Отчёт по лабораторной работе 3

Анализ трафика в Wireshark

Элсаиед Адел

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

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

2 Выполнение работы

2.1 Анализ сетевого соединения

1. На рабочей станции под управлением ОС Windows в консоли выполнен вывод параметров текущего сетевого соединения командой `ipconfig` с расширенной опцией `/all`, позволяющей получить полную информацию о сетевом адаптере (MAC-адрес, параметры DHCP и DNS, сведения об аренде адреса).

   В ходе анализа установлено, что активным интерфейсом является беспроводной адаптер **Intel(R) Wi-Fi 6E AX211 160MHz**. В выводе команды отражены:

   - Physical Address — аппаратный MAC-адрес сетевого интерфейса;
   - DHCP enabled — автоматическое получение сетевых параметров;
   - IPv6 link-local — локальный IPv6-адрес канального уровня;
   - IPv4 address — адрес узла в локальной сети;
   - Subnet Mask — маска подсети;
   - Default Gateway — адрес маршрутизатора;
   - DHCP Server и DNS Server — сетевые службы, обслуживающие подключение;
   - Lease Obtained / Lease Expires — сроки действия DHCP-аренды.

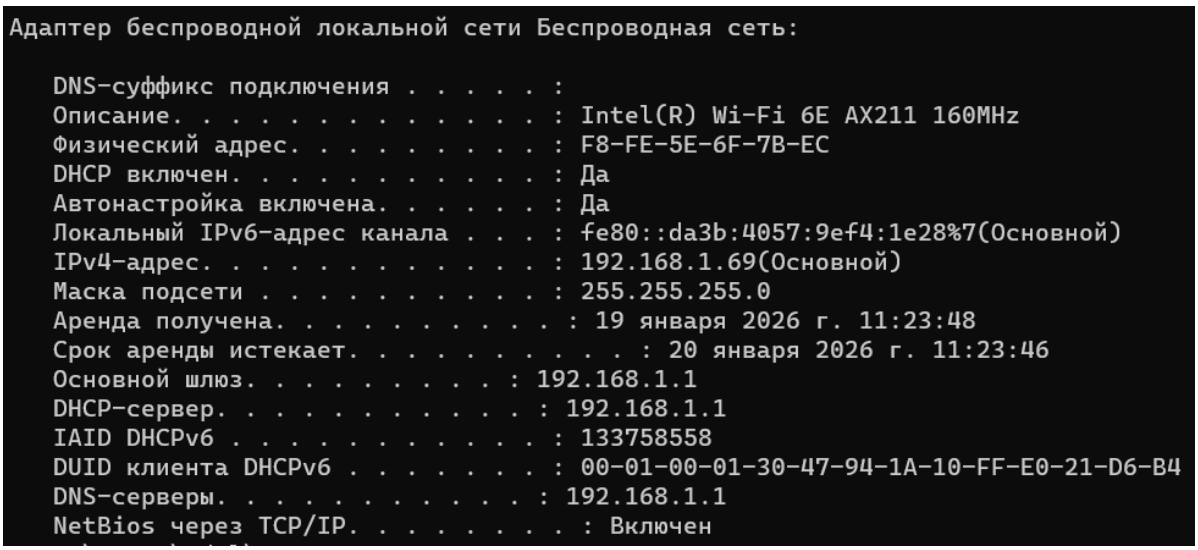

Рис. 2.1: Вывод параметров сетевого соединения командой ipconfig /all

2. По результатам выполнения команды `ipconfig /all` определены основные параметры сетевого подключения:
    - IPv4-адрес: `192.168.1.69`;
    - Маска подсети: `255.255.255.0`;
    - Шлюз по умолчанию: `192.168.1.1`;
    - DHCP-сервер: `192.168.1.1`;
    - DNS-сервер: `192.168.1.1`;
    - MAC-адрес интерфейса Wi-Fi: `F8-FE-5E-6F-7B-EC`.
3. Для проверки доступности шлюза по умолчанию выполнена команда `ping 192.168.1.1`. В результате получены ответы на все отправленные ICMP-запросы без потерь пакетов, а время отклика составило менее 1 мс, что свидетельствует о корректной работе локального сетевого сегмента.

