

# Лабораторная работа №3

Сетевые технологии

---

Иванов Сергей Владимирович, НПИбд-01-23

4 октября 2025

Российский университет дружбы народов, Москва, Россия

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

```
C:\Windows\System32>
C:\Windows\System32>ipconfig

Настройка протокола IP для Windows

Адаптер Ethernet Ethernet 3:

    DNS-суффикс подключения . . . . . :
    Локальный IPv6-адрес канала . . . : fe80::926a:5553:d565:4c13%8
    IPv4-адрес. . . . . : 192.168.56.1
    Маска подсети . . . . . : 255.255.255.0
    Основной шлюз. . . . . :

Адаптер беспроводной локальной сети Подключение по локальной сети* 9:

    Состояние среды. . . . . : Среда передачи недоступна.
    DNS-суффикс подключения . . . . . :

Адаптер беспроводной локальной сети Подключение по локальной сети* 10:

    Состояние среды. . . . . : Среда передачи недоступна.
    DNS-суффикс подключения . . . . . :

Адаптер Ethernet outline-tap0:

    Состояние среды. . . . . : Среда передачи недоступна.
    DNS-суффикс подключения . . . . . :
```

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.

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
```

```
C:\Windows\System32>ipconfig /all

Настройка протокола IP для Windows

Имя компьютера . . . . . : Rtiop
Основной DNS-суффикс . . . . . :
Тип узла. . . . . : Гибридный
IP-маршрутизация включена . . . . : Нет
WINS-прокси включен . . . . . : Нет

Адаптер Ethernet Ethernet 3:

DNS-суффикс подключения . . . . . :
Описание. . . . . : VirtualBox Host-Only Ethernet Adapter
Физический адрес. . . . . : 0A-00-27-00-00-08
DHCP включен. . . . . : Нет
Автонастройка включена. . . . . : Да
Локальный IPv6-адрес канала . . . : fe80::926a:5553:d565:4c13%8(Основной)
IPv4-адрес. . . . . : 192.168.56.1(Основной)
Маска подсети . . . . . : 255.255.255.0
Основной шлюз. . . . . :
IAID DHCPv6 . . . . . : 722075687
DUID клиента DHCPv6 . . . . . : 00-01-00-01-2D-00-EA-E7-58-11-22-88-F5-69
NetBios через TCP/IP. . . . . : Включен

Адаптер беспроводной локальной сети Подключение по локальной сети* 9:

Состояние среды. . . . . : Среда передачи недоступна.
DNS-суффикс подключения . . . . . :
Описание. . . . . : Microsoft Wi-Fi Direct Virtual Adapter
Физический адрес. . . . . : 92-E8-68-FA-55-A3
DHCP включен. . . . . : Да
Автонастройка включена. . . . . : Да
```

Рис. 3: Команда ipconfig /all

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.

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 шлюз: 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.

Определим MAC-адрес интерфейса 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.

```
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
```

Рис. 4: MAC-адрес



90-E8-68 принадлежит компании AzureTechnolody (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)

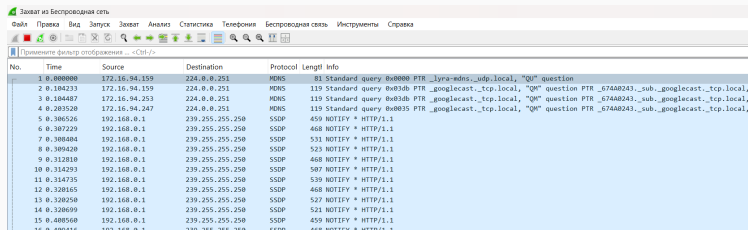


Рис. 5: Захват трафика

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)

```
Адаптер беспроводной локальной сети Беспроводная сеть:
```

```
DNS-суффикс подключения . . . . . :  
Локальный IPv6-адрес канала . . . : fe80::2fef:1cd:c5e8:217a%14  
IPv4-адрес. . . . . : 172.16.94.216  
Маска подсети . . . . . : 255.255.254.0  
Основной шлюз. . . . . : 172.16.94.1
```

**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\lserg>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

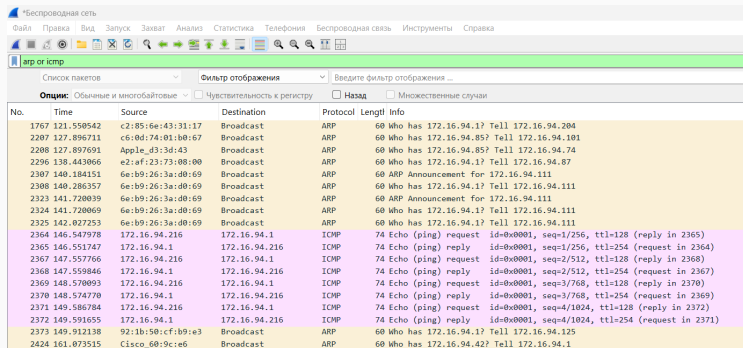
Статистика Ping для 172.16.94.1:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 2мсек, Максимальное = 4 мсек, Среднее = 3 мсек

C:\Users\lserg>
```

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 network protocol analyzer interface. The top menu bar includes 'Файл', 'Правка', 'Вид', 'Запуск', 'Захват', 'Анализ', 'Статистика', 'Телефония', 'Беспроводная связь', 'Инструменты', and 'Справка'. Below the menu is a toolbar with various icons. The main window is divided into three panes. The top pane shows the filter 'arp or icmp' in the 'Фильтр отображения' field. The middle pane shows the 'Опции' (Options) section with 'Обычные и многобайтовые' selected and 'Чувствительность к регистру' unchecked. The bottom pane displays a list of captured packets, filtered to show only ARP and ICMP traffic. The table has columns for 'No.', 'Time', 'Source', 'Destination', 'Protocol', 'Length', and 'Info'.

