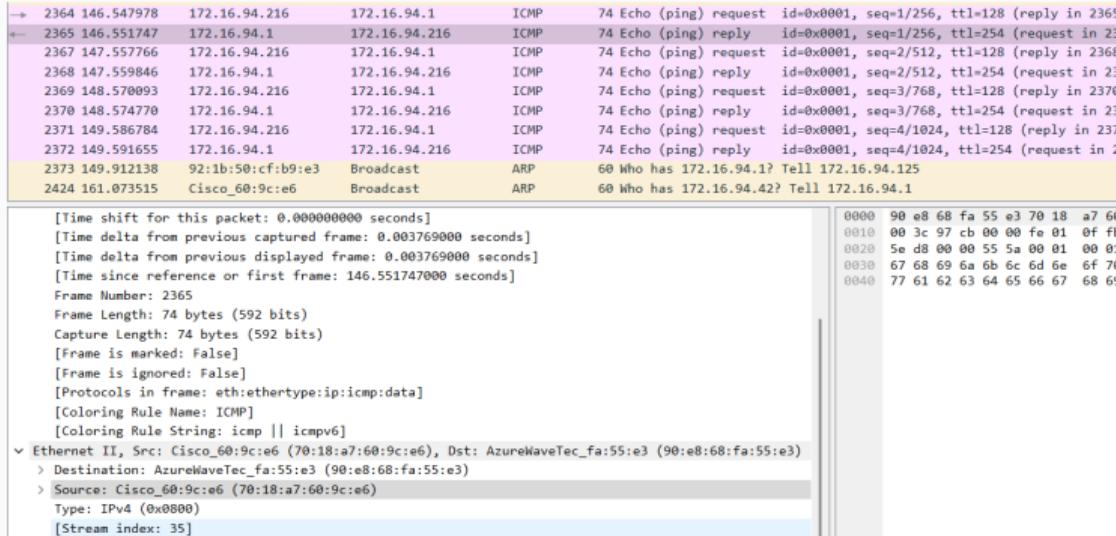


Рис. 10: Эхо ответ

## Анализ кадров канального уровня в Wireshark

Изучим кадры данных протокола ARP. Изучим данные в полях заголовка Ethernet II.

MAC-адрес назначения (Destination): ff:ff:ff:ff:ff:ff

Тип адреса: Широковещательный (Broadcast). ARP-запрос отправляется всем узлам в сети, так как отправителю неизвестен MAC-адрес получателя.

MAC-адрес источника (Source): 70:18:a7:60:9c:e6

Тип адреса: Индивидуальный, глобально администрируемый.

Производитель (OUI): 70:18:a7 – Cisco Systems. (рис. 11)

## Анализ кадров канального уровня в Wireshark

968	57.752156	Cisco_60:9c:e6	Broadcast	ARP	60 Who has 172.16.94.29? Tell 172.16.94.1
1021	65.739480	Cisco_60:9c:e6	Broadcast	ARP	60 Who has 172.16.94.29? Tell 172.16.94.1
1025	65.740822	TplinkTechno_59:85:..	Broadcast	ARP	60 Who has 172.16.94.204? Tell 172.16.94.121
1026	65.741263	TplinkTechno_59:9d:..	Broadcast	ARP	60 Who has 172.16.94.204? Tell 172.16.94.103
1035	67.275318	Cisco_60:9c:e6	Broadcast	ARP	60 Who has 172.16.94.75? Tell 172.16.94.1
1049	70.245344	Cisco_60:9c:e6	Broadcast	ARP	60 Who has 172.16.94.75? Tell 172.16.94.1
1057	72.087877	Intel_07:6e:99	Broadcast	ARP	60 Who has 172.16.94.110? Tell 172.16.95.24
1058	72.087917	AzureWaveTec_73:80:..	Broadcast	ARP	60 Who has 172.16.94.110? Tell 172.16.94.27
> Frame 968: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF_{7C57B6A3-1A75-4f					
▼ Ethernet II, Src: Cisco_60:9c:e6 (70:18:a7:60:9c:e6), Dst: Broadcast (ff:ff:ff:ff:ff:ff)					
▼ Destination: Broadcast (ff:ff:ff:ff:ff:ff)					
....1..... = LG bit: Locally administered address (this is NOT the factory default)					
....1..... = IG bit: Group address (multicast/broadcast)					
▼ Source: Cisco_60:9c:e6 (70:18:a7:60:9c:e6)					
....0..... = LG bit: Globally unique address (factory default)					
....0..... = IG bit: Individual address (unicast)					
Type: ARP (0x0806)					
[Stream index: 71]					
Padding: 00					
▼ Address Resolution Protocol (request)					
Hardware type: Ethernet (1)					
Protocol type: IPv4 (0x0800)					
Hardware size: 6					
Protocol size: 4					
Opcode: request (1)					
Sender MAC address: Cisco_60:9c:e6 (70:18:a7:60:9c:e6)					
Sender IP address: 172.16.94.1					
Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)					

Рис. 11: Протокол ARP

# Анализ кадров канального уровня в Wireshark

Начнем новый процесс захвата трафика в Wireshark. На устройстве в консоли пропингуем по имени какой-нибудь известный адрес, я возьму ya.ru (рис. 12)

```
C:\Windows\System32>ping ya.ru

Обмен пакетами с ya.ru [77.88.44.242] с 32 байтами данных:
Ответ от 77.88.44.242: число байт=32 время=32мс TTL=57
Ответ от 77.88.44.242: число байт=32 время=38мс TTL=57
Ответ от 77.88.44.242: число байт=32 время=7мс TTL=57
Ответ от 77.88.44.242: число байт=32 время=3мс TTL=57

Статистика Ping для 77.88.44.242:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
        (0% потеря)
Приблизительное время приема-передачи в мс:
    Минимальное = 3мсек, Максимальное = 38 мсек, Среднее = 20 мсек

C:\Windows\System32>
```

Рис. 12: Пинг ya.ru

## Анализ кадров канального уровня в Wireshark

Остановим захват трафика. Изучим запросы и ответы протоколов ARP и ICMP. Определим MAC-адреса источника и получателя, определим тип MAC-адресов.

Источник данных: IP-адрес: 172.16.94.39

MAC-адрес: 38:fc:98:ea:cf:68. Производитель (OUI): 38:fc:98 – Intel Corporate.  
Тип MAC-адреса: Индивидуальный, Глобально администрируемый.

Получатель данных (Destination): IP-адрес: 172.16.94.162.

MAC-адрес: ff:ff:ff:ff:ff:ff. Тип MAC-адреса: Широковещательный (Broadcast).  
Это адрес для отправки кадра всем устройствам в сети.