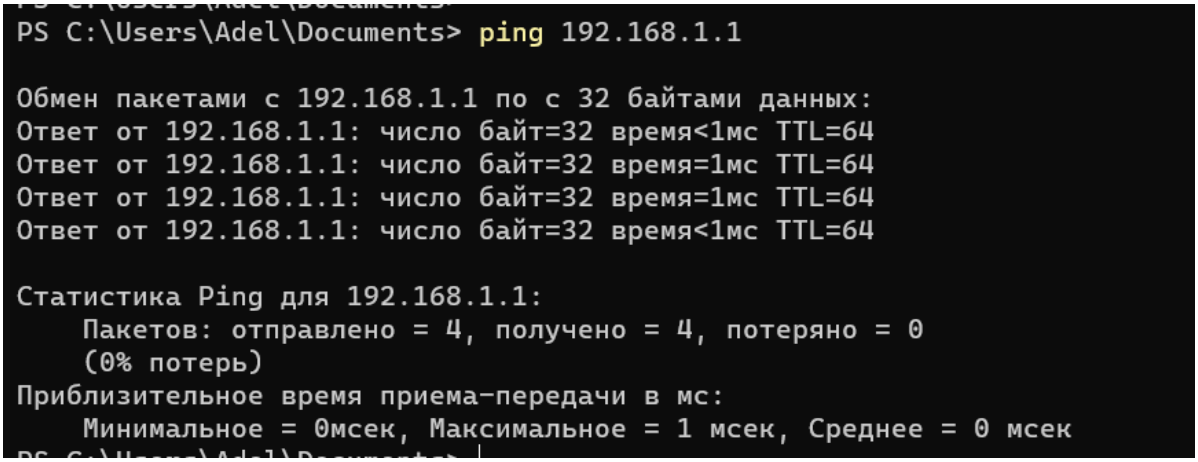

Рис. 2.2: Проверка доступности шлюза по умолчанию

4. MAC-адреса сетевых интерфейсов определены:
   - из вывода `ipconfig /all`: `F8-FE-5E-6F-7B-EC`;
   - из захваченных кадров Ethernet II в Wireshark (поле Source): `f8:fe:5e:6f:7b:ec`.

   Также определён MAC-адрес шлюза по умолчанию: `c8:7f:54:78:b6:f2`.

5. Структура MAC-адреса рассмотрена на примере `F8-FE-5E-6F-7B-EC`:
   - первые 24 бита (`F8-FE-5E`) — идентификатор производителя (OUI);
   - последние 24 бита (`6F-7B-EC`) — уникальный идентификатор сетевого интерфейса.

   Первый октет `F8` в двоичном виде равен `11111000`.
   - младший бит равен 0 — адрес индивидуальный (unicast);
   - второй младший бит равен 0 — адрес глобально администрируемый (назначен производителем).

2.2 Анализ кадров канального уровня в Wireshark

6. На рабочем устройстве установлен и запущен анализатор сетевого трафика Wireshark. В качестве интерфейса захвата выбран активный беспроводной

сетевой адаптер.

7. В консоли определены IP-адрес устройства и шлюз по умолчанию с использованием команды `ipconfig`.

8. Для генерации трафика выполнена команда `ping 192.168.1.1`. После этого захват трафика в Wireshark был остановлен и применён фильтр отображения `arp or icmp`.

9. Проанализирован кадр ICMP Echo Request:

    - длина кадра — 74 байта;
    - тип канального кадра — Ethernet II;
    - MAC-адрес источника — `f8:fe:5e:6f:7b:ec`;
    - MAC-адрес назначения — `c8:7f:54:78:b6:f2`;
    - оба MAC-адреса являются индивидуальными и глобально администрируемыми.