No.	Time	Source	Destination	Protocol	Length	Info
1767	121.559542	c2:85:6e:43:31:17	Broadcast	ARP	60	Who has 172.16.94.1? Tell 172.16.94.204
2207	127.896711	c6:0d: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	138.443066	e2:af:23:73:08:00	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	148.574779	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: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

Рис. 8: фильтр `arp or icmp`

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 2371)
	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: 0.00000000 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:ethertype:ip:icmp:data]  
[Coloring Rule Name: ICMP]  
[Coloring Rule String: icmp || icmpv6]  
▼ 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]

0000	70	18	a7	60	9c	e6	90	e8	68	fa
0010	00	3c	97	cb	00	00	80	01	8d	fd
0020	5e	01	08	00	4d	5a	00	01	00	00
0030	67	68	69	6a	6b	6c	6d	6e	6f	7e
0040	77	61	62	63	64	65	66	67	68	6e

Рис. 9: Эхо запрос

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

→	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: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]	0000	90 e8 68 fa 55 e3 70 18 a7 60
[Time delta from previous captured frame: 0.003769000 seconds]	0010	00 3c 97 cb 00 00 fe 01 0f fb
[Time delta from previous displayed frame: 0.003769000 seconds]	0020	5e d8 00 00 55 5a 00 01 00 01
[Time since reference or first frame: 146.551747000 seconds]	0030	67 68 69 6a 6b 6c 6d 6e 6f 70
Frame Number: 2365	0040	77 61 62 63 64 65 66 67 68 69
Frame Length: 74 bytes (592 bits)		
Capture Length: 74 bytes (592 bits)		
[Frame is marked: False]		
[Frame is ignored: False]		
[Protocols in frame: eth:ethertype:ip:icmp:data]		
[Coloring Rule Name: ICMP]		
[Coloring Rule String: icmp    icmpv6]		
▼ Ethernet II, Src: Cisco_60:9c:e6 (70:18:a7:60:9c:e6), Dst: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3)		
> Destination: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3)		
> Source: Cisco_60:9c:e6 (70:18:a7:60:9c:e6)		
Type: IPv4 (0x0800)		
[Stream index: 35]		

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_{7C57B6A3-1A75-4
< 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)
    .... 01. .... = LG bit: Locally administered address (this is NOT the factory default)
    .... 01. .... = IG bit: Group address (multicast/broadcast)
  < Source: Cisco_60:9c:e6 (70:18:a7:60:9c:e6)
    .... 00. .... = LG bit: Globally unique address (factory default)
    .... 00. .... = IG bit: Individual address (unicast)
  Type: ARP (0x0806)
  [Stream index: 71]
  Padding: 00000000000000000000000000000000
< 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% потерь)
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_{7C57B6A3-1A75-4t	0000
▼	Ethernet II, Src: Intel_ea:cf:68 (38:fc:98:ea:cf:68), Dst: Broadcast (ff:ff:ff:ff:ff:ff)	0010
▼	Destination: Broadcast (ff:ff:ff:ff:ff:ff)	0020
	.....1..... = LG bit: Locally administered address (this is NOT the factory default)	0030
	.....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)	

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.

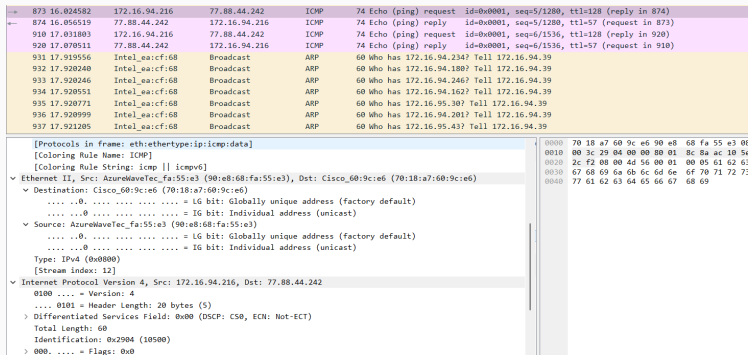
Получатель данных (Destination) в кадре Ethernet II:

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)