Устройство с IP 172.16.94.39 (Intel) не знает MAC-адрес устройства с IP 172.16.94.162 и широковещательно запрашивает его у всех в сети. (рис. 13)

# Анализ кадров канального уровня в Wireshark

870 15.877289	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.94.162? Tell 172.16.94.39
873 16.024582	172.16.94.216	77.88.44.242	ICMP	74 Echo (ping) request id=0x0001, seq=5/1280,
874 16.056519	77.88.44.242	172.16.94.216	ICMP	74 Echo (ping) reply id=0x0001, seq=5/1280,
910 17.031803	172.16.94.216	77.88.44.242	ICMP	74 Echo (ping) request id=0x0001, seq=6/1536,
920 17.070511	77.88.44.242	172.16.94.216	ICMP	74 Echo (ping) reply id=0x0001, seq=6/1536,
931 17.919556	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.94.234? Tell 172.16.94.39
932 17.920240	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.94.180? Tell 172.16.94.39
933 17.920246	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.94.246? Tell 172.16.94.39
934 17.920551	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.94.162? Tell 172.16.94.39
935 17.920771	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.95.30? Tell 172.16.94.39
936 17.920999	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.94.201? Tell 172.16.94.39
937 17.921205	Intel_ea:cf:68	Broadcast	ARP	60 Who has 172.16.95.43? Tell 172.16.94.39

> Frame 870: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF\_{7C57B6A3-1A75-4...

▼ Ethernet II, Src: Intel\_ea:cf:68 (38:fc:98:ea:cf:68), Dst: Broadcast (ff:ff:ff:ff:ff:ff)

  ▼ Destination: Broadcast (ff:ff:ff:ff:ff:ff)

    .... .1. .... .... .... = LG bit: Locally administered address (this is NOT the factory default)

    .... .1. .... .... .... = IG bit: Group address (multicast/broadcast)

  ▼ Source: Intel\_ea:cf:68 (38:fc:98:ea:cf:68)

    .... ..0. .... .... .... = LG bit: Globally unique address (factory default)

    .... ..0. .... .... .... = IG bit: Individual address (unicast)

  Type: ARP (0x0806)

  [Stream index: 16]

  Padding: 00

▼ Address Resolution Protocol (request)

  Hardware type: Ethernet (1)

  Protocol type: IPv4 (0x0800)

  Hardware size: 6

  Protocol size: 4

  Opcode: request (1)

  Sender MAC address: Intel\_ea:cf:68 (38:fc:98:ea:cf:68)

  Sender IP address: 172.16.94.39

  Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)

0000  
0010  
0020  
0030

Рис. 13: анализ ARP протокола

# Анализ кадров канального уровня в Wireshark

## Анализ ICMP-протокола

ICMP (эхо-запрос) это пакет, который отправляется с компьютера на удаленный сервер.

Содержание: 172.16.94.216 -> 77.88.44.242

Источник данных в кадре Ethernet II:

MAC-адрес: 90:e8:68:fa:55:e3 (Это беспроводной адаптер моего компьютера).

Тип MAC-адреса: Индивидуальный, Глобально администрируемый.

Получатель данных (Destination) в кадре Ethernet II:

MAC-адрес: 70:18:a7:60:9c:e6

Тип MAC-адреса: Индивидуальный, Глобально администрируемый.

# Анализ кадров канального уровня в Wireshark

Мой компьютер хочет отправить пакет на внешний IP 77.88.44.242. По правилам маршрутизации, он отправляет пакет на свой основной шлюз. Он уже знает MAC-адрес шлюза, поэтому кадр Ethernet адресован напрямую на MAC-адрес маршрутизатора. (рис. 14)

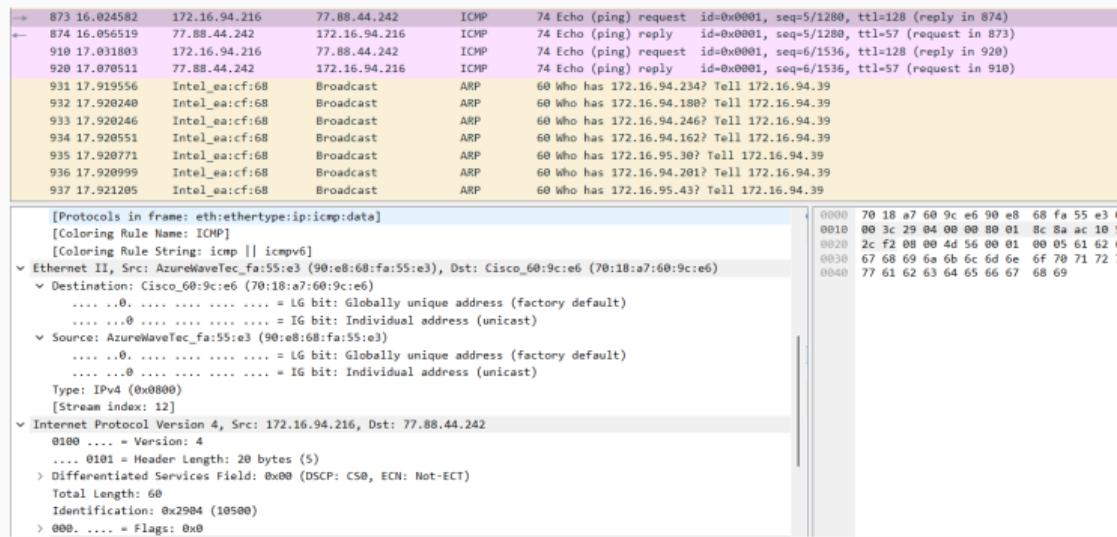


Рис. 14. ICMP в логе Wireshark

# Анализ кадров канального уровня в Wireshark

Это ответ, который возвращается с удаленного сервера на ваш компьютер.

Содержание: 77.88.44.242 -> 172.16.94.216

Источник данных в кадре Ethernet II:

MAC-адрес: 70:18:a7:60:9c:e6 (Cisco)

Тип MAC-адреса: Индивидуальный, Глобально администрируемый.

Получатель данных (Destination) в кадре Ethernet II:

MAC-адрес: 90:e8:68:fa:55:e3 (мой комп)

Тип MAC-адреса: Индивидуальный, Глобально администрируемый.

