

# **Лабораторная работа №3**

**Анализ трафика в Wireshark**

**Демидова Екатерина Алексеевна**

# **Содержание**

<b>1 Цель работы</b>	<b>4</b>
<b>2 Задание</b>	<b>5</b>
<b>3 Выполнение лабораторной работы</b>	<b>6</b>
3.1 MAC-адресация . . . . .	6
3.2 Анализ кадров канального уровня в Wireshark . . . . .	9
3.3 Анализ протоколов транспортного уровня в Wireshark . . . . .	16
3.4 Анализ handshake протокола TCP в Wireshark . . . . .	22
<b>4 Выводы</b>	<b>27</b>

# Список иллюстраций

3.1	Вывод команды ifconfig . . . . .	6
3.2	Вывод информации о конкретном сетевом интерфейсе‘ . . . . .	6
3.3	Выключение сетевого интерфейса и демонстрация опции -a . . . . .	7
3.4	Использование опции -s . . . . .	8
3.5	Демонстрация MAC-адреса . . . . .	8
3.6	Демонстрация производителя сетевого оборудования . . . . .	9
3.7	Захват трафика в wireshark . . . . .	10
3.8	Определение IP-адреса устройства и шлюза по умолчанию . . . . .	10
3.9	Пингование шлюза по умолчанию . . . . .	11
3.10	Кадр ICMP - эхо-запрос . . . . .	11
3.11	Кадр ICMP - эхо-ответ . . . . .	12
3.12	Кадр ARP . . . . .	13
3.13	Пингование сайта wikipedia.org . . . . .	13
3.14	Пингование сайта wikipedia.org. Кадр ICMP - эхо-запрос . . . . .	14
3.15	Пингование сайта wikipedia.org. Кадр ICMP - эхо-ответ . . . . .	14
3.16	Пингование сайта wikipedia.org. Кадр ARP - запрос . . . . .	15
3.17	Пингование сайта wikipedia.org. Кадр ARP - ответ . . . . .	15
3.18	Запрос HTTP по протоколу TCP . . . . .	16
3.19	Ответ HTTP по протоколу TCP . . . . .	17
3.20	Раздел HTP . . . . .	17
3.21	Раздел Line-based text data . . . . .	18
3.22	Запрос dns по протоколу UDP . . . . .	19
3.23	Ответ DNS по протоколу UDP . . . . .	19
3.24	Запрос QUIC по протоколу UDP . . . . .	20
3.25	Ответ QUIC по протоколу UDP . . . . .	21
3.26	Вкладка QUIK IETF в случае запроса quik . . . . .	21
3.27	Вкладка QUIK IETF в случае ответа quik . . . . .	22
3.28	Первая ступень handshake протокола TCP . . . . .	23
3.29	Вторая ступень handshake протокола TCP . . . . .	24
3.30	Третья ступень handshake протокола TCP . . . . .	25
3.31	График потока . . . . .	26

# **1 Цель работы**

Изучение посредством Wireshark кадров Ethernet, анализ PDU протоколов транспортного и прикладного уровней стека TCP/IP.

## **2 Задание**

1. MAC-адресация
  1. Изучение возможностей команды ipconfig для ОС типа Windows (ifconfig для систем типа Linux).
  2. Определение MAC-адреса устройства и его типа.
2. Анализ кадров канального уровня в Wireshark
  1. Установить на домашнем устройстве Wireshark.
  2. С помощью Wireshark захватить и проанализировать пакеты ARP и ICMP в части кадров канального уровня
3. Анализ протоколов транспортного уровня в Wireshark. С помощью Wireshark захватить и проанализировать пакеты HTTP, DNS в части заголовков и информации протоколов TCP, UDP, QUIC.
4. Анализ handshake протокола TCP в Wireshark. С помощью Wireshark проанализировать handshake протокола TCP.

# 3 Выполнение лабораторной работы

## 3.1 MAC-адресация

С помощью команды `ifconfig` выведем информацию о текущем сетевом соединении (рис. 3.1).

```
eademidova@Demidrol: ~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Локальная петля (Loopback))
            RX packets 34787 bytes 2798391 (2.7 MB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 34787 bytes 2798391 (2.7 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp0s20f3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.147 netmask 255.255.255.0 broadcast 192.168.2.255
        inet6 fe80::17ff:4909:39e4:9973 prefixlen 64 scopeid 0x20<link>
            ether e8:f4:08:c4:e3:ca txqueuelen 1000 (Ethernet)
            RX packets 824722 bytes 1017809698 (1.0 GB)
            RX errors 0 dropped 80 overruns 0 frame 0
            TX packets 253712 bytes 56982702 (56.9 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

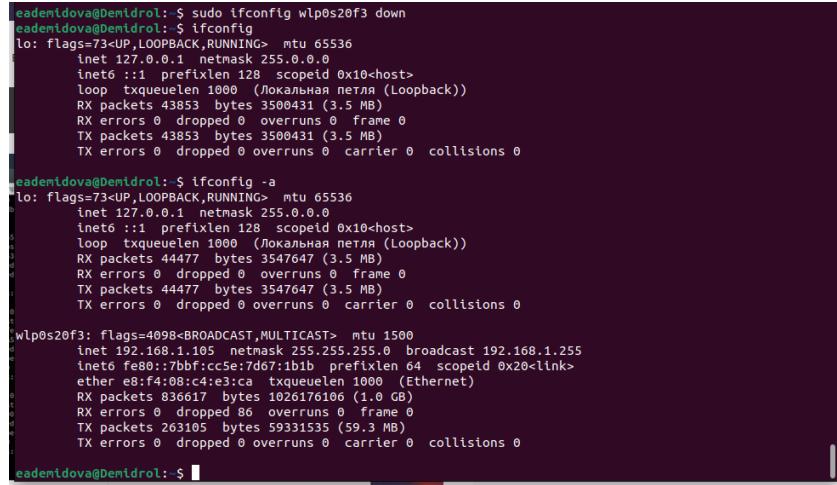
Рис. 3.1: Вывод команды `ifconfig`

Так же добавив название сетевого интерфейса можно вывести информацию только о нём(рис. 3.2).

```
eademidova@Demidrol: ~$ ifconfig wlp0s20f3
wlp0s20f3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.147 netmask 255.255.255.0 broadcast 192.168.2.255
        inet6 fe80::17ff:4909:39e4:9973 prefixlen 64 scopeid 0x20<link>
            ether e8:f4:08:c4:e3:ca txqueuelen 1000 (Ethernet)
            RX packets 825254 bytes 1017952582 (1.0 GB)
            RX errors 0 dropped 82 overruns 0 frame 0
            TX packets 254436 bytes 57254208 (57.2 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Рис. 3.2: Вывод информации о конкретном сетевом интерфейсе