No.	Time	Source	Destination	Protocol	Length	Info
873	16.024582	172.16.94.216	77.88.44.242	ICMP	74	Echo (ping) request id=0x0001, seq=5/1280, ttl=128 (reply in 874)
874	16.056519	77.88.44.242	172.16.94.216	ICMP	74	Echo (ping) reply id=0x0001, seq=5/1280, ttl=57 (request in 873)
910	17.031803	172.16.94.216	77.88.44.242	ICMP	74	Echo (ping) request id=0x0001, seq=6/1536, ttl=128 (reply in 920)
920	17.070511	77.88.44.242	172.16.94.216	ICMP	74	Echo (ping) reply id=0x0001, seq=6/1536, ttl=57 (request in 910)
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

[Protocols in frame: eth:ethertype:ip:icmp:data]  
[Coloring Rule Name: ICMP]  
[Coloring Rule String: icmp || icmpv6]

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)
  - ....0. .... = IG bit: Globally unique address (factory default)
  - ....0. .... = IG bit: Individual address (unicast)
- Source: AzureWaveTec\_fa:55:e3 (90:e8:68:fa:55:e3)
  - ....0. .... = IG bit: Globally unique address (factory default)
  - ....0. .... = IG bit: Individual address (unicast)
- Type: IPv4 (0x0800)
- [Stream index: 12]

Internet Protocol Version 4, Src: 172.16.94.216, Dst: 77.88.44.242

- 0100 .... = Version: 4
- .... 0101 = Header Length: 20 bytes (5)
- > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 60
- Identification: 0x2904 (10500)
- > 0000. .... = Flags: 0x0

0000 70 18 a7 60 9c e6 90 e8 68 fa 55 e3 0f  
0010 00 3c 29 04 00 00 00 01 8c 8a ac 10 5e  
0020 2c f2 08 00 4d 56 00 01 00 05 61 62 61  
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73  
0040 77 61 62 63 64 65 66 67 68 69



## 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)

→	873	16.024582	172.16.94.216	77.88.44.242	ICMP	74	Echo (ping) request	id=0x0001, seq=5/1280, ttl=128 (reply in 874)
←	874	16.056519	77.88.44.242	172.16.94.216	ICMP	74	Echo (ping) reply	id=0x0001, seq=5/1280, ttl=57 (request in 873)
	910	17.031803	172.16.94.216	77.88.44.242	ICMP	74	Echo (ping) request	id=0x0001, seq=6/1536, ttl=128 (reply in 920)
	920	17.070511	77.88.44.242	172.16.94.216	ICMP	74	Echo (ping) reply	id=0x0001, seq=6/1536, ttl=57 (request in 910)
	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

[Protocols in frame: eth:ethertype:ip:icmp:data]	
[Coloring Rule Name: ICMP]	
[Coloring Rule String: icmp    icmpv6]	
▼ Ethernet II, Src: Cisco_60:9c:e6 (70:18:a7:60:9c:e6), Dst: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3)	
▼ Destination: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3)	
...0. .... = LG bit: Globally unique address (factory default)	
...0. .... = IG bit: Individual address (unicast)	
▼ 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: IPv4 (0x0800)	
[Stream index: 12]	
▼ Internet Protocol Version 4, Src: 77.88.44.242, Dst: 172.16.94.216	
0100 .... = Version: 4	
.... 0101 = Header Length: 20 bytes (5)	
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)	
Total Length: 60	
Identification: 0x2904 (10500)	
> AAA. .... = Flags: 0x00	

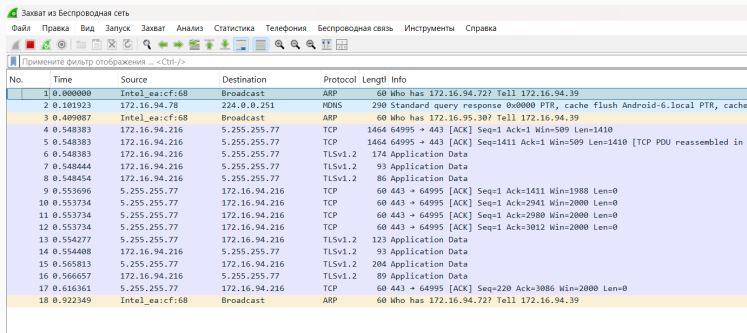
  

0000	90 e8 68 fa 55 e3 70 18 a
0010	00 3c 29 04 00 00 39 01 c
0020	5e d8 00 00 55 56 00 01 6
0030	67 68 69 6a 6b 6c 6d 6e 6
0040	77 61 62 63 64 65 66 67 6

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)

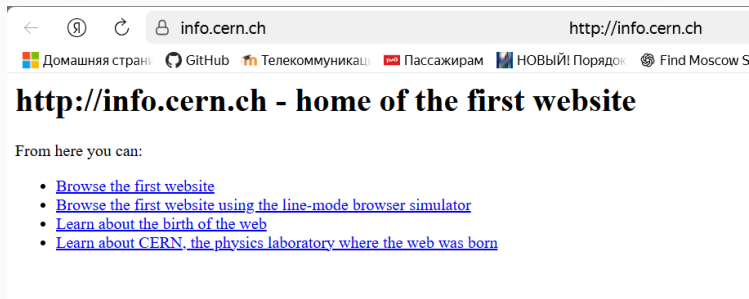


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Intel_ea:cf:68	Broadcast	ARP	60	Who has 172.16.94.72? Tell 172.16.94.39
2	0.101923	172.16.94.78	224.0.0.251	NDNS	290	Standard query response 0x0000 PTR, cache flush Android-6.local PTR, cache
3	0.409087	Intel_ea:cf:68	Broadcast	ARP	60	Who has 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.