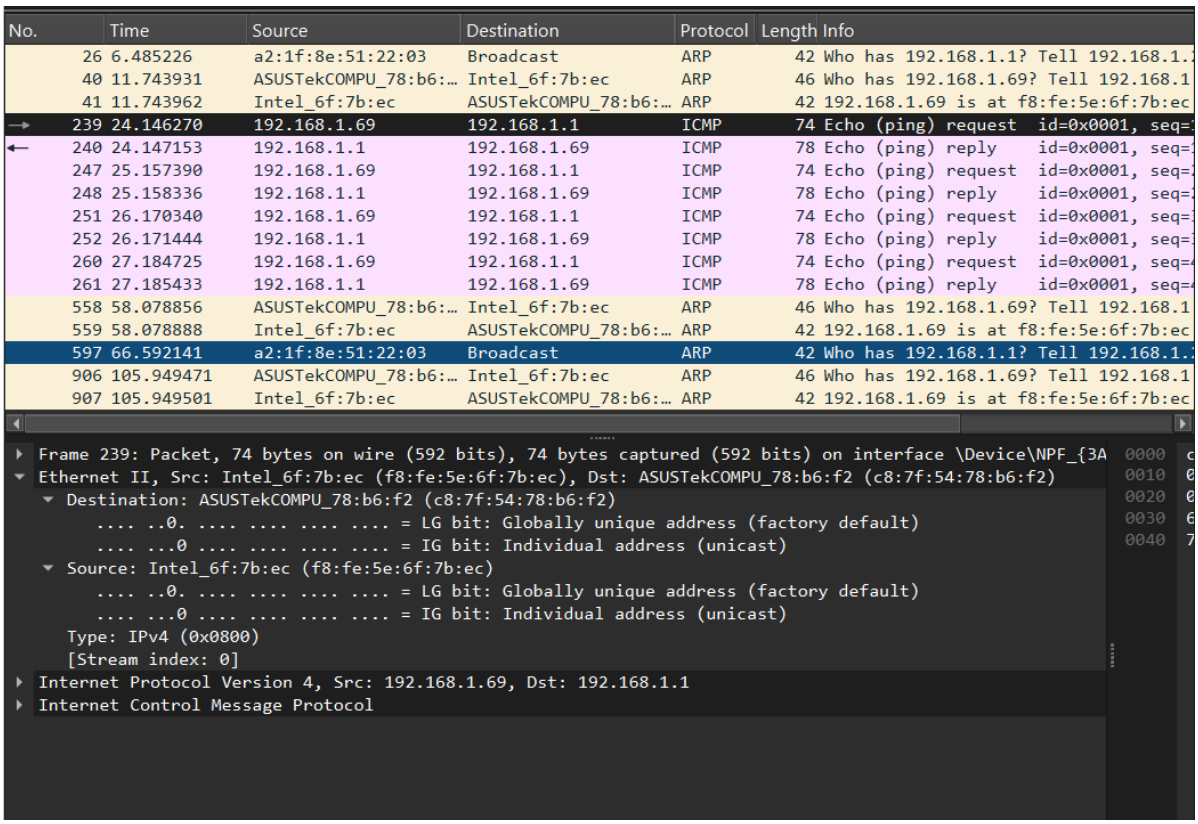

Рис. 2.3: ICMP Echo request

10. Проанализирован кадр ICMP Echo Reply:

    - длина кадра — 78 байт;
    - тип кадра — Ethernet II с тегом 802.1Q VLAN;
    - MAC-адрес источника — `c8:7f:54:78:b6:f2`;
    - MAC-адрес назначения — `f8:fe:5e:6f:7b:ec`.