Теперь выключим сетевой интерфейс добавив команду `down`(с помощью команды `ip` его можно включить). При использовании команды `ifconfig` его не видно, так как это сетевое соединение больше не активно, но можно использовать опцию `-a`, которая показывает все сетевые интерфейсы, даже не активные(рис. 3.3).



```
eademidova@Dentidrol: $ sudo ifconfig wlp0s20f3 down
eademidova@Dentidrol: $ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopcid 0x10<host>
            loop txqueuelen 1000 (Локальная петля (Loopback))
            RX packets 43853 bytes 3500431 (3.5 MB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 43853 bytes 3500431 (3.5 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eademidova@Dentidrol: $ ifconfig -a
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopcid 0x10<host>
            loop txqueuelen 1000 (Локальная петля (Loopback))
            RX packets 44477 bytes 3547647 (3.5 MB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 44477 bytes 3547647 (3.5 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp0s20f3: flags=4098<POINTOPOINT,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.1.105 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::7bbf:cc5e:7d67:1b1b prefixlen 64 scopcid 0x20<link>
            ether e8:f4:08:c4:e3:ca txqueuelen 1000 (Ethernet)
            RX packets 836617 bytes 1026176106 (1.0 GB)
            RX errors 0 dropped 86 overruns 0 frame 0
            TX packets 263105 bytes 59331535 (59.3 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eademidova@Dentidrol: $
```

Рис. 3.3: Выключение сетевого интерфейса и демонстрация опции `-a`

Также можно использовать опцию `-s`, с помощью которой выводится краткий список интерфейсов(рис. 3.4). При таком выводе можно увидеть информацию только о следующих параметрах:

- MTU – Максимальное число байтов в пакете
- RX-OK – Пакеты, принятые без ошибок
- RX-ERR – Пакеты, принятые с ошибками
- RX-DRR – Пропавшие пакеты
- RX-OVR – Ошибки из-за превышения скорости
- TX-OK – Пакеты, переданные без ошибок
- TX-ERR – Пакеты, переданные с ошибками
- TZ-DRR – Пакеты, потерянные при передаче
- TX-OVR – Пакеты, которые не смогли передать

- Flags – Характеристики интерфейса
  - A – принимает пакеты в случае многоадресной передачи
  - B – принимает широковещательные пакеты
  - D – отладка включена
  - L – закольцовывающий интерфейс
  - M – изменяется динамически (переадресация)
  - N – без обработки завершителей пакетов
  - O – протокол преобразования адресов выключен
  - P – интерфейс “точка-точка”
  - R – интерфейс работает
  - U – интерфейс активизирован

```
eademidova@Demidrol:~$ ifconfig -s
Iface      MTU     RX-OK RX-ERR RX-DRP RX-OVR     TX-OK TX-ERR TX-DRP TX-OVR Flg
lo       65536    58233     0     0 0      58233     0     0     0 LRU
wlp0s20f  1500   851363     0    86 0      278785     0     0     0 BMRU
eademidova@Demidrol:~$
```

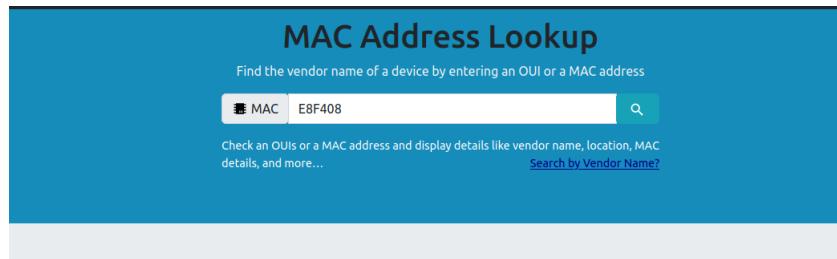
Рис. 3.4: Использование опции `-s`

MAC-адрес сетевого интерфейса можно увидеть так же с помощью команды `ifconfig`, посмотрев на поле `ether`, но MAC-адрес виртуального интерфейса не показывается. Можно использовать команду `ip -brief link`, чтобы увидеть только MAC-адреса (рис. 3.5). Видно, что у виртуального интерфейса все биты в нуле, а у второго сетевого интерфейса MAC-адрес `e8:f4:08:c4:e3:ca`.

```
eademidova@Demidrol:~$ ip -brief link
lo        UNKNOWN      00:00:00:00:00:00 <LOOPBACK,UP,LOWER_UP>
wlp0s20f3   DOWN       e8:f4:08:c4:e3:ca <BROADCAST,MULTICAST>
eademidova@Demidrol:~$
```

Рис. 3.5: Демонстрация MAC-адреса

Первые три байта `e8:f4:08` в этом адресе соответствуют индентификатору производителя, в Интернете можно найти, какой компании соответствует этот индентификатор(рис. 3.6). Последние три байта `c4:e3:ca` индентифицируют сетевой интерфейс.



### Intel Corporate

[Vendor](#) [Details](#)

OUI: E8:F4:08

Vendor name: [Intel Corporate](#) ↗

Address:

Lot 8  
Jalan Hi-Tech 2/3  
Kulim Kedah 09000  
MY.

Assignment Type MA-L

Mac Address Block Large (previously named OUI). Number of  
addresses  $2^{24}$  (~16 Million)

Initial registration: 23 October 2020



Рис. 3.6: Демонстрация производителя сетевого оборудования

Возьмем первый байт e8 и переведём его в двоичную систему. Получим 11101000. Так как последний бит ноль, то адрес является индивидуальным. А так как предпоследний бит ноль, то адрес глобально администрируемый.

## 3.2 Анализ кадров канального уровня в Wireshark

Запустим wireshark и начнем захват трафика (рис. 3.7).

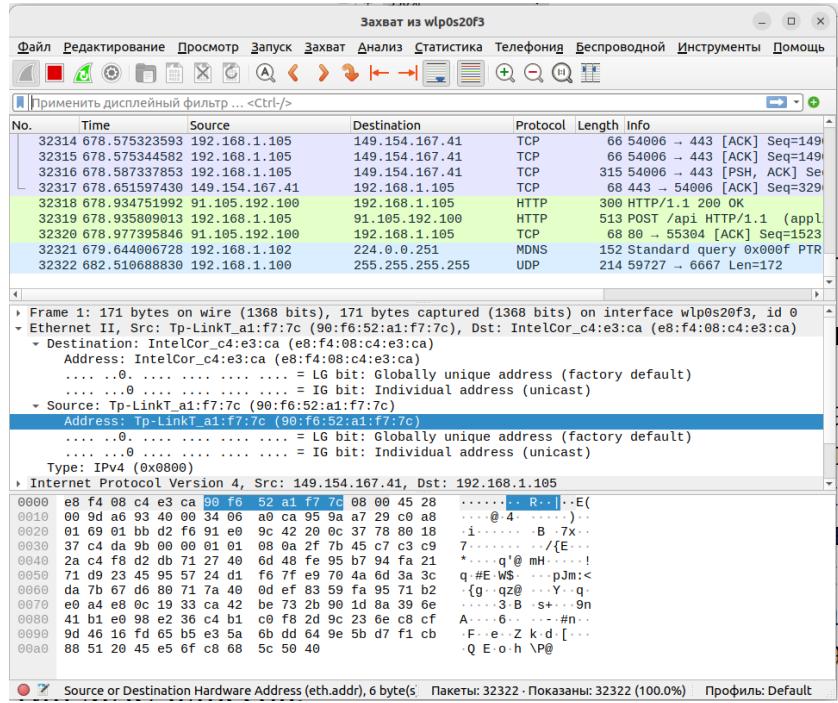


Рис. 3.7: Захват трафика в wireshark

С помощью команды `ifconfig` определим IP-адрес устройства – 192.168.1.105.