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.cer HTTP/1.1

> Frame 561: 369 bytes on wire (2952 bits), 369 bytes captured (2952 bits) on interface \Device\NPF_{7C57B6A3-1A75-4000-8000-000000000000}	0000	90 e8 68 fa 5
> Ethernet II, Src: Cisco_60:9c:e6 (70:18:a7:60:9c:e6), Dst: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3)	0010	01 63 e7 ed 4
> Internet Protocol Version 4, Src: 104.81.99.218, Dst: 172.16.94.216	0020	5e d8 00 50 f
▼ Transmission Control Protocol, Src Port: 80, Dst Port: 65232, Seq: 1, Ack: 255, Len: 315	0030	01 f5 60 ed 0
Source Port: 80	0040	30 34 20 4e 6
Destination Port: 65232	0050	0a 43 6f 6e 7
[Stream index: 37]	0060	70 70 6c 69 6
[Stream Packet Number: 6]	0070	63 72 6c 0d 0
> [Conversation completeness: Incomplete, DATA (15)]	0080	65 64 3a 20 5
[TCP Segment Len: 315]	0090	32 30 32 35 2
Sequence Number: 1 (relative sequence number)	00a0	54 0d 0a 45 5
Sequence Number (raw): 2237714506	00b0	37 65 2d 34 3
[Next Sequence Number: 316 (relative sequence number)]	00c0	6f 6e 74 72 6
Acknowledgment Number: 255 (relative ack number)	00d0	6d 61 78 2d 6
Acknowledgment number (raw): 4193938129	00e0	70 69 72 65 7
0101 .... = Header Length: 20 bytes (5)	00f0	65 70 20 32 3
> Flags: 0x018 (PSH, ACK)	0100	20 47 4d 54 0
Window: 501	0110	20 32 39 20 5
[Calculated window size: 64128]	0120	32 33 3a 35 3
[Window size scaling factor: 128]	0130	63 74 69 6f 6
Checksum: 0x60ed [unverified]	0140	65 0d 0a 41 6
[Checksum Status: Unverified]	0150	2e 36 35 33 6
Urgent Pointer: 0	0160	38 36 33 37 2
> [Timestamps]	0170	0a
> [SEQ/ACK analysis]		
TCP payload (315 bytes)		

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_{7C57B6A3-1A75-4000-8000-000000000000}	0000	70 18 a7 60 9c e6 90 e8 68
>	Ethernet II, Src: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3), Dst: Cisco_60:9c:e6 (78:18:a7:60:9c:e6)	0010	01 26 4e fe 40 00 80 06 d3
>	Internet Protocol Version 4, Src: 172.16.94.216, Dst: 104.81.99.218	0020	63 da fe 00 00 50 f9 fa 69
>	Transmission Control Protocol, Src Port: 65232, Dst Port: 80, Seq: 1, Ack: 1, Len: 254	0030	00 ff 64 4d 00 00 47 45 54
>	Source Port: 65232	0040	65 72 74 47 6c 6f 62 61 6c
>	Destination Port: 80	0050	63 72 6c 20 48 54 54 50 2f
>	[Stream index: 37]	0060	63 68 65 2d 43 6f 6e 74 72
>	[Stream Packet Number: 4]	0070	2d 61 67 65 20 3d 20 33 35
>	[Conversation completeness: Incomplete, DATA (15)]	0080	6e 65 63 74 69 6f 6e 3a 20
>	[TCP Segment Len: 254]	0090	69 76 65 0d 0a 41 63 63 65
>	Sequence Number: 1 (relative sequence number)	00a0	0d 0a 49 66 2d 4d 6f 64 69
>	Sequence Number (raw): 4193937875	00b0	6e 63 65 3a 20 57 65 64 2c
>	[Next Sequence Number: 255 (relative sequence number)]	00c0	20 32 30 32 35 20 32 30 3a
>	Acknowledgment Number: 1 (relative ack number)	00d0	4d 54 0d 0a 49 66 2d 4e 6f
>	Acknowledgment number (raw): 2237714506	00e0	68 3a 20 22 36 38 64 34 35
>	0101 ... = Header Length: 20 bytes (5)	00f0	22 0d 0a 55 73 65 72 2d 41
>	Flags: 0x018 (PSH, ACK)	0100	69 63 72 6f 73 6f 66 74 2d
>	Window: 255	0110	50 49 2f 31 30 2e 30 0d 0a
>	[Calculated window size: 65280]	0120	72 6c 33 2e 64 69 67 69 63
>	[Window size scaling factor: 256]	0130	0d 0a 0d 0a
>	Checksum: 0x644d [unverified]		
>	[Checksum Status: Unverified]		
>	Urgent Pointer: 0		
>	[Timestamps]		
>	[SEQ/ACK analysis]		
>	TCP payload (254 bytes)		
>	Hypertext Transfer Protocol		