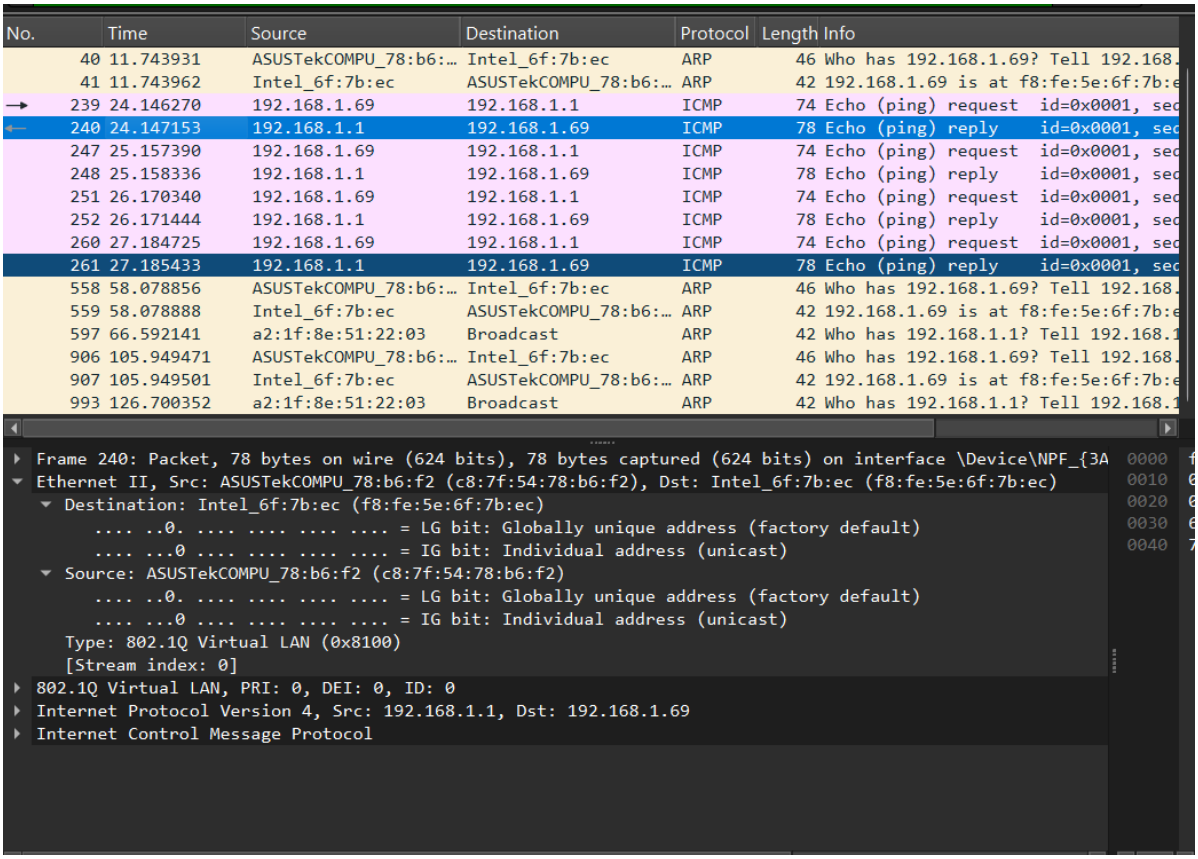

Рис. 2.4: ICMP Echo reply

11. Изучены кадры протокола ARP. Рассмотрен ARP-запрос, в котором шлюз определяет MAC-адрес узла по его IP-адресу. В кадре зафиксированы поля Ethernet II и заголовок ARP.