Шлюз по умолчанию на моём устройстве не отображается с помощью этой команды, поэтому я использовала также команду `ip route`, шлюз по умолчанию – 192.168.1.1(рис. 3.8).

```
wlp0s20f3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.105 brd 192.168.1.255 netmask 255.255.255.0
        broadcast 192.168.1.255
        inet6 fe80::7bfcc5e7d67:1b1b brd fe80::ff:fe1b:1b1b/16 scopeid 260<link>
            ether e8:f4:08:c4:e3:ca txqueuelen 1000 (Ethernet)
            RX packets 958063 bytes 1118510172 (1.1 GB)
            RX errors 0 dropped 88 overruns 0 frame 0
            TX packets 367820 bytes 93223431 (93.2 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
eademidova@Demidrol: $ ip route
default via 192.168.1.1 dev wlp0s20f3 proto dhcp metric 600
192.168.1.0/24 dev wlp0s20f3 proto kernel scope link src 192.168.1.105 metric 600
eademidova@Demidrol: $
```

Рис. 3.8: Определение IP-адреса устройства и шлюза по умолчанию

С помощью команды `ping 192.168.1.1` пропингуем шлюз по умолчанию (рис. 3.9).

```

eademidovag@Demidrol: $ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=3.57 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=8.40 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=3.54 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=64 time=3.34 ms
^C
--- 192.168.1.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 3.335/4.712/8.403/2.132 ms
eademidovag@Demidrol:-

```

Рис. 3.9: Пингование шлюза по умолчанию

Затем остановим захват трафика wireshark и пропишем фильтр arp or icmp. В списке пакетов отобразились только пакеты ARP и ICMP, в частности пакеты, которые были сгенерированы с помощью команды ping (рис. 3.10). Изучим эхо-запрос и эхо-ответ ICMP в программе Wireshark. На панели списка пакетов (верхний раздел) выберем первый указанный кадр ICMP — эхо-запрос (3.10). Изучим информацию на панели сведений о пакете в средней части экрана. Длинна кадра равняется 98 байт, заголовок Ethernet первые 14 байт кадра, кадр относится к типу Ethernet II. MAC-адрес шлюза — это первые 6 байт заголовка Ethernet, а MAC-адрес источника — следующие 6 байт заголовка Ethernet, оба MAC-адреса являются индивидуальными и глобально администрируемыми.

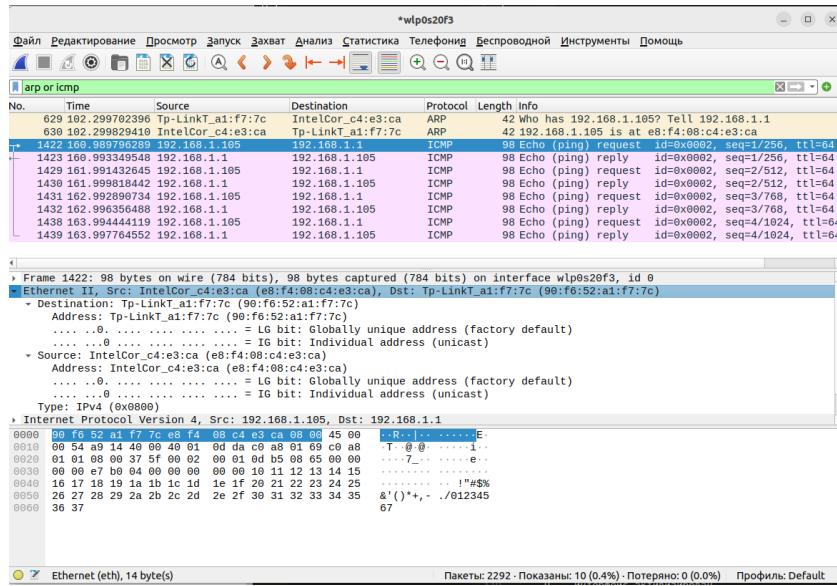


Рис. 3.10: Кадр ICMP - эхо-запрос

На панели списка пакетов (верхний раздел) выберем второй указанный кадр

ICMP — эхо-ответ (3.11). Изучим информацию на панели сведений о пакете в средней части экрана. Длина кадра равняется 98 байт, заголовок Ethernet первые 14 байт кадра, кадр относится к типу Ethernet II. MAC-адрес пункта назначения – это первые 6 байт заголовка Ethernet, а MAC-адрес шлюза(источника) - следующие 6 байт заголовка Ethernet, оба MAC-адреса являются индивидуальными и глобально администрируемыми.

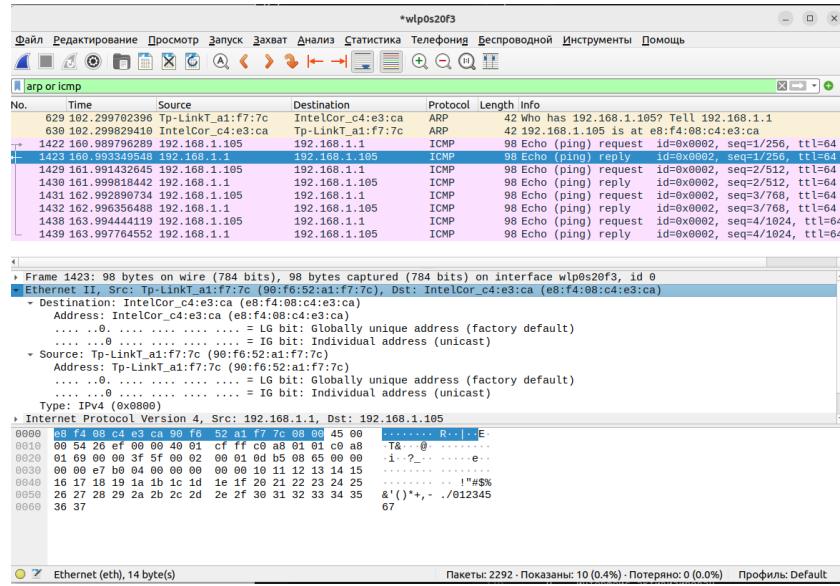


Рис. 3.11: Кадр ICMP - эхо-ответ

На панели списка пакетов (верхний раздел) выберем второй указанный кадр ARP (3.12). Изучим информацию на панели сведений о пакете в средней части экрана. Длина кадра равняется 42 байт, заголовок Ethernet первые 14 байт кадра, кадр относится к типу Ethernet II. MAC-адрес пункта назначения – это первые 6 байт заголовка Ethernet, а MAC-адрес источника - следующие 6 байт заголовка Ethernet, оба MAC-адреса являются индивидуальными и глобально администрируемыми. Также в заголовке Ethernet последние два байта обозначают вложенным пакетом типа ARP.

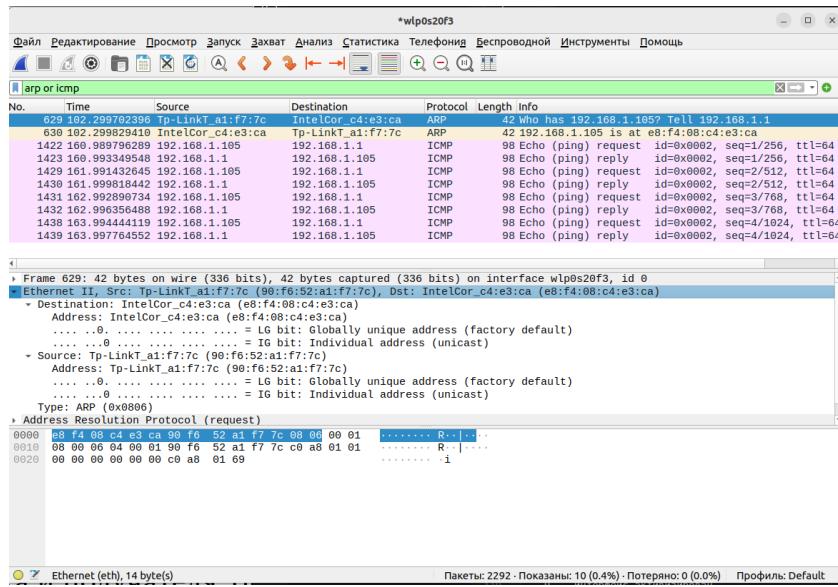


Рис. 3.12: Кадр ARP

Теперь начнём новый процесс захвата трафика и пропингуем сайт wikipedia.org (3.13).