Рис. 19: Анализ TCP (ответ)

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

693	46.057470	37.18.92.5	172.16.94.216	DNS	540 Standard query response 0xa181 A v10.events.data.microsoft.com CNAME win-global-asm
692	46.053078	172.16.94.216	37.18.92.5	DNS	89 Standard query 0xa181 A v10.events.data.microsoft.com
554	39.363036	193.232.218.194	172.16.94.216	DNS	212 Standard query response 0x4159 A crl3.digicert.com CNAME crl.edge.digicert.com CNAME
552	39.347310	37.18.92.5	172.16.94.216	DNS	533 Standard query response 0x4159 A crl3.digicert.com CNAME crl.edge.digicert.com CNAME
551	39.325490	172.16.94.216	193.232.218.194	DNS	77 Standard query 0x4159 A crl3.digicert.com
550	39.289182	172.16.94.216	37.18.92.5	DNS	77 Standard query 0x4159 A crl3.digicert.com
398	27.857349	37.18.92.5	172.16.94.216	DNS	149 Standard query response 0x6f86 HTTPS suggest.sso.dzen.ru SOA ns1.mail.ru
391	27.845653	37.18.92.5	172.16.94.216	DNS	263 Standard query response 0xf268 A dzen.ru A 5.61.23.39 A 185.180.200.2 A 83.222.28.15
390	27.845653	37.18.92.5	172.16.94.216	DNS	145 Standard query response 0x9d92 HTTPS suggest.dzen.ru SOA ns1.mail.ru
389	27.845653	37.18.92.5	172.16.94.216	DNS	247 Standard query response 0xd6ec A suggest.dzen.ru A 87.250.254.106 NS ns1.mail.ru NS n
384	27.845653	37.18.92.5	172.16.94.216	DNS	255 Standard query response 0x0a8a A suggest.sso.dzen.ru A 87.250.254.106 NS ns2.mail.ru
376	27.827637	37.18.92.5	172.16.94.216	DNS	137 Standard query response 0xec7f HTTPS dzen.ru SOA ns1.mail.ru
374	27.786857	172.16.94.216	37.18.92.5	DNS	91 Standard query 0x6f86 HTTPS suggest.sso.dzen.ru
> Frame 693: 540 bytes on wire (4320 bits), 540 bytes captured (4320 bits) on interface \Device\NPF_{7C57B6A3-1A75-4000-8000-000000000000}					0020 5e d8 00 35 f5 f6 01 fa bb e2 a1 81 81 i
> Ethernet II, Src: Cisco_60:9c:e6 (70:18:a7:60:9c:e6), Dst: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3)					0030 00 03 00 04 00 08 03 76 31 30 06 65 76 i
> Internet Protocol Version 4, Src: 37.18.92.5, Dst: 172.16.94.216					0040 73 04 64 61 74 61 09 6d 69 63 72 6f 73 i
User Datagram Protocol, Src Port: 53, Dst Port: 62966					0050 03 63 6f 6d 00 00 01 00 01 c0 0c 00 05 i
Source Port: 53					0060 00 00 01 00 38 23 77 69 6e 2d 67 6c 6f i
Destination Port: 62966					0070 2d 61 73 69 6d 6f 76 2d 6c 65 61 66 73 i
Length: 506					0080 65 6e 74 73 2d 64 61 74 61 0e 74 72 61 i
Checksum: 0xbbe2 [unverified]					0090 63 6d 61 6e 61 67 65 72 03 6a 65 74 00 i
[Checksum Status: Unverified]					00a0 05 00 01 00 00 00 0b 00 2c 10 6f 6e 65 i
[Stream index: 37]					00b0 6f 6e 70 72 64 63 75 73 30 34 09 63 65 i
[Stream Packet Number: 2]					00c0 61 6c 75 73 08 63 6c 6f 75 64 61 70 70 i
[Timestamps]					00d0 75 72 65 c0 26 c0 7f 00 01 00 01 00 00 i
[Time since first frame: 0.004392000 seconds]					00e0 04 34 b6 8f d0 c0 90 00 02 00 01 00 00 i
[Time since previous frame: 0.004392000 seconds]					00f0 18 07 6e 73 34 2d 32 30 31 09 61 7a 75 i
UDP payload (498 bytes)					0100 64 6e 73 04 69 6e 66 6f 00 c0 90 00 02 i
					0110 00 0d 25 00 17 07 6e 73 33 2d 32 30 31 i
					0120 75 72 65 2d 64 6e 73 03 6f 72 67 00 c0 i
					0130 00 01 00 00 0d 25 00 14 07 6e 73 31 2d i

Рис. 20: Анализ UDP (запрос)