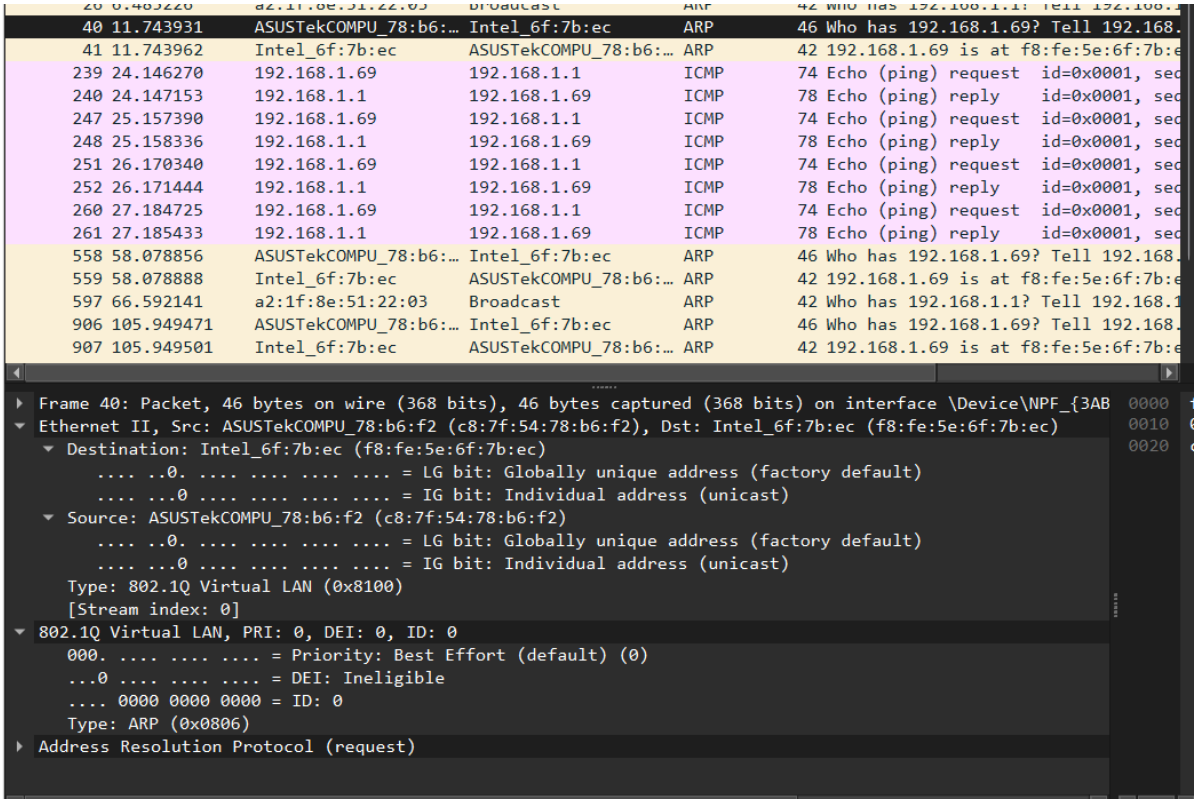

Рис. 2.5: ARP request

12. Выполнён новый захват трафика и отправлен ICMP-запрос по доменному имени `www.yandex.ru`. Имя было успешно разрешено в IP-адрес `77.88.55.88`, и получены ответы без потерь пакетов.

```
PS C:\Users\Adel\Documents> ping www.yandex.ru

Обмен пакетами с www.yandex.ru [77.88.55.88] с 32 байтами данных:
Ответ от 77.88.55.88: число байт=32 время=10мс TTL=53
Ответ от 77.88.55.88: число байт=32 время=10мс TTL=53
Ответ от 77.88.55.88: число байт=32 время=10мс TTL=53
Ответ от 77.88.55.88: число байт=32 время=9мс TTL=53

Статистика Ping для 77.88.55.88:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 9мсек, Максимальное = 10 мсек, Среднее = 9 мсек
```

Рис. 2.6: Ping по доменному имени

13. Проанализирован ARP-ответ, содержащий сопоставление IP-адреса `192.168.1.69` с MAC-адресом `f8:fe:5e:6f:7b:ec`.