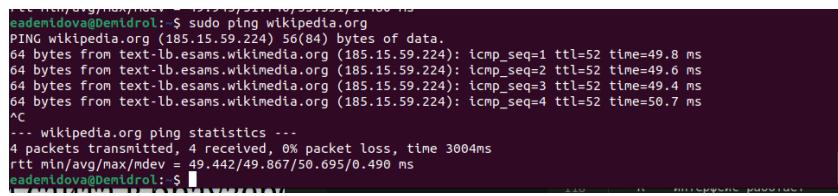


Рис. 3.13: Пингование сайта wikipedia.org

Изучим эхо-запрос и эхо-ответ ICMP в программе Wireshark (3.14, ??). Изучим информацию на панели сведений о пакете в средней части экрана. В обоих случаях длина кадра равняется 98 байт, заголовок Ethernet первые 14 байт кадра и кадр относится к типу Ethernet II. В случае эхо-запроса точка назначения – сайт, а источник – сетевой интерфейс моего устройства, в случае же эхо-ответа – наоборот. MAC-адрес точки назначения – это первые 6 байт заголовка Ethernet, а MAC-адрес источника – следующие 6 байт заголовка Ethernet, оба MAC-адреса являются индивидуальными и глобально администрируемыми.

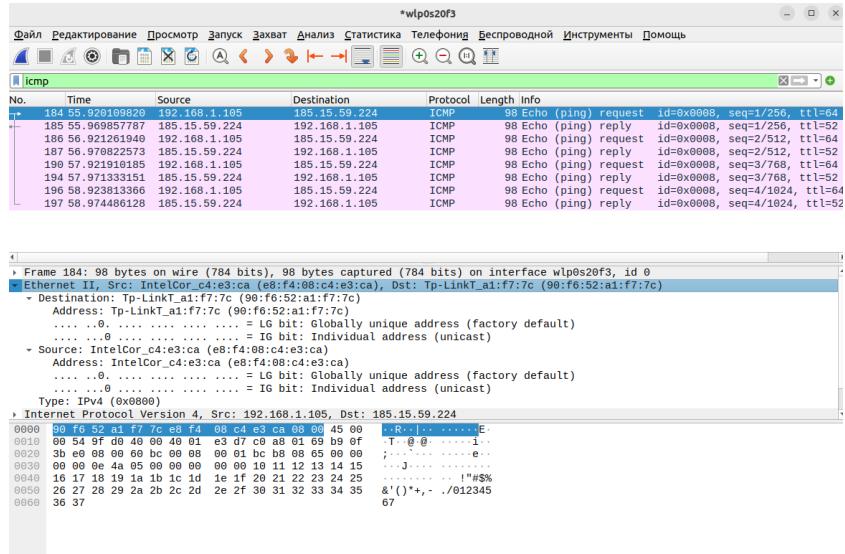


Рис. 3.14: Пингование сайта wikipedia.org. Кадр ICMP - эхо-запрос

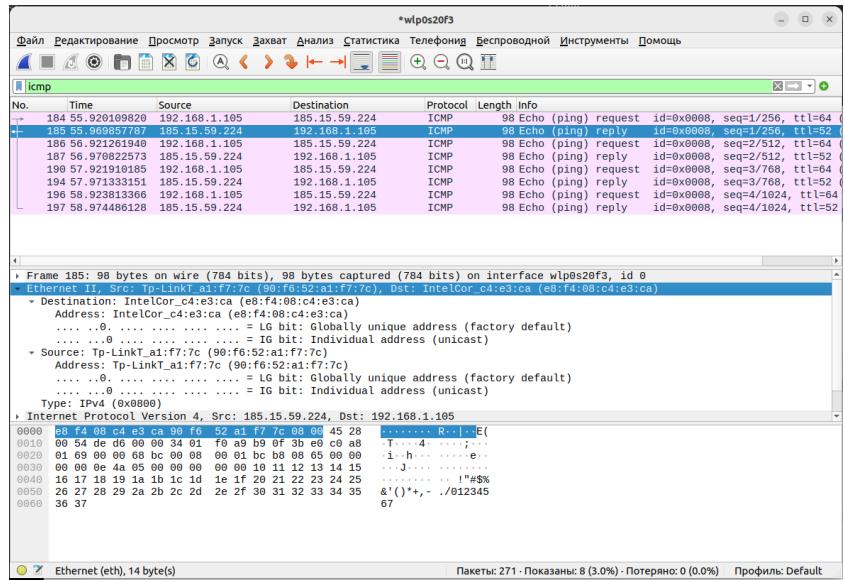


Рис. 3.15: Пингование сайта wikipedia.org. Кадр ICMP - эхо-ответ

Изучим запрос и ответ ARP в программе Wireshark(3.16,??). Изучим информацию на панели сведений о пакете в средней части экрана. В обоих случаях длина кадра равняется 42 байт, заголовок Ethernet первые 14 байт кадра и кадр относится к типу Ethernet II. В случае запроса точка назначения – сетевой интер-

фейс оборудования, а источник – шлюз по умолчанию, в случае же эхо-ответа – наоборот, эхо-ответ сообщает шлюзу по умолчанию MAC-адрес сетевого интерфейса. MAC-адрес точки назначения – это первые 6 байт заголовка Ethernet, а MAC-адрес источника – следующие 6 байт заголовка Ethernet, оба MAC-адреса являются индивидуальными и глобально администрируемыми.