# Анализ кадров канального уровня в Wireshark

Маршрутизатор получил ответ от сервера и теперь должен доставить его моему компьютеру в локальной сети. Он знает MAC-адрес ПК и отправляет кадр Ethernet напрямую на мой MAC-адрес. (рис. 15)

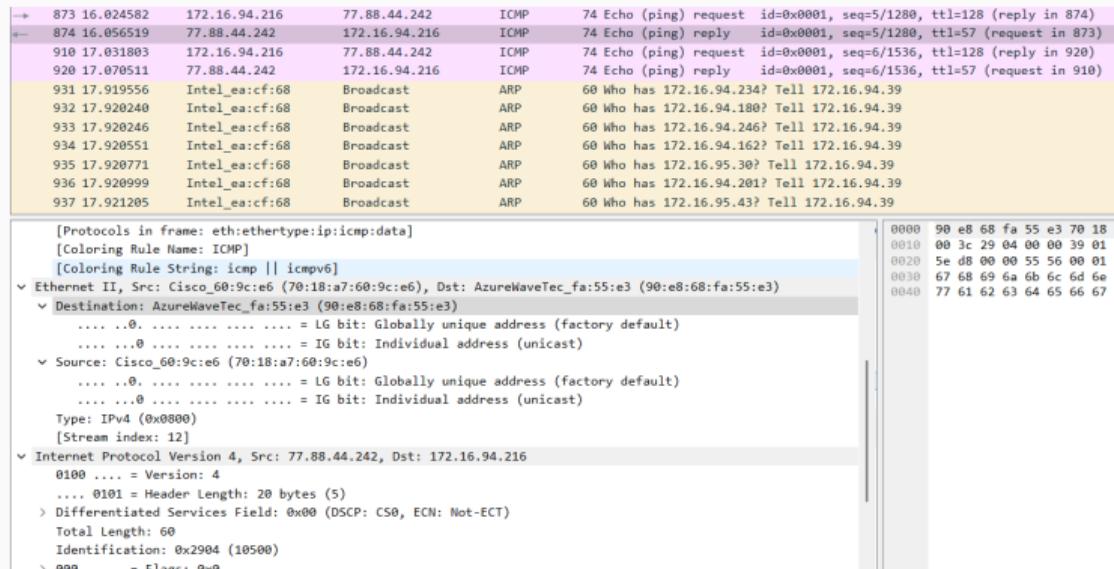
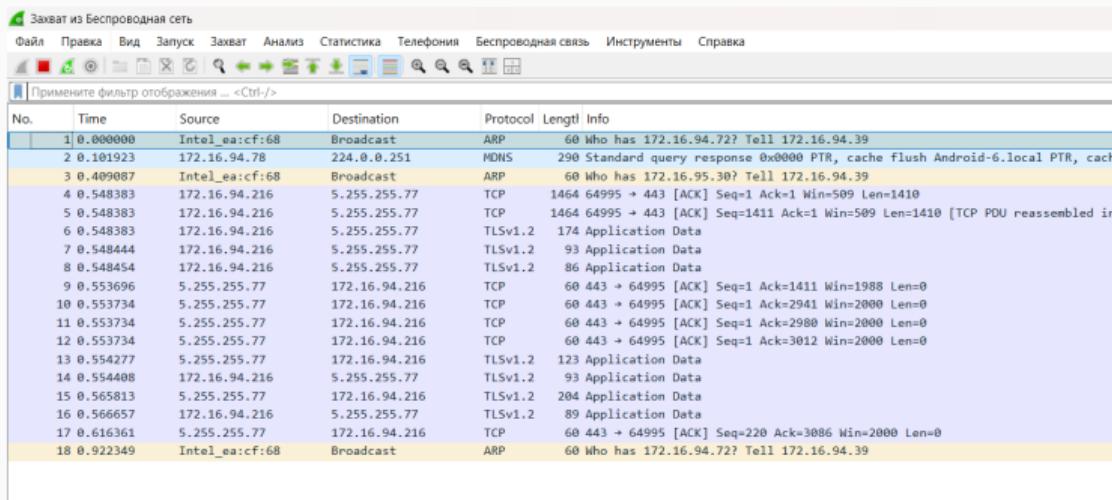


Рис. 15: ICMP эхо-ответ

# Анализ протоколов транспортного уровня в Wireshark

Запустим Wireshark. Выберем активный на устройстве сетевой интерфейс. Убедимся, что начался процесс захвата трафика. (рис. 16)



The screenshot shows the Wireshark interface with a list of captured network frames. The columns in the table are: No., Time, Source, Destination, Protocol, Length, and Info. The captured frames include various protocols such as ARP, DNS, TCP, and TLSv1.2, with details like sequence numbers, acknowledgments, and window sizes.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Intel_ea:cf:68	Broadcast	ARP	68	Who has 172.16.94.72? Tell 172.16.94.39
2	0.101923	172.16.94.78	224.0.0.251	MDNS	290	Standard query response 0x0000 PTR, cache flush Android-6.local PTR, cache
3	0.409087	Intel_ea:cf:68	Broadcast	ARP	60	Who has 172.16.95.30? Tell 172.16.94.39
4	0.548383	172.16.94.216	5.255.255.77	TCP	1464	64995 + 443 [ACK] Seq=1 Ack=1 Win=509 Len=1410
5	0.548383	172.16.94.216	5.255.255.77	TCP	1464	64995 + 443 [ACK] Seq=1411 Ack=1 Win=509 Len=1410 [TCP PDU reassembled in
6	0.548383	172.16.94.216	5.255.255.77	TLSv1.2	174	Application Data
7	0.548444	172.16.94.216	5.255.255.77	TLSv1.2	93	Application Data
8	0.548454	172.16.94.216	5.255.255.77	TLSv1.2	86	Application Data
9	0.553696	5.255.255.77	172.16.94.216	TCP	60	443 + 64995 [ACK] Seq=1 Ack=1411 Win=1988 Len=0
10	0.553734	5.255.255.77	172.16.94.216	TCP	60	443 + 64995 [ACK] Seq=1 Ack=2941 Win=2000 Len=0
11	0.553734	5.255.255.77	172.16.94.216	TCP	60	443 + 64995 [ACK] Seq=1 Ack=2980 Win=2000 Len=0
12	0.553734	5.255.255.77	172.16.94.216	TCP	60	443 + 64995 [ACK] Seq=1 Ack=3012 Win=2000 Len=0
13	0.554277	5.255.255.77	172.16.94.216	TLSv1.2	123	Application Data
14	0.554408	172.16.94.216	5.255.255.77	TLSv1.2	93	Application Data
15	0.565813	5.255.255.77	172.16.94.216	TLSv1.2	204	Application Data
16	0.566657	172.16.94.216	5.255.255.77	TLSv1.2	89	Application Data
17	0.616361	5.255.255.77	172.16.94.216	TCP	60	443 + 64995 [ACK] Seq=220 Ack=3086 Win=2000 Len=0
18	0.922349	Intel_ea:cf:68	Broadcast	ARP	60	Who has 172.16.94.72? Tell 172.16.94.39

Рис. 16: Захват трафика

# Анализ протоколов транспортного уровня в Wireshark

В браузере перейдем на сайт, работающий по протоколу HTTP (например, на сайт CERN <http://info.cern.ch/>). (рис. 17)