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

693	46.057470	37.18.92.5	172.16.94.216	DNS	540 Standard query response 0xa181 A v10.events.data.microsoft.com CNAME win-g
692	46.053078	172.16.94.216	37.18.92.5	DNS	89 Standard query 0xa181 A v10.events.data.microsoft.com
554	39.363036	193.232.218.194	172.16.94.216	DNS	212 Standard query response 0x4159 A crl3.digicert.com CNAME crl.edge.digicert
552	39.347310	37.18.92.5	172.16.94.216	DNS	533 Standard query response 0x4159 A crl3.digicert.com CNAME crl.edge.digicert
551	39.325490	172.16.94.216	193.232.218.194	DNS	77 Standard query 0x4159 A crl3.digicert.com
550	39.289182	172.16.94.216	37.18.92.5	DNS	77 Standard query 0x4159 A crl3.digicert.com
398	27.857349	37.18.92.5	172.16.94.216	DNS	149 Standard query response 0x6f86 HTTPS suggest.sso.dzen.ru SOA ns1.mail.ru
391	27.845653	37.18.92.5	172.16.94.216	DNS	263 Standard query response 0xf268 A dzen.ru A 5.61.23.39 A 185.180.200.2 A 83
390	27.845653	37.18.92.5	172.16.94.216	DNS	145 Standard query response 0x9d92 HTTPS suggest.dzen.ru SOA ns1.mail.ru
389	27.845653	37.18.92.5	172.16.94.216	DNS	247 Standard query response 0xd6ec A suggest.dzen.ru A 87.250.254.106 NS ns1.m
384	27.845653	37.18.92.5	172.16.94.216	DNS	255 Standard query response 0x0a8a A suggest.sso.dzen.ru A 87.250.254.106 NS n
376	27.827637	37.18.92.5	172.16.94.216	DNS	137 Standard query response 0xec7f HTTPS dzen.ru SOA ns1.mail.ru
374	27.786857	172.16.94.216	37.18.92.5	DNS	91 Standard query 0x6f86 HTTPS suggest.sso.dzen.ru
> Frame 692: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface \Device\NPF_{7C57B6A3-1A75-461D-8000-000000000000}					0000 70 18 a7 60 9c e6 90 e8 68 f0
> Ethernet II, Src: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3), Dst: Cisco_60:9c:e6 (70:18:a7:60:9c:e6)					0010 00 4b 60 68 00 00 80 11 4e 3e
> Internet Protocol Version 4, Src: 172.16.94.216, Dst: 37.18.92.5					0020 5c 05 f5 f6 00 35 00 37 3b 31
v User Datagram Protocol, Src Port: 62966, Dst Port: 53					0030 00 00 00 00 00 00 63 76 31 3e
Source Port: 62966					0040 73 04 64 61 74 61 09 6d 69 6f
Destination Port: 53					0050 03 63 6f 6d 00 00 01 00 01
Length: 55					
Checksum: 0x3b31 [unverified]					
[Checksum Status: Unverified]					
[Stream index: 37]					
[Stream Packet Number: 1]					
v [Timestamps]					
[Time since first frame: 0.000000000 seconds]					
[Time since previous frame: 0.000000000 seconds]					
UDP payload (47 bytes)					
> Domain Name System (query)					

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)

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

3092	63.830678	172.16.94.216	173.194.73.198	QUIC	76 Protected Payload (KP0), DCID=f8a9378badb7bea3
3089	63.819914	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3088	63.819613	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3087	63.819613	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3086	63.819557	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3085	63.819557	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3084	63.819159	173.194.73.198	172.16.94.216	QUIC	1003 Protected Payload (KP0)
3083	63.819159	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
Frame 3092: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface \Device\NPF_{7C57B6A3-1A75-461...}					
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: 173.194.73.198					
User Datagram Protocol, Src Port: 62198, Dst Port: 443					
QUIC IETF					
QUIC Connection information					
[Connection Number: 0]					
[Packet Length: 34]					
QUIC Short Header DCID=f8a9378badb7bea3					
0... .... = Header Form: Short Header (0)					
.1.. .... = Fixed Bit: True					
..0. .... = Spin Bit: False					
Destination Connection ID: f8a9378badb7bea3					
Remaining Payload: d6e43ad86a110708f3b1b89052457c50edfdf56ae83f6b9d3d					

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: 3fcea12c11cd... Аналогично клиентскому пакету, полезная нагрузка сервера также является “Protected Payload” (Защищенной) и зашифрована. (рис. 23)

# Анализ протоколов транспортного уровня в Wireshark

3089	63.819914	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3088	63.819613	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3087	63.819613	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3086	63.819557	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3085	63.819557	173.194.73.198	172.16.94.216	QUIC	1292 Protected Payload (KP0)
3084	63.819159	173.194.73.198	172.16.94.216	QUIC	1003 Protected Payload (KP0)
3083	63.819159	173.194.73.198	172.16.94.216	QUIC	1003 Protected Payload (KP0)

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 [...]:	3fcea12c11c43be28dfbb8ba073cdfb9edfe90082cc44a85085309ced00b8034443b5f3420bc3a1d7bf57