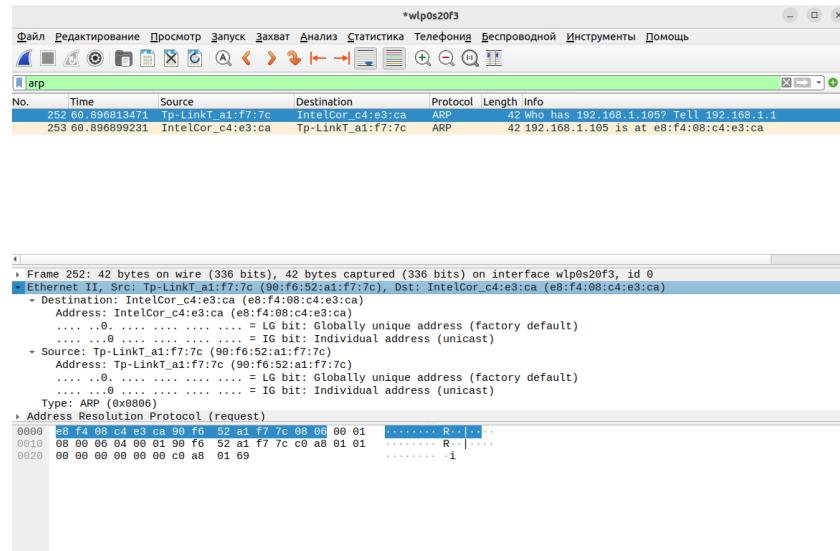


Рис. 3.16: Пингование сайта wikipedia.org. Кадр ARP - запрос

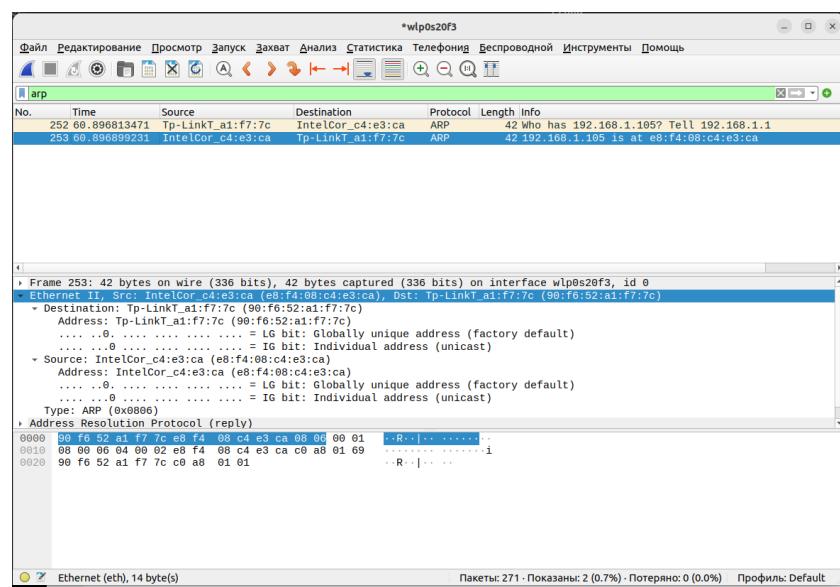


Рис. 3.17: Пингование сайта wikipedia.org. Кадр ARP - ответ

### 3.3 Анализ протоколов транспортного уровня в Wireshark

В браузере перейдем на сайт, работающий по протоколу HTTP, а именно на сайт CERN <http://info.cern.ch/>. Попремещаемся по ссылкам и разделам сайта. В Wireshark в строке фильтра укажим http и проанализируем информацию по протоколу TCP в случае запросов и ответов.

Порт источника задан случайно, выбором из непривилегированных и незанятых портов, и равен 555882, порт назначения равен 80 - это стандартный порт HTTP(3.18).

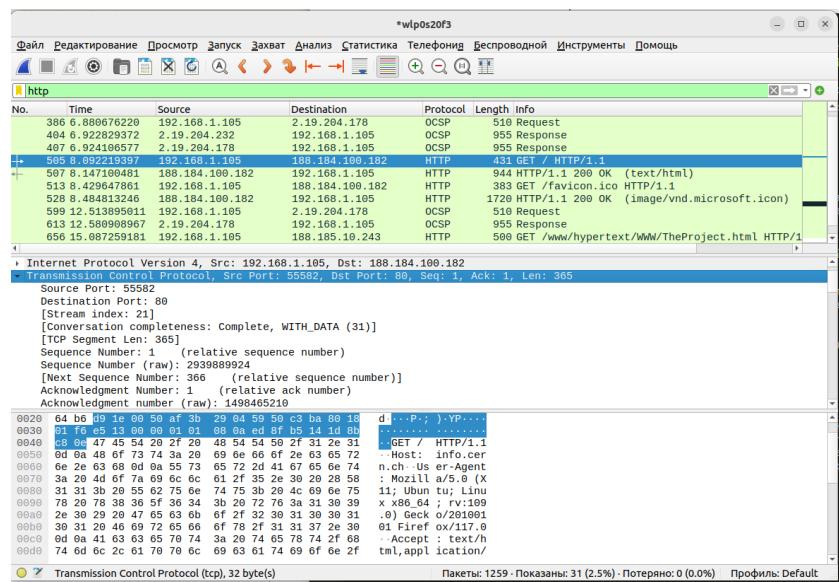


Рис. 3.18: Запрос HTTP по протоколу TCP

В случае ответа порты заданы наоборот, то есть источник - 80 порт, назначение - 55582 (3.19). В разделе HTTP можно увидеть ссылку на html-страницу(3.20). А в разделе Line-based text data находится содержимое страницы, которую перехватил wireshark(3.21).