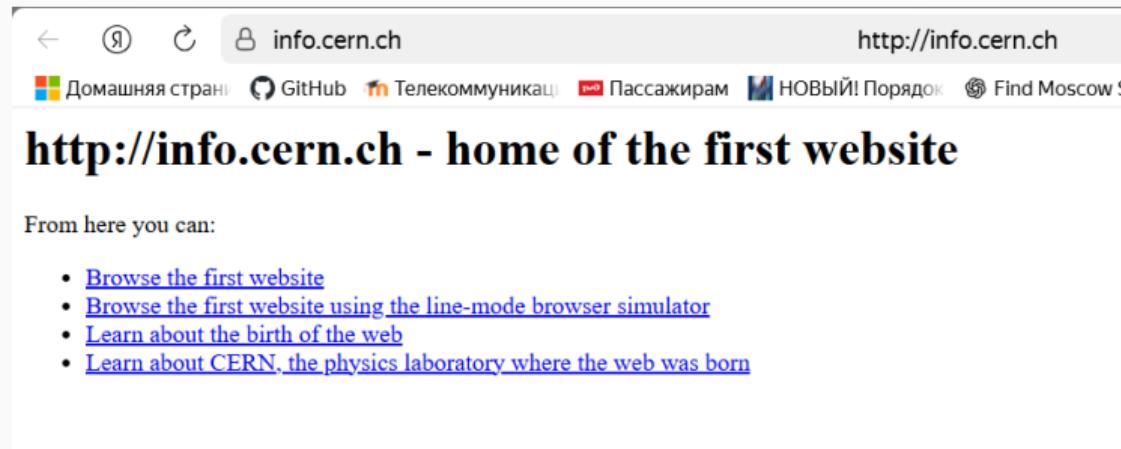


Рис. 17: Переход на сайт cern.ch

## Анализ протоколов транспортного уровня в Wireshark

В Wireshark в строке фильтра укажем http. Это пакет, в котором компьютер отправляет HTTP-запрос на сервер. Анализируем TCP (запрос)

Порты: Source Port: 65232 — это исходный порт, который случайным образом выбирается компьютером. Destination Port: 80 — это порт назначения

Sequence Number (последовательный номер): Relative (относительный): 1, Raw (абсолютный): 4193937875. Это номер первого байта данных в этом сегменте. Wireshark для удобства показывает относительный номер, начиная с 1.

## Анализ протоколов транспортного уровня в Wireshark

Acknowledgment Number (номер подтверждения): Relative (относительный): 1, Raw (абсолютный): 2237714506. Это номер следующего байта, который отправитель этого пакета ожидает получить от противоположной стороны. Подтверждает успешное получение всех данных до этого номера.

Flags: PSH, ACK. ACK (Acknowledgment): Установлен, чтобы подтвердить получение предыдущих пакетов от сервера. PSH (Push): Указывает получателю (серверу) немедленно передать данные приложению (веб-серверу), не дожидаясь заполнения буфера.

TCP Segment Len (длина сегмента): 254. (рис. 18)

## Анализ протоколов транспортного уровня в Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
+ 561	39.445586	104.81.99.218	172.16.94.216	HTTP	369	HTTP/1.1 304 Not Modified
+ 557	39.398068	172.16.94.216	104.81.99.218	HTTP	308	GET /DigiCertGlobalRootG2.crl HTTP/1.1

---

> Frame 561: 369 bytes on wire (2952 bits), 369 bytes captured (2952 bits) on interface \Device\NPF\_{7C57B6A3-1A75-471F-BE9E-33E870D95D9C} (Intel PRO/100 MT Desktop (v3) Intel(R) Dual Band Wireless-AC 7265 (PCIe))  
 > Ethernet II, Src: Cisco\_60:9c:e6 (70:18:a7:60:9c:e6), Dst: AzureWaveTec\_fa:55:e3 (90:e8:68:fa:55:e3)  
 > Internet Protocol Version 4, Src: 104.81.99.218, Dst: 172.16.94.216  
 ✓ Transmission Control Protocol, Src Port: 80, Dst Port: 65232, Seq: 1, Ack: 255, Len: 315  
     Source Port: 80  
     Destination Port: 65232  
     [Stream index: 37]  
     [Stream Packet Number: 6]  
 > [Conversation completeness: Incomplete, DATA (15)]  
     [TCP Segment Len: 315]  
     Sequence Number: 1 (relative sequence number)  
     Sequence Number (raw): 2237714506  
     [Next Sequence Number: 316 (relative sequence number)]  
     Acknowledgment Number: 255 (relative ack number)  
     Acknowledgment number (raw): 4193938129  
     0101 .... = Header Length: 20 bytes (5)  
 > Flags: 0x018 (PSH, ACK)  
     Window: 501  
     [Calculated window size: 64128]  
     [Window size scaling factor: 128]  
     Checksum: 0x60ed [unverified]  
     [Checksum Status: Unverified]  
     Urgent Pointer: 0  
 > [Timestamps]  
 > [SEQ/ACK analysis]  
 TCP payload (315 bytes)

0000	90 e8 68 fa
0010	01 63 e7 ed
0020	5e d8 00 50
0030	01 f5 60 ed
0040	30 34 20 4e
0050	0a 43 6f 6e
0060	70 70 6c 69
0070	63 72 6c 0d
0080	65 64 3a 20
0090	32 30 32 35
00a0	54 0d 0a 45
00b0	37 65 2d 34
00c0	6f 6e 74 72
00d0	6d 61 78 2d
00e0	70 69 72 65
00f0	65 70 20 32
0100	20 47 4d 54
0110	20 32 39 20
0120	32 33 3a 35
0130	63 74 69 6f
0140	65 0d 0a 41
0150	2e 36 35 33
0160	38 36 33 37
0170	0a

**Рис. 18:** Анализ TCP (запрос)

## Анализ протоколов транспортного уровня в Wireshark

Порты: Source Port (порт источника): 80. Destination Port (порт назначения): 65232.

Sequence Number (последовательный номер): Relative (относительный): 1, Raw (абсолютный): 2237714506. Это номер первого байта данных, которые сервер отправляет в этом сегменте.

Acknowledgment Number (номер подтверждения): Relative (относительный): 255, Raw (абсолютный): 4193938129

Flags: PSH, ACK. ACK (Acknowledgment): Подтверждает получение HTTP-запроса от клиента. PSH (Push): Указывает компьютеру немедленно передать данные (HTTP-ответ) приложению (браузеру).