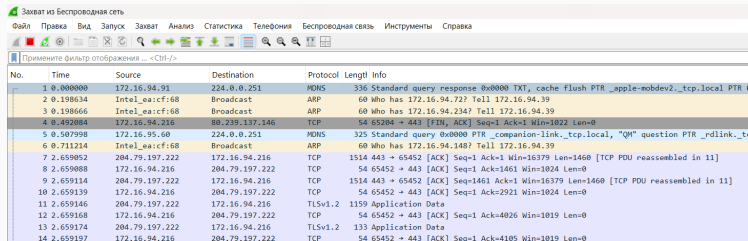
  

0020	5e d8 01 bb f
0030	c4 3b e2 8d f
0040	c4 4a 85 08 5
0050	bc 3a 1d 7b f
0060	08 41 2e 3f 6
0070	ed ba dc f5 f
0080	3f 4f bf eb b
0090	9e b7 5e 31 e
00a0	9e 77 71 2b a
00b0	49 59 84 04 b
00c0	8f b2 39 c7 8
00d0	a5 c7 d7 41 9
00e0	68 1b d3 79 b
00f0	06 02 0e f9 7
0100	f0 ad 7d ed 3
0110	f6 8f e1 b2 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 network protocol analyzer interface. The top menu bar includes 'Захват из Беспроводная сеть' (Capture from Wireless Network), 'Файл' (File), 'Правка' (Edit), 'Вид' (View), 'Запуск' (Start), 'Захват' (Capture), 'Анализ' (Analyze), 'Статистика' (Statistics), 'Телефония' (Telephony), 'Беспроводная связь' (Wireless), 'Инструменты' (Tools), and 'Справка' (Help). Below the menu is a toolbar with icons for various functions. A filter bar shows 'Примените фильтр отображения ... <Ctrl-/>'. The main packet list pane displays 14 captured packets with columns for No., Time, Source, Destination, Protocol, Length, and Info. The packets include MDNS, ARP, and TCP traffic.

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_ea:cf:68	Broadcast	ARP	60	Who has 172.16.94.72? Tell 172.16.94.39
3	0.198666	Intel_ea:cf:68	Broadcast	ARP	60	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=0
5	0.507998	172.16.95.60	224.0.0.251	MDNS	325	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" question PTR _rdlink._t
6	0.711214	Intel_ea:cf:68	Broadcast	ARP	60	Who has 172.16.94.148? Tell 172.16.94.39
7	2.659052	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.

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

1400	7.043388	172.16.94.216	122.189.32.168	TCP	66	50451 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM	
1429	7.272378	122.189.32.168	172.16.94.216	TCP	66	80 → 50451 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=	
1430	7.272437	172.16.94.216	122.189.32.168	TCP	54	50451 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0	
1431	7.272794	172.16.94.216	122.189.32.168	HTTP	488	GET /knowledge/2022/0731/FktNcluXYIN2MD8ieGs5zx5X3a2 HTTP/1.1	
1432	7.276121	111.6.201.146	172.16.94.216	TCP	66	80 → 50450 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=	
1433	7.276159	172.16.94.216	111.6.201.146	TCP	54	50450 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0	
1434	7.276376	172.16.94.216	111.6.201.146	HTTP	488	GET /knowledge/2022/0731/FrAbg_kxVspqcnHKWF5XzGb7zEjp HTTP/1.1	
1443	7.480627	122.189.32.168	172.16.94.216	TCP	66	[TCP Out-Of-Order] 80 → 50451 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS	

> Frame 1400: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{7C57B6A3-1A75-4000-8000-000000000000} on interface 0	0000	70 18 a7 60 9c e6 90 e8 68
> Ethernet II, Src: AzureWaveTec_fa:55:e3 (90:e8:68:fa:55:e3), Dst: Cisco_60:9c:e6 (70:18:a7:60:9c:e6)	0010	00 34 f0 ed 40 00 80 06 63
> Internet Protocol Version 4, Src: 172.16.94.216, Dst: 122.189.32.168	0020	20 a8 c5 13 00 50 b6 f2 59
> Transmission Control Protocol, Src Port: 50451, Dst Port: 80, Seq: 0, Len: 0	0030	ff ff f2 70 00 00 02 04 05
Source Port: 50451	0040	04 02
Destination Port: 80		
[Stream index: 251]		
[Stream Packet Number: 1]		
> [Conversation completeness: Incomplete, DATA (15)]		
[TCP Segment Len: 0]		
Sequence Number: 0 (relative sequence number)		
Sequence Number (raw): 3069336059		
[Next Sequence Number: 1 (relative sequence number)]		
Acknowledgment Number: 0		
Acknowledgment number (raw): 0		
1000 .... = Header Length: 32 bytes (8)		
> Flags: 0x002 (SYN)		
000. .... = Reserved: Not set		
...0 .... = Accurate ECN: Not set		