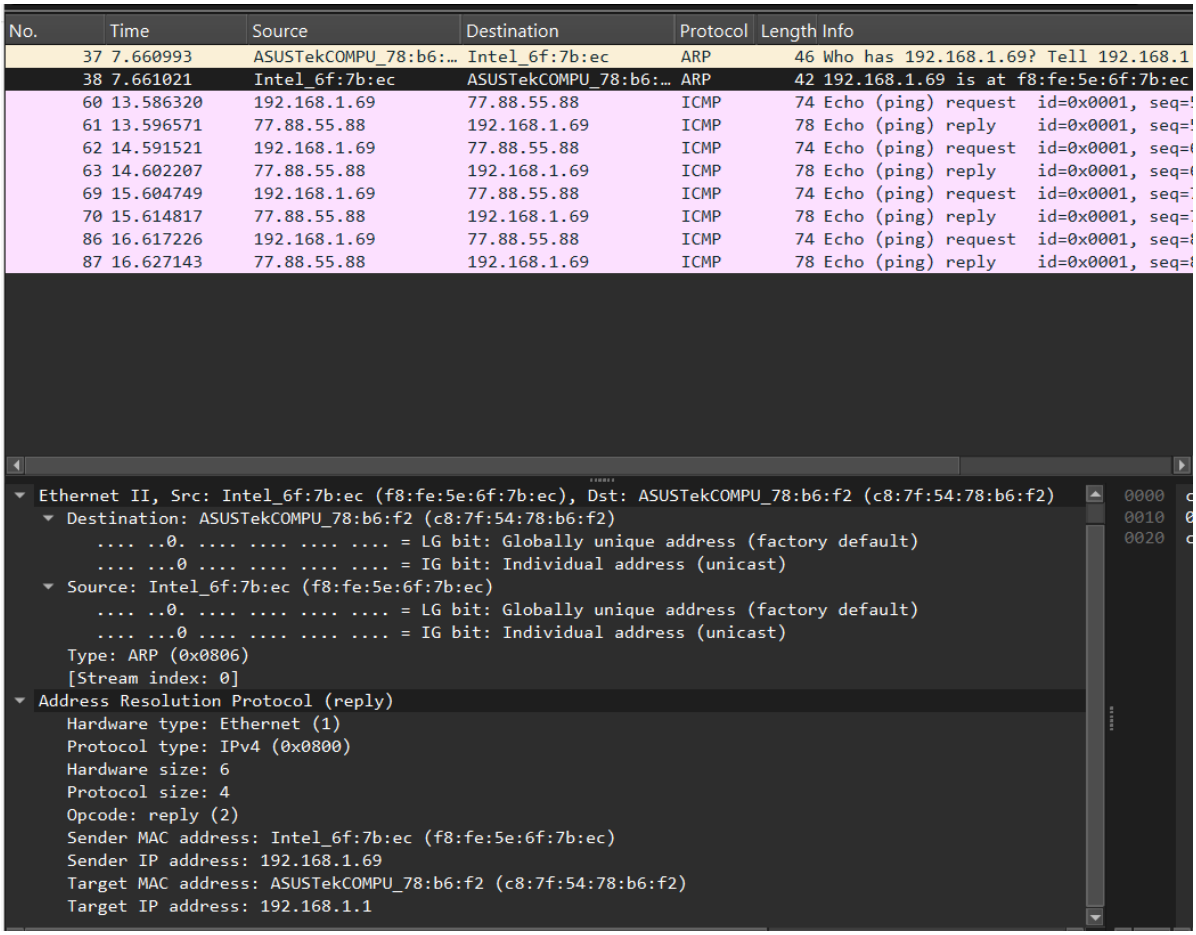

Рис. 2.7: ARP reply

14. Изучен ICMP Echo Request к внешнему узлу `77.88.55.88`. На канальном уровне кадр направляется на MAC-адрес шлюза, так как целевой IP-адрес находится за пределами локальной подсети.