TCP Segment Len (длина сегмента): 315. Размер полезных данных (HTTP-ответа), переносимых в этом TCP-сегменте. (рис. 19)

# Анализ протоколов транспортного уровня в Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
561	39.445586	104.81.99.218	172.16.94.216	HTTP	369	HTTP/1.1 304 Not Modified
557	39.398068	172.16.94.216	104.81.99.218	HTTP	308	GET /DigiCertGlobalRootG2.crl HTTP/1.1

> Frame 557: 308 bytes on wire (2464 bits), 308 bytes captured (2464 bits) on interface \Device\NPF\_{7C57B6A3-1A75-471F-BE9E-33E871D9A2D9  
> Ethernet II, Src: AzureWaveTec\_fa:55:e3 (90:e8:68:fa:55:e3), Dst: Cisco\_60:9c:e6 (70:18:a7:60:9c:e6)  
> Internet Protocol Version 4, Src: 172.16.94.216, Dst: 104.81.99.218  
▼ Transmission Control Protocol, Src Port: 65232, Dst Port: 80, Seq: 1, Ack: 1, Len: 254  
    Source Port: 65232  
    Destination Port: 80  
    [Stream index: 37]  
    [Stream Packet Number: 4]  
    > [Conversation completeness: Incomplete, DATA (15)]  
    [TCP Segment Len: 254]  
    Sequence Number: 1 (relative sequence number)  
    Sequence Number (raw): 4193937875  
    [Next Sequence Number: 255 (relative sequence number)]  
    Acknowledgment Number: 1 (relative ack number)  
    Acknowledgment number (raw): 2237714566  
    0101 .... = Header Length: 20 bytes (5)  
    > Flags: 0x018 (PSH, ACK)  
    Window: 255  
    [Calculated window size: 65280]  
    [Window size scaling factor: 256]  
    Checksum: 0x644d [unverified]  
    [Checksum Status: Unverified]  
    Urgent Pointer: 0  
    > [Timestamps]  
    > [SEQ/ACK analysis]  
    TCP payload (254 bytes)  
    > Hypertext Transfer Protocol

0000 70 18 a7 60 9c e6 90 e8 66 ·  
0018 01 26 4e fe 40 00 80 06 d3 1 ·  
0020 63 da fe d0 00 50 f9 fa 69 ·  
0030 08 ff 64 4d 00 00 47 45 54 :  
0040 65 72 74 47 6c f6 62 61 6c :  
0050 63 72 6c 28 48 54 54 58 2f :  
0060 63 68 65 2d 43 f6 66 74 72 :  
0070 2d 61 67 65 20 3d 20 33 35 :  
0080 6e 65 63 74 69 f6 6e 3a 20 :  
0090 69 76 65 0d 0a 41 63 63 65 :  
00a0 0d 0a 49 66 2d 4d 6f 64 69 :  
00b0 6e 63 65 3a 20 57 65 64 2c :  
00c0 20 32 30 32 35 20 32 30 3a :  
00d0 4d 54 0d 0a 49 66 2d 4e 6f :  
00e0 68 3a 20 22 36 38 64 34 35 :  
00f0 22 0d 0a 55 73 65 72 2d 41 :  
0100 69 63 72 6f 73 6f 66 74 2d :  
0110 50 49 2f 31 30 2e 30 0d 0a :  
0120 72 6c 33 2e 64 69 67 69 63 :  
0130 0d 0a 0d 0a

Рис. 19: Анализ TCP (ответ)

## Анализ протоколов транспортного уровня в Wireshark

UDP в DNS-запросе. Это пакет, в котором компьютер отправляет вопрос DNS-серверу.

Source Port (порт источника): 62966. Destination Port (порт назначения): 53. Это стандартный порт для службы DNS.

Length (длина): 55 байт. Общая длина UDP-датаграммы (заголовок + данные). (рис. 20)

# Анализ протоколов транспортного уровня в Wireshark

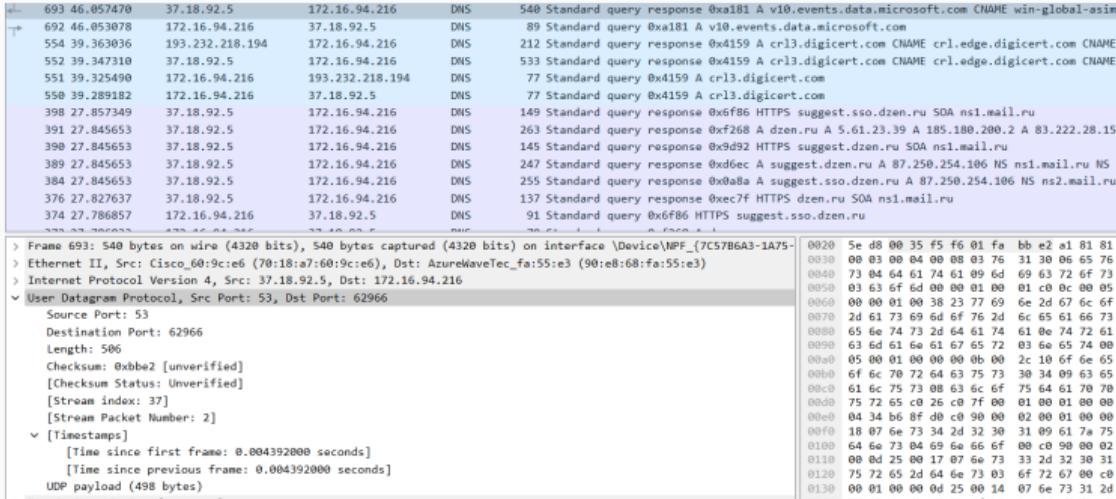


Рис. 20: Анализ UDP (запрос)

## Анализ протоколов транспортного уровня в Wireshark

UDP в DNS-ответе. Это пакет, в котором DNS-сервер возвращает ответ компьютеру.

Source Port (порт источника): 53. Destination Port (порт назначения): 62966.

Length (длина): 506 байт. Общая длина этой UDP-датаграммы. (рис. 21)