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

1428	7.262008	172.16.94.216	111.6.201.146	TCP	66	50454 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
1429	7.272378	122.189.32.168	172.16.94.216	TCP	66	80 → 50451 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=1024
1430	7.272437	172.16.94.216	122.189.32.168	TCP	54	50451 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1431	7.272794	172.16.94.216	122.189.32.168	HTTP	488	GET /knowledge/2022/0731/FktNcluXYIM2dD8ieGs5zx5X3a2 HTTP/1.1
1432	7.276121	111.6.201.146	172.16.94.216	TCP	66	80 → 50450 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=1024
1433	7.276159	172.16.94.216	111.6.201.146	TCP	54	50450 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1434	7.276376	172.16.94.216	111.6.201.146	HTTP	488	GET /knowledge/2022/0731/FrAbg_kxVspqcnHKWf5XzGb7sEjp HTTP/1.1
1436	7.371056	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50448 → 53 [FIN, ACK] Seq=35 Ack=385 Win=65024 Len=0
1437	7.371098	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50449 → 53 [FIN, ACK] Seq=35 Ack=108 Win=65280 Len=0
1438	7.417988	172.16.94.216	37.18.92.5	TCP	54	50395 → 53 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1439	7.466151	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50452 → 53 [FIN, ACK] Seq=42 Ack=584 Win=64768 Len=0
1440	7.466208	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50453 → 53 [FIN, ACK] Seq=42 Ack=234 Win=65280 Len=0
1443	7.480627	122.189.32.168	172.16.94.216	TCP	66	[TCP Out-Of-Order] 80 → 50451 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 S

[Stream index: 251]  
[Stream Packet Number: 2]  
> [Conversation completeness: Incomplete, DATA (15)]  
[TCP Segment Len: 0]  
Sequence Number: 0 (relative sequence number)  
Sequence Number (raw): 2590184539  
[Next Sequence Number: 1 (relative sequence number)]  
Acknowledgment Number: 1 (relative ack number)  
Acknowledgment number (raw): 3069336060  
1000 .... = Header Length: 32 bytes (8)  
▼ **Flags: 0x012 (SYN, ACK)**  
000. .... = Reserved: Not set  
...0 .... = Accurate ECN: Not set  
.... 0... = Congestion Window Reduced: Not set  
.... .0.. = ECN-Echo: Not set  
.... .0. .... = Urgent: Not set  
.... ..1 .... = Acknowledgment: Set  
.... .... 0... = Push: Not set  
.... .... .0.. = Reset: Not set

0000 90 e8 68 fa 55 e3 70 18 a7 60 9c 1  
0010 00 34 00 00 00 00 2d 06 a7 76 7a 1  
0020 5e d8 00 50 c5 13 9a 63 14 5b b6  
0030 ff ff 43 b3 00 00 02 04 05 a0 01 1  
0040 03 0a

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

1428	7.262008	172.16.94.216	111.6.201.146	TCP	66	50454 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
1429	7.272378	122.189.32.168	172.16.94.216	TCP	66	80 → 50451 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=1024
1430	7.272437	172.16.94.216	122.189.32.168	TCP	54	50451 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1431	7.272794	172.16.94.216	122.189.32.168	HTTP	488	GET /knowledge/2022/0731/FktNcluXYIW2HDd8ieGs5zxSX3a2 HTTP/1.1
1432	7.276121	111.6.201.146	172.16.94.216	TCP	66	80 → 50450 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=1024
1433	7.276159	172.16.94.216	111.6.201.146	TCP	54	50450 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1434	7.276376	172.16.94.216	111.6.201.146	HTTP	488	GET /knowledge/2022/0731/FrAbg_kxVspqcnHKMF5XzGb7sEjp HTTP/1.1
1436	7.371056	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50448 → 53 [FIN, ACK] Seq=35 Ack=385 Win=65024 Len=0
1437	7.371098	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50449 → 53 [FIN, ACK] Seq=35 Ack=108 Win=65280 Len=0
1438	7.417988	172.16.94.216	37.18.92.5	TCP	54	50395 → 53 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1439	7.466151	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50452 → 53 [FIN, ACK] Seq=42 Ack=584 Win=64768 Len=0
1440	7.466208	172.16.94.216	37.18.92.5	TCP	54	[TCP Retransmission] 50453 → 53 [FIN, ACK] Seq=42 Ack=234 Win=65280 Len=0
1441	7.480627	122.189.32.168	172.16.94.216	TCP	66	[TCP Out-Of-Order] 80 → 50451 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440

[Stream index: 251]  
[Stream Packet Number: 3]  
> [Conversation completeness: Incomplete, DATA (15)]  
[TCP Segment Len: 0]  
Sequence Number: 1 (relative sequence number)  
Sequence Number (raw): 3069336060  
[Next Sequence Number: 1 (relative sequence number)]  
Acknowledgment Number: 1 (relative ack number)  
Acknowledgment number (raw): 2590184540  
0101 .... = Header Length: 20 bytes (5)  
v Flags: 0x010 (ACK)  
000. .... = Reserved: Not set  
...0 .... = Accurate ECN: Not set  
.... 0.... = Congestion Window Reduced: Not set  
.... .0.. = ECN-Echo: Not set  
.... .0. .... = Urgent: Not set  
.... ..1 .... = Acknowledgment: Set  
.... ....0... = Push: Not set  
.... ....0.. = Reset: Not set  
.... ....0. .... = Same-Network: Not set

0000 70 18 a7 60 9c e6 90 e8 68 fa 5  
0010 00 28 f0 ee 40 00 80 06 63 93 a  
0020 20 a8 c5 13 00 50 b6 f2 59 fc 9  
0030 00 ff 83 75 00 00

Рис. 27: TCP handshake шаг 3

## Вывод

---

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