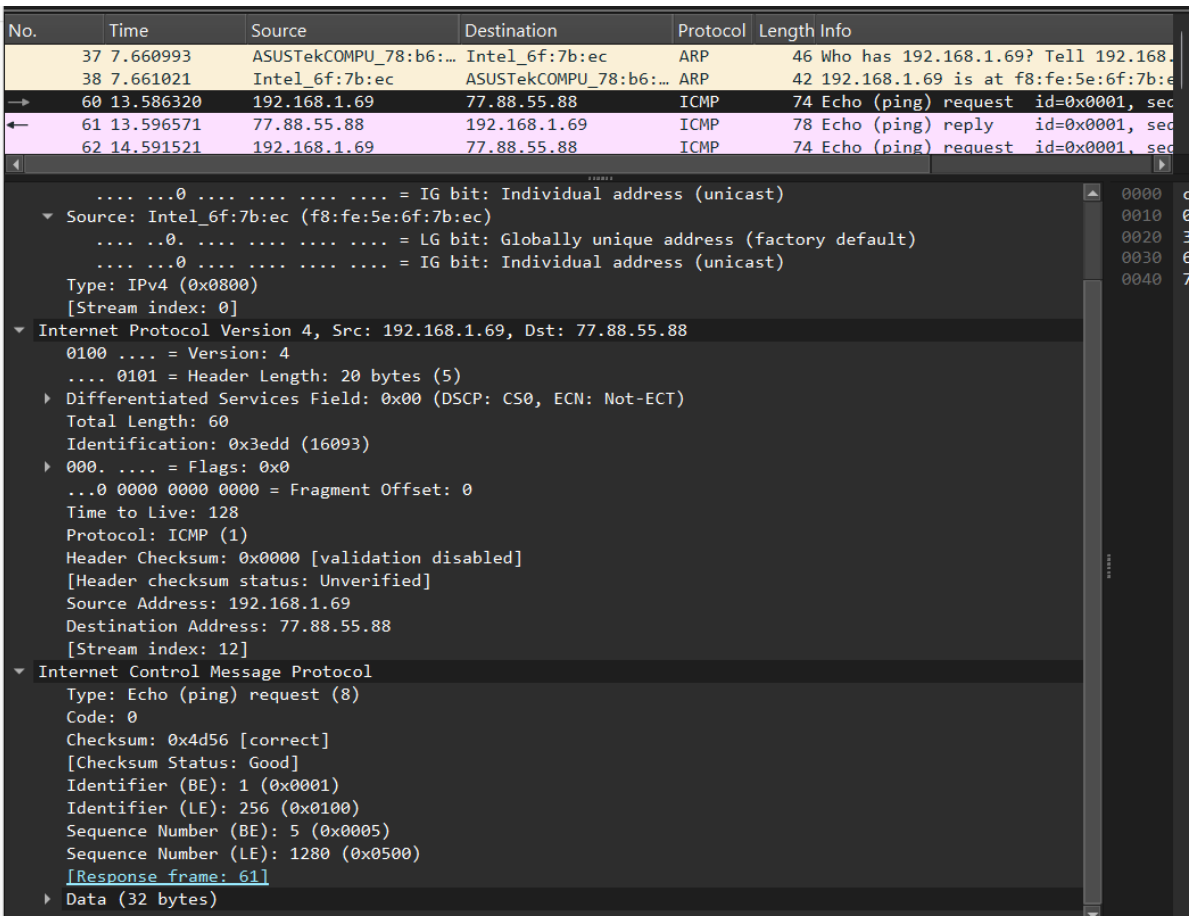

Рис. 2.8: ICMP request к внешнему узлу

15. Изучен ICMP Echo Reply от внешнего узла. На канальном уровне источник кадра — MAC-адрес шлюза, что подтверждает корректную работу маршрутизации и повторную инкапсуляцию пакетов при передаче в локальную сеть.