# Анализ протоколов транспортного уровня в Wireshark

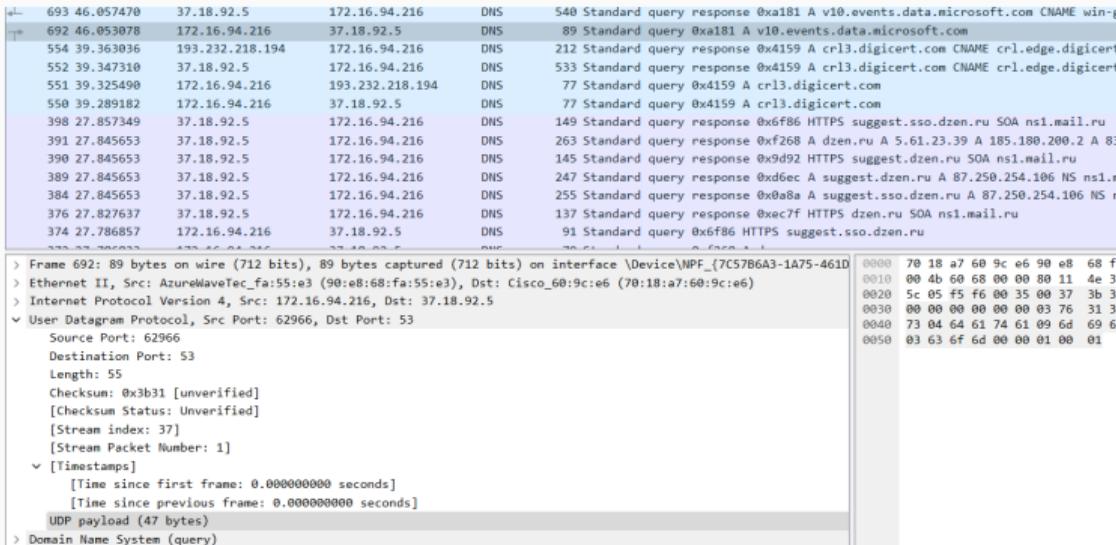


Рис. 21: Анализ UDP (ответ)

## Анализ протоколов транспортного уровня в Wireshark

Wireshark в строке фильтра укажем `quic` и проанализируем информацию по протоколу `quic` в случае запросов и ответов.

QUIC работает поверх UDP, поэтому в заголовках мы сначала видим Ethernet, затем IP, затем UDP, и только потом – QUIC.

QUIC-пакет от Клиента. Это пакет, который компьютер отправляет на сервер.

Транспорт (UDP): Source Port (порт источника): 62198, Destination Port (порт назначения): 443 (порт QUIC/HTTPS сервера)

# Анализ протоколов транспортного уровня в Wireshark

QUIC Заголовок:

Destination Connection ID (DCID): f8a9378badb7bea3. Идентификатор соединения назначения. Это уникальный идентификатор, который клиент использует для указания, с каким именно серверным соединением связан этот пакет. Сервер изначально предоставляет этот ID клиенту.

Полезная нагрузка: Remaining Payload: d6e43ad86a11... Основные данные протокола. Как видно из названия пакета в общем списке – “Protected Payload” – эта часть зашифрована. Это одно из ключевых преимуществ QUIC – шифрование, и даже заголовки полей, относящиеся к управлению соединением, шифруются. (рис. 22)

# Анализ протоколов транспортного уровня в Wireshark

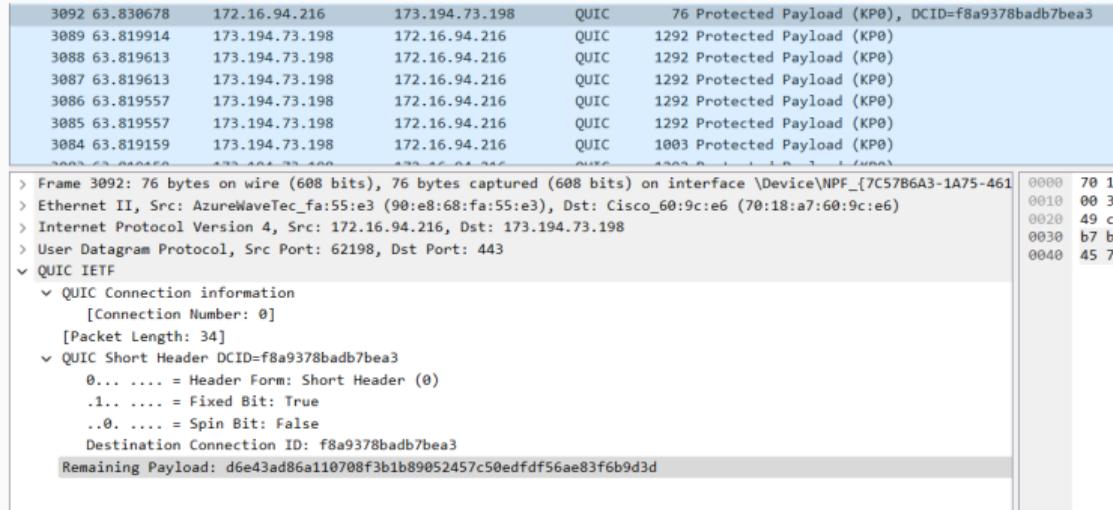


Рис. 22: Анализ QUIC (запрос)

# Анализ протоколов транспортного уровня в Wireshark

Транспорт (UDP): Source Port (порт источника): 443, Destination Port (порт назначения): 62198 (сервер отправляет ответ на исходный порт ПК)

QUIC Заголовок:

В ответах сервер будет использовать тот Connection ID, который был предоставлен клиенту (или новый, согласованный в процессе установления соединения).

Полезная нагрузка: Remaining Payload: 3fce... Аналогично клиентскому пакету, полезная нагрузка сервера также является “Protected Payload” (Защищенной) и зашифрована. (рис. 23)

# Анализ протоколов транспортного уровня в Wireshark

The screenshot shows a list of QUIC packets from a network capture. The first few packets are highlighted in blue, indicating they are selected. The interface displays columns for packet number, source IP, destination IP, source port, destination port, protocol, and payload type. Below the list, detailed information is provided for the selected packet (packet 3089), including its length, checksum status, timestamps, and QUIC-specific fields like connection information and short header details. The right side of the interface shows the raw hex and ASCII data for the selected packet.

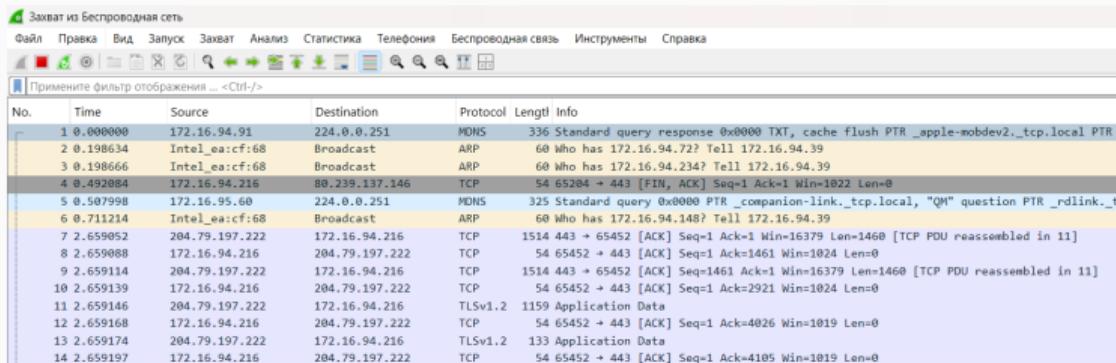
Length:	1258
Checksum:	0x2c11 [unverified]
[Checksum Status:	Unverified]
[Stream index:	84]
[Stream Packet Number:	819]
[Timestamps]	
[Time since first frame:	0.292084000 seconds]
[Time since previous frame:	0.000301000 seconds]
UDP payload (1250 bytes)	
QUIC IETF	
QUIC Connection information	
[Connection Number:	0]
[Packet Length:	1250]
QUIC Short Header	
0... .... = Header Form: Short Header (0)	
.1... .... = Fixed Bit: True	
..0. .... = Spin Bit: False	
Remaining Payload [...]:	3fceal2c11c43be28dfbb8ba073cdfb9edfe90082cc44a85085309ced00b8034443b5f3420bc3a1d7bf57