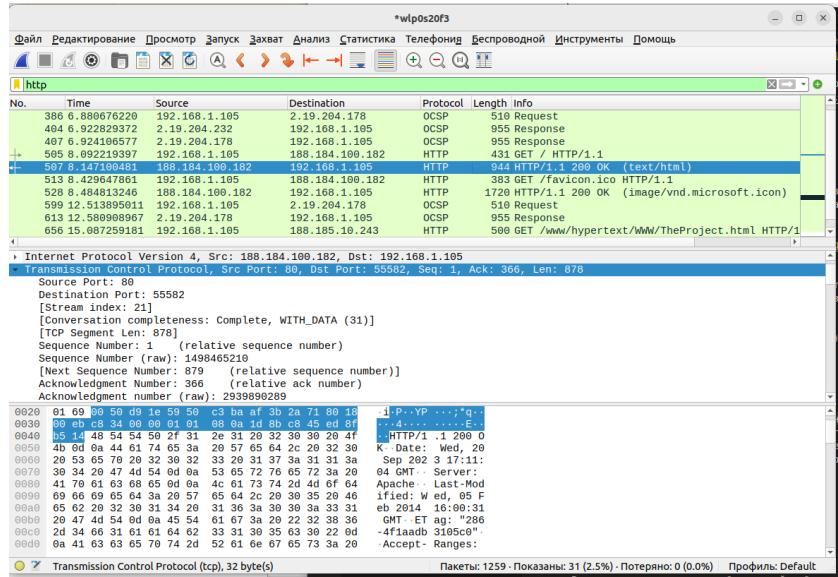


Рис. 3.19: Ответ HTTP по протоколу TCP

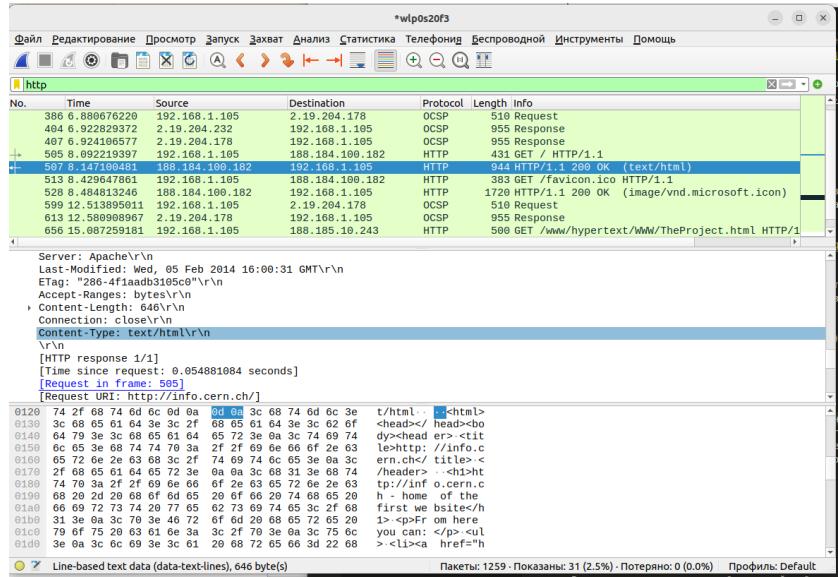


Рис. 3.20: Раздел НТР

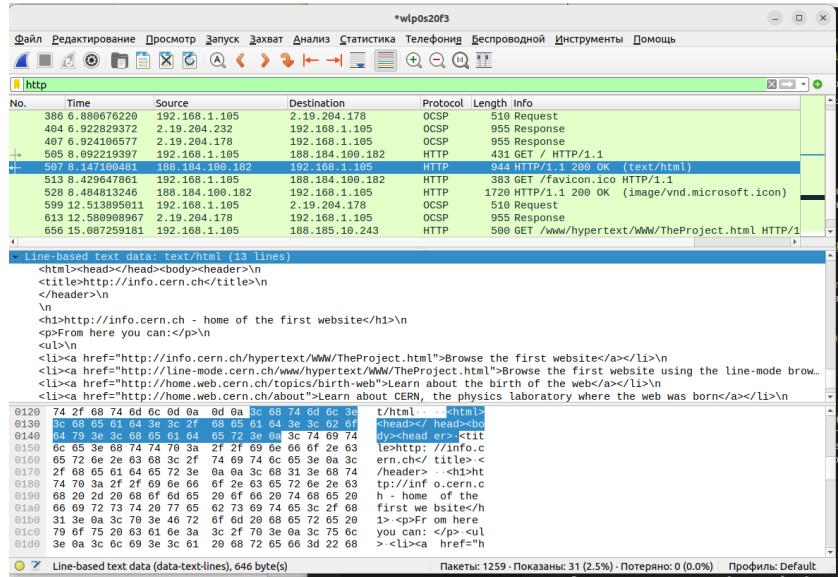


Рис. 3.21: Раздел Line-based text data

Wireshark в строке фильтра укажем dns и проанализируем информацию по протоколу UDP в случае запросов и ответов. Протокол UDP добавляет номера портов, то есть вводит адрес транспортного уровня. Порт источника задан случайно, выбором из непривилегированных и незанятых портов, и равен 555882, порт назначения равен 53 - это стандартный порт DNS(3.22).

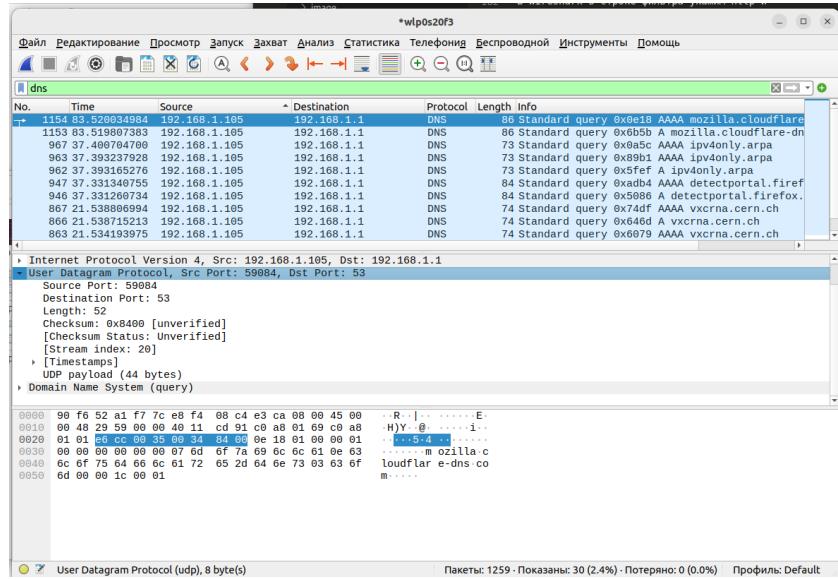


Рис. 3.22: Запрос dns по протоколу UDP

В случае ответа порты заданы наоборот, то есть источник - 53 порт, назначение - 59084 (3.23).

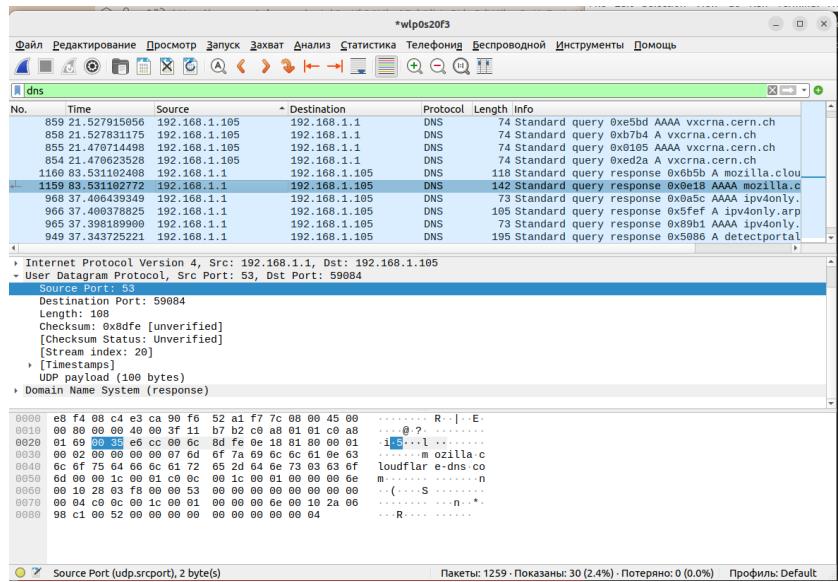


Рис. 3.23: Ответ DNS по протоколу UDP

В Wireshark в строке фильтра укажем `quic` и проанализируем информацию по протоколу `quic` в случае запросов и ответов. Как и в случае `dns` можем посмотреть

информацию транспортного уровня по протоколу UDP. Порт источника задан случайно, выбором из непривилегированных и незанятых портов, и равен 47597, порт назначения равен 443 - это стандартный порт HTTPS, то есть quic сразу криптуется(3.24).

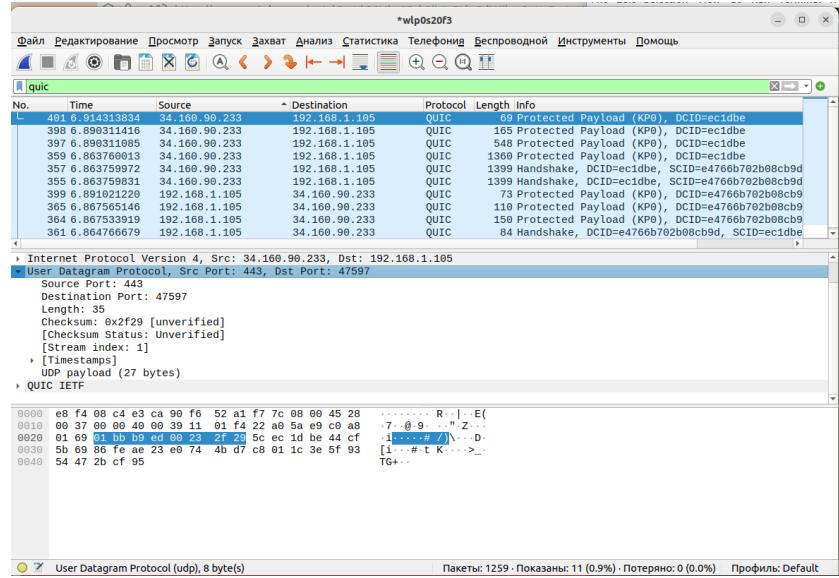


Рис. 3.24: Запрос QUIC по протоколу UDP

В случае ответа порты заданы наоборот, то есть источник - 443 порт, назначение - 47597 (3.24).