0020 5e d8 01 bb f  
0030 c4 3b e2 8d f  
0040 c4 4a 85 08 5  
0050 bc 3a 1d 7b f  
0060 08 41 2e 3f 6  
0070 ed ba dc f5 f  
0080 3f 4f bf eb b  
0090 9e b7 5e 31 e  
00a0 9e 77 71 2b a  
00b0 49 59 84 04 b  
00c0 8f b2 39 c7 8  
00d0 a5 c7 d7 41 9  
00e0 68 1b d3 79 b  
00f0 06 02 0e f9 7  
0100 f0 ad 7d ed 3  
0110 f6 8f e1 b2 1  
0120 69 fb cb 13 4  
0130 5d 7f 80 66 d  
0140 f4 ce 9b 5a 8  
0150 fe 83 8d 93 f  
0160 30 16 1c 0e b  
0170 2c 4c 9b dc 2

Рис. 23: Анализ QUIC (ответ)

# Анализ handshake протокола TCP в Wireshark

Запустим Wireshark. Выберем активный на устройстве сетевой интерфейс. Убедимся, что начался процесс захвата трафика. (рис. 24)



The screenshot shows the Wireshark interface with a single packet selected for inspection. The packet details pane shows a DNS query from 172.16.94.91 to 224.0.0.251. The bytes pane shows the raw hex and ASCII data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.94.91	224.0.0.251	MDNS	336	Standard query response 0x0000 TXT, cache flush PTR _apple-mobdev2._tcp.local PTR (
2	0.198634	Intel_eacf:68	Broadcast	ARP	60	Who has 172.16.94.72? Tell 172.16.94.39
3	0.198666	Intel_eacf:68	Broadcast	ARP	60	68 Who has 172.16.94.234? Tell 172.16.94.39
4	0.492084	172.16.94.216	80.239.137.146	TCP	54	65204 → 443 [FIN, ACK] Seq=1 Ack=1 Win=1022 Len=8
5	0.507998	172.16.95.60	224.0.0.251	MDNS	325	Standard query 0x0000 PTR _companiion-link._tcp.local, "QM" question PTR _rdlink._tc
6	0.711214	Intel_eacf:68	Broadcast	ARP	60	68 Who has 172.16.94.148? Tell 172.16.94.39
7	2.659852	204.79.197.222	172.16.94.216	TCP	1514	443 → 65452 [ACK] Seq=1 Ack=1 Win=16379 Len=1460 [TCP PDU reassembled in 11]
8	2.659088	172.16.94.216	204.79.197.222	TCP	54	65452 → 443 [ACK] Seq=1 Ack=1461 Win=1024 Len=0
9	2.659114	204.79.197.222	172.16.94.216	TCP	1514	443 → 65452 [ACK] Seq=1461 Ack=1 Win=16379 Len=1460 [TCP PDU reassembled in 11]
10	2.659139	172.16.94.216	204.79.197.222	TCP	54	65452 → 443 [ACK] Seq=1 Ack=2921 Win=1024 Len=0
11	2.659146	204.79.197.222	172.16.94.216	TLSv1.2	1159	Application Data
12	2.659168	172.16.94.216	204.79.197.222	TCP	54	65452 → 443 [ACK] Seq=1 Ack=4026 Win=1019 Len=0
13	2.659174	204.79.197.222	172.16.94.216	TLSv1.2	133	Application Data
14	2.659197	172.16.94.216	204.79.197.222	TCP	54	65452 → 443 [ACK] Seq=1 Ack=4105 Win=1019 Len=0

Рис. 24: Захват трафика

# Анализ handshake протокола TCP в Wireshark

Использую соединение по HTTP с каким-то сайтом для захвата в Wireshark пакетов TCP. В Wireshark проанализируем handshake протокола TCP.

Шаг 1: SYN (Синхронизация). Клиент инициирует соединение, отправляя серверу специальный пакет.

Ключевые поля TCP:

Sequence Number: 0. Клиент генерирует начальный номер последовательности (ISN).

Flags: SYN (0x002). Установлен только флаг SYN. Это запрос на синхронизацию и начало соединения.

## Анализ handshake протокола TCP в Wireshark

---

Параметры (Options): MSS=1460 — максимальный размер сегмента, который клиент может принять, WS=256 — фактор масштабирования окна для увеличения его эффективного размера, SACK\_PERM — поддержка выборочных подтверждений (Selective Acknowledgements).

Клиент говорит серверу: я хочу установить соединение. Мой начальный номер последовательности — 0, и вот мои параметры. (рис. 25)

# Анализ handshake протокола TCP в Wireshark

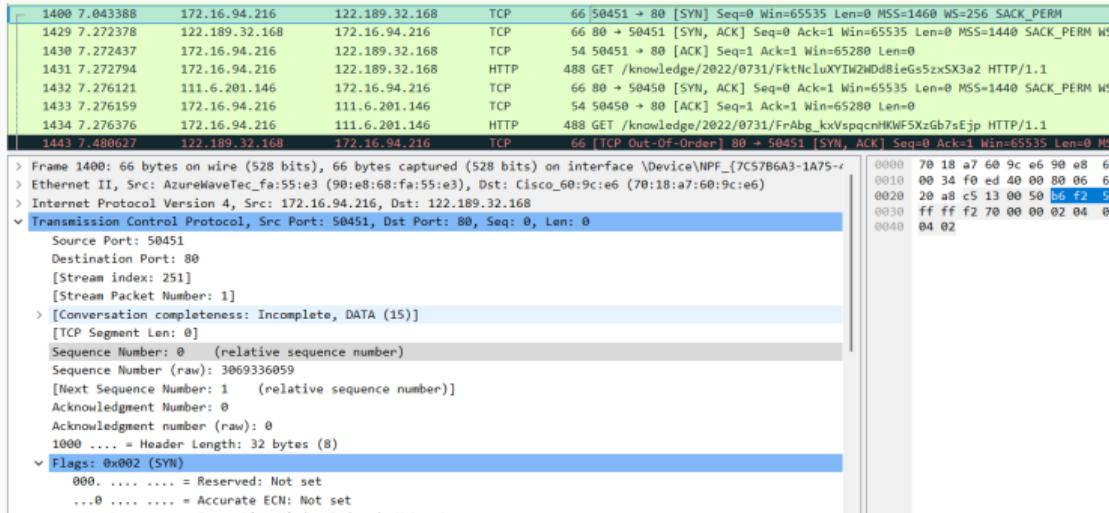


Рис. 25: TCP handshake шаг 1

## Анализ handshake протокола TCP в Wireshark

Шаг 2: SYN-ACK (Синхронизация-Подтверждение). Сервер отвечает, подтверждая запрос клиента и отправляя свой собственный запрос на синхронизацию.

Sequence Number: 0. Сервер генерирует свой собственный начальный номер последовательности (ISN).

Acknowledgment Number: 1. Это поле подтверждения. Сервер подтверждает получение SYN-пакета клиента.

Flags: SYN, ACK (0x012). Установлены флаги SYN (запрос синхронизации от сервера) и ACK (подтверждение пакета клиента).

Сервер отвечает клиенту: Я получил запрос на соединение (ACK=1). Я согласен установить соединение, и мой начальный номер последовательности — 0 (SYN). Вот мои параметры. (рис. 26)

# Анализ handshake протокола TCP в Wireshark

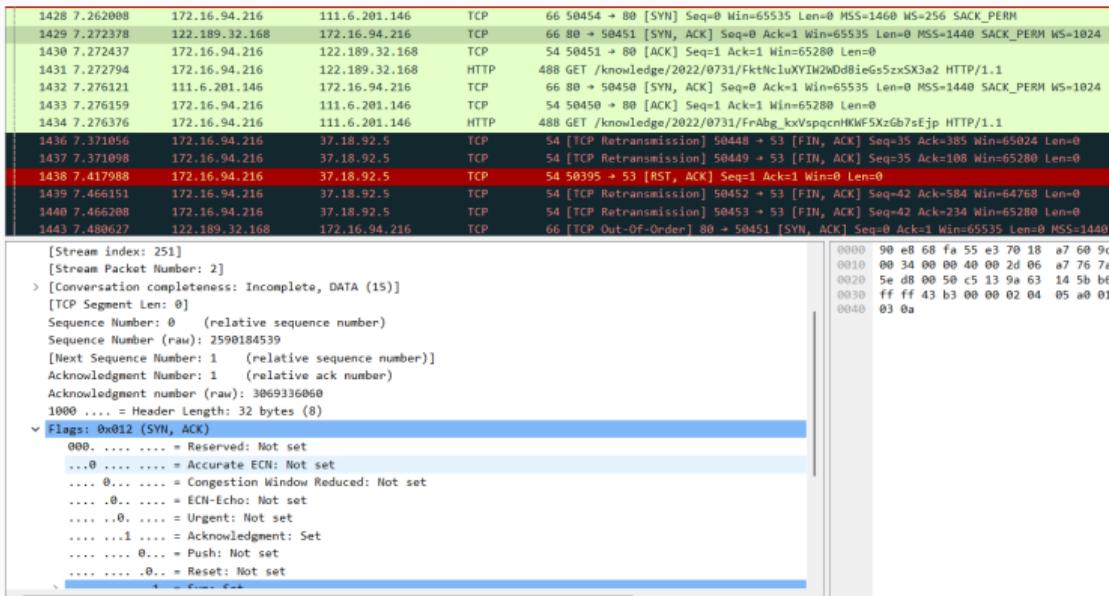


Рис. 26: TCP handshake шаг 2

# Анализ handshake протокола TCP в Wireshark

Шаг 3: ACK (Подтверждение). Клиент завершает рукопожатие, подтверждая SYN-пакет сервера.

Ключевые поля TCP:

Sequence Number (относительный): 1. Номер последовательности клиента теперь 1, так как его SYN-пакет “потребил” номер 0.

Acknowledgment Number (относительный): 1. Это поле подтверждения.

Клиент подтверждает получение SYN-пакета сервера. Номер подтверждения равен ISN сервера + 1. Ack=1.

Flags: ACK (0x010). Установлен только флаг ACK.

Клиент говорит серверу: Я получил SYN-пакет. Соединение установлено.  
(рис. 27)

# Анализ handshake протокола TCP в Wireshark

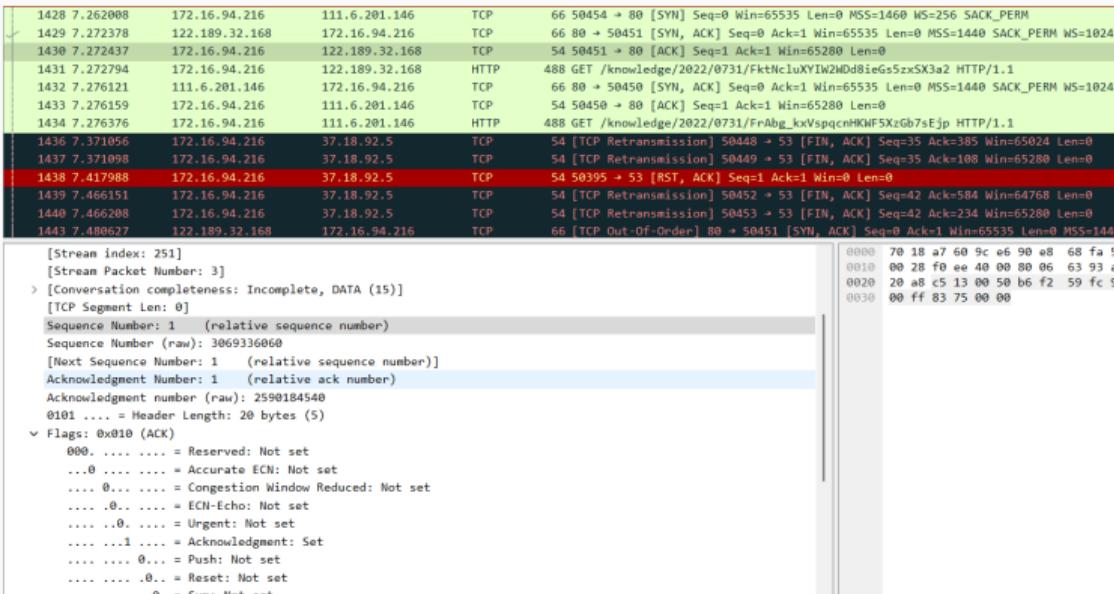


Рис. 27: TCP handshake шаг 3

## **Вывод**

---

## Вывод

В ходе выполнения лабораторной работы мы изучили посредством Wireshark кадры Ethernet, проанализировали PDU протоколы транспортного и прикладного уровней стека TCP/IP.