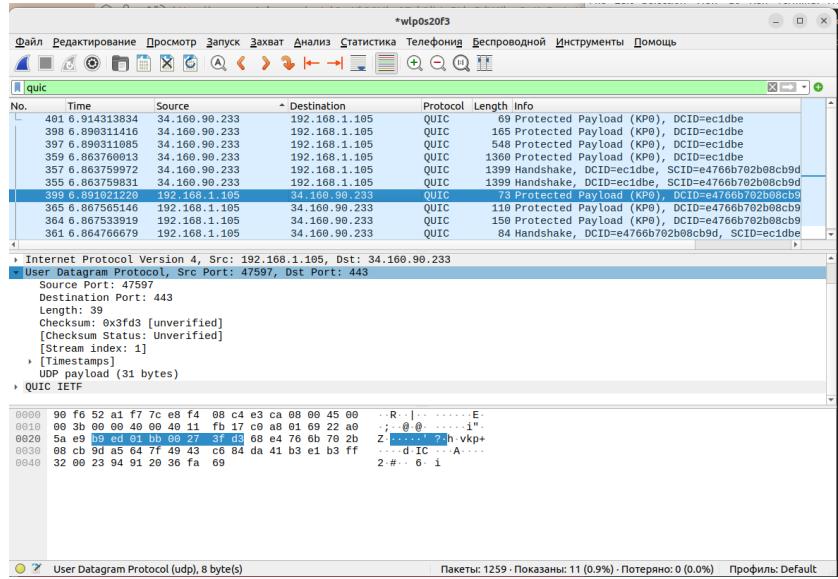


Рис. 3.25: Ответ QUIC по протоколу UDP

Для создания альтернативы TCP поверх UDP строятся протоколы прикладного уровня QUIC IETF, которые управляют трафиком, управляют качеством обслуживания, устанавливают и разрывают соединение (3.26, 3.27).

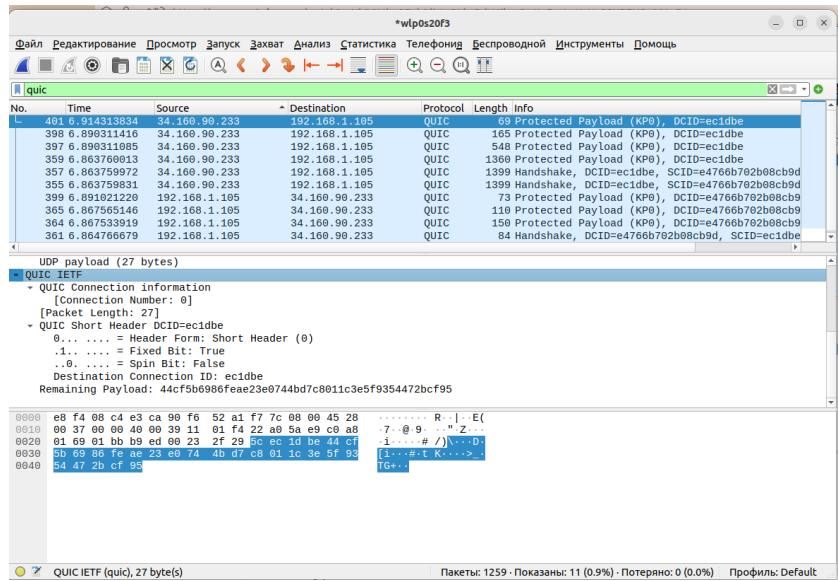


Рис. 3.26: Вкладка QUIK IETF в случае запроса quik

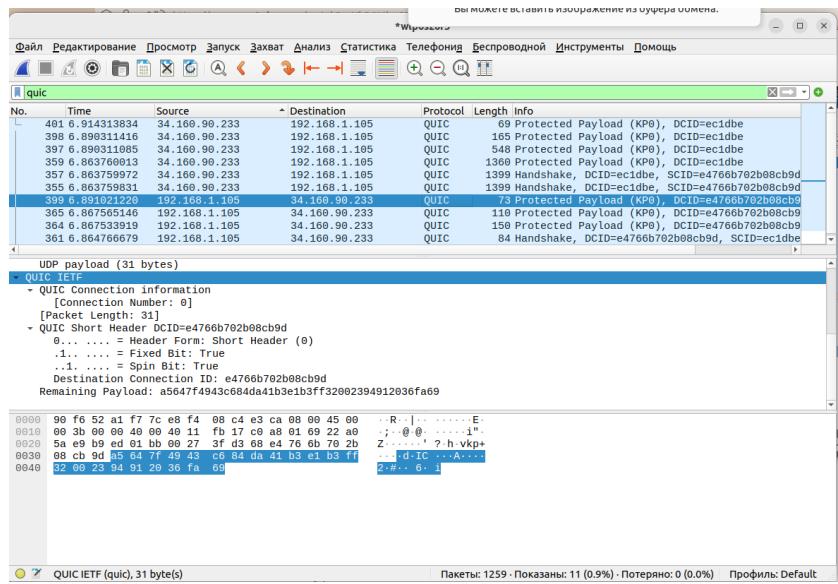


Рис. 3.27: Вкладка QUIC IETF в случае ответа quik

### 3.4 Анализ handshake протокола TCP в Wireshark

Используем соединение по HTTP с сайтом <http://info.cern.ch> для захвата в Wireshark пакетов TCP.

Проанализируем handshake протокола TCP. Установление связи клиент-сервер в TCP осуществляется в три этапа (трёхступенчатый handshake). Опишем эти этапы на примере.

1. Режим активного доступа. Клиент посыпает сообщение SYN, ISSa, т.е. в передаваемом сообщении установлен бит SYN (Synchronize Sequence Number), а в поле Порядковый номер (Sequence Number) — начальное 32-битное значение ISSa (Initial Sequence Number).

Во вкладке с флагами видно, что установлен бит SYN(Syn: set). Порядковый номер равен 3980370530(3.28).

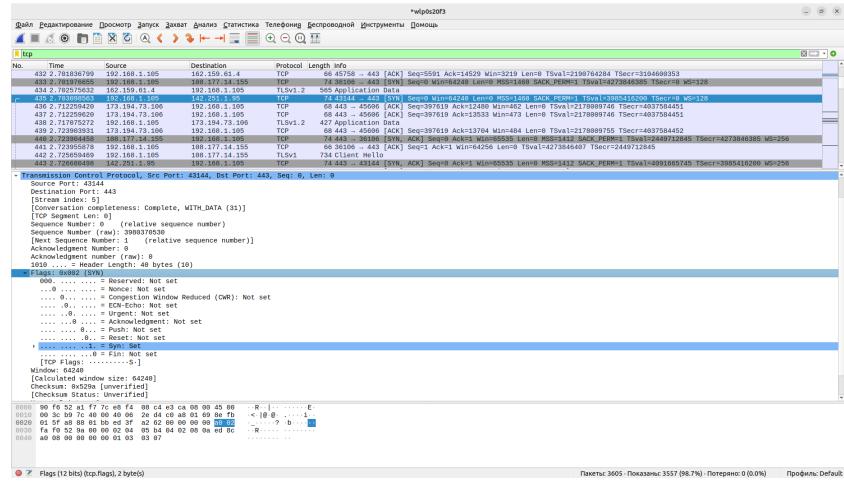


Рис. 3.28: Первая ступень handshake протокола TCP

2. Режим пассивного доступа. Сервер откликается, посылая сообщение SYN, ACK, ISSb, ACK(ISSa+1), т.е. установлены биты SYN и ACK; в поле Порядковый номер (Sequence Number) хостом В устанавливается начальное значение счётчика — ISSb; поле Номер подтверждения (Acknowledgment Number) содержит значение ISSa, полученное в первом пакете от хоста А и увеличенное на единицу.

Во вкладке с флагами видно, что установлены биты SYN и ACK(Syn: set, Acknowledgment: set). Порядковый номер содержит значение ISSb и равен 452625189. Поле номер подтверждения равно значению ISSa, которое получил в предыдущем пакете, плюс 1, то есть 3980370531(3.29).

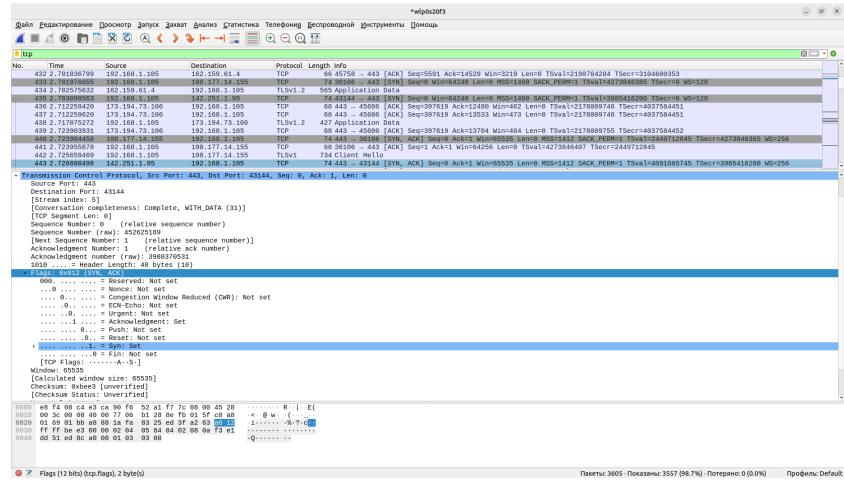


Рис. 3.29: Вторая ступень handshake протокола TCP

3. Завершение рукопожатия. Клиент отправляет подтверждение получения SYN-сегмента от сервера с идентификатором, равным ISN (сервера)+1: ACK, ISS<sub>a</sub>+1, ACK(ISS<sub>b</sub>+1). В этом пакете установлен бит ACK, поле Порядковый номер (Sequence Number) содержит ISS<sub>a</sub>+1, поле Номер подтверждения (Acknowledgment Number) содержит значение ISS<sub>b</sub>+1. Посылкой этого пакета заканчивается трёхступенчатый handshake, и TCP-соединение считается установленным.

Во вкладке с флагами видно, что установлен бит ACK(Acknowledgment: set). Порядковый номер равен ISS<sub>a</sub>+1 и равен 3980370531. Поле номер подтверждения равен значению ISS<sub>b</sub>, которое получил в предыдущем пакете, плюс 1, то есть 452625190.(3.30).

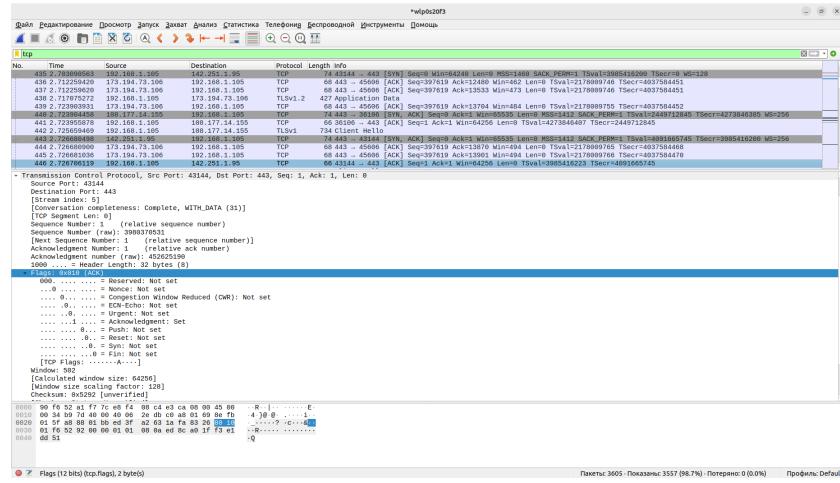


Рис. 3.30: Третья ступень handshake протокола TCP

Теперь клиент может посыпать пакеты с данными на сервер по только что созданному виртуальному TCP-каналу: ACK, ISSa+1, ACK(ISSb+1).

В Wireshark в меню «Статистика» выберем «График Потока». В отчёте приведите пояснения по изменениям значений соответствующих сообщений при установлении соединения по TCP.

На графике видно, что сначала клиент послал сообщение на сервер(стрелка вправо), а значение seq = 0. Затем сервер откликнулся(стрелка влево), значение seq = 0, а значение ack = 1. И в третьем пакете клиент оправил подтверждение получение SYN-сегмента(стрелка влево), оба значения syn и ack стали равны 1 (3.31).

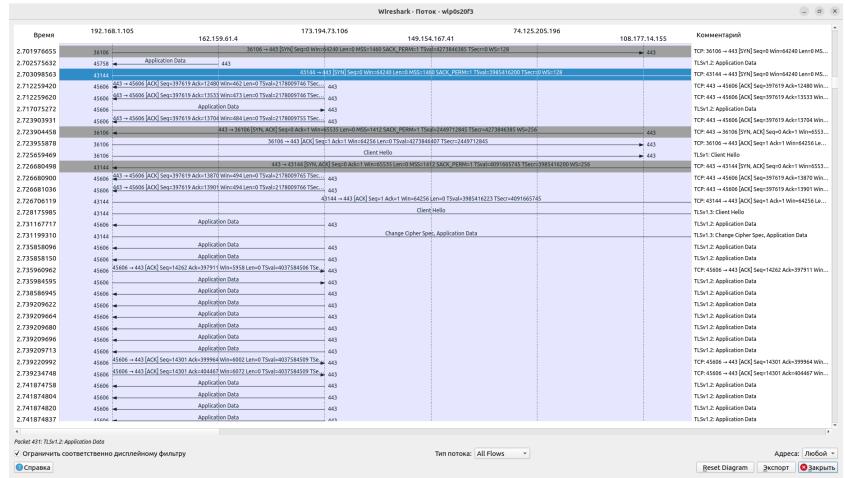


Рис. 3.31: График потока

## **4 Выводы**

В результате выполнения лабораторной работы посредством Wireshark были изучены кадры Ethernet, а также проанализированы PDU протоколы транспортного и прикладного уровней стека TCP/IP.