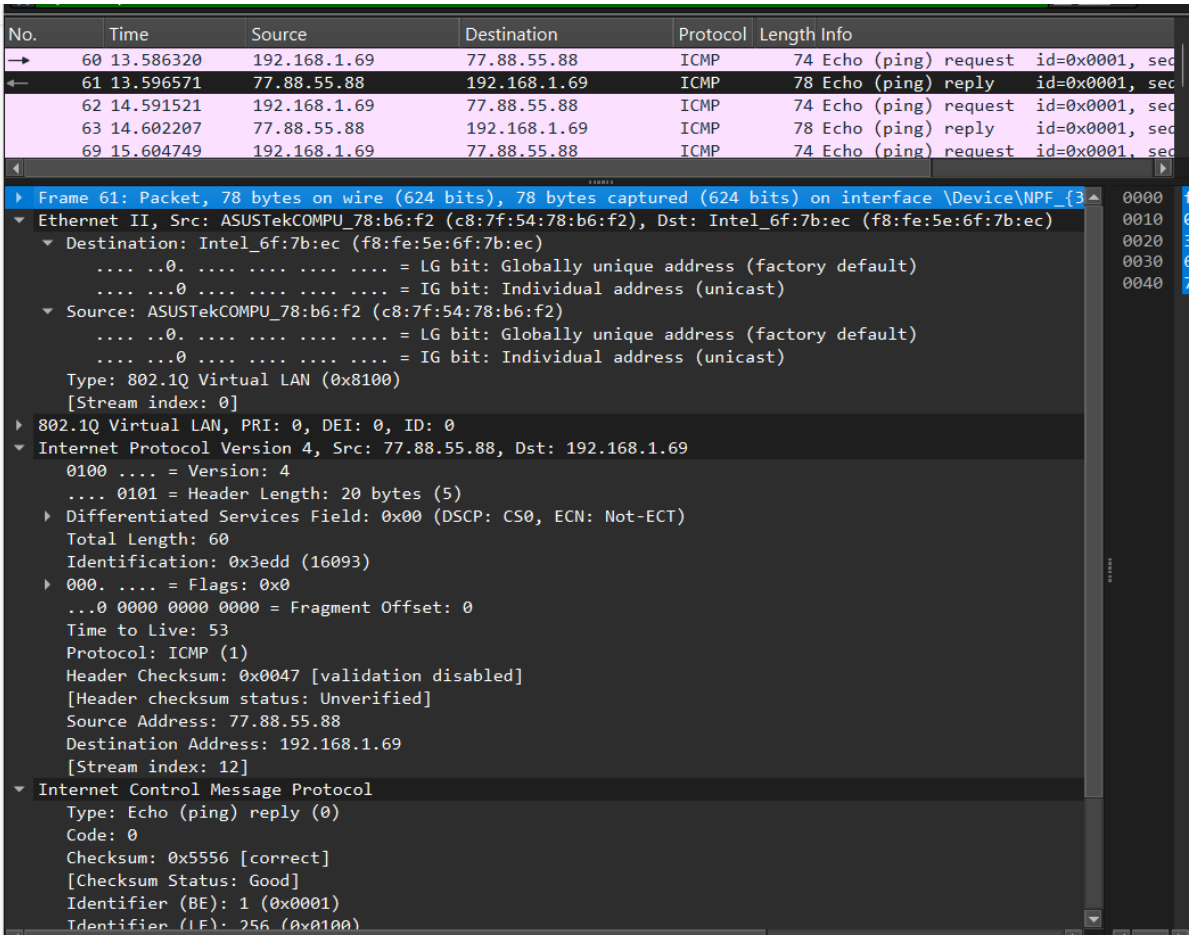

Рис. 2.9: ICMP reply от внешнего узла

2.3 Анализ протоколов транспортного уровня в Wireshark (HTTP, DNS, QUIC)

1. На рабочем устройстве запущен анализатор сетевого трафика Wireshark. В качестве интерфейса захвата выбран активный беспроводной сетевой адаптер. Убедившись в начале захвата пакетов, выполнены действия для генерации сетевого трафика.

2. В браузере выполнен переход на сайт, работающий по протоколу HTTP. В процессе загрузки страницы и перехода по ссылкам был сгенерирован HTTP-трафик, который зафиксирован в Wireshark.

3. Для анализа HTTP-трафика в строке фильтра Wireshark указан фильтр `http`. В результате отображены HTTP-запросы и ответы, инкапсулированные в сегменты TCP.

   Проанализирован HTTP-запрос **GET**:

   - IP-адрес источника: `192.168.1.69`;
   - IP-адрес назначения: `188.184.67.127`;
   - Протокол транспортного уровня: TCP;
   - Порт источника: `54457` (динамический клиентский порт);
   - Порт назначения: `80` (HTTP);
   - Флаги TCP: `PSH, ACK`, указывают на передачу прикладных данных и подтверждение ранее полученных сегментов;
   - Длина полезной нагрузки TCP: 473 байта;
   - На прикладном уровне зафиксирован HTTP-запрос GET ресурса `/hypertext/WWW/TheProject.html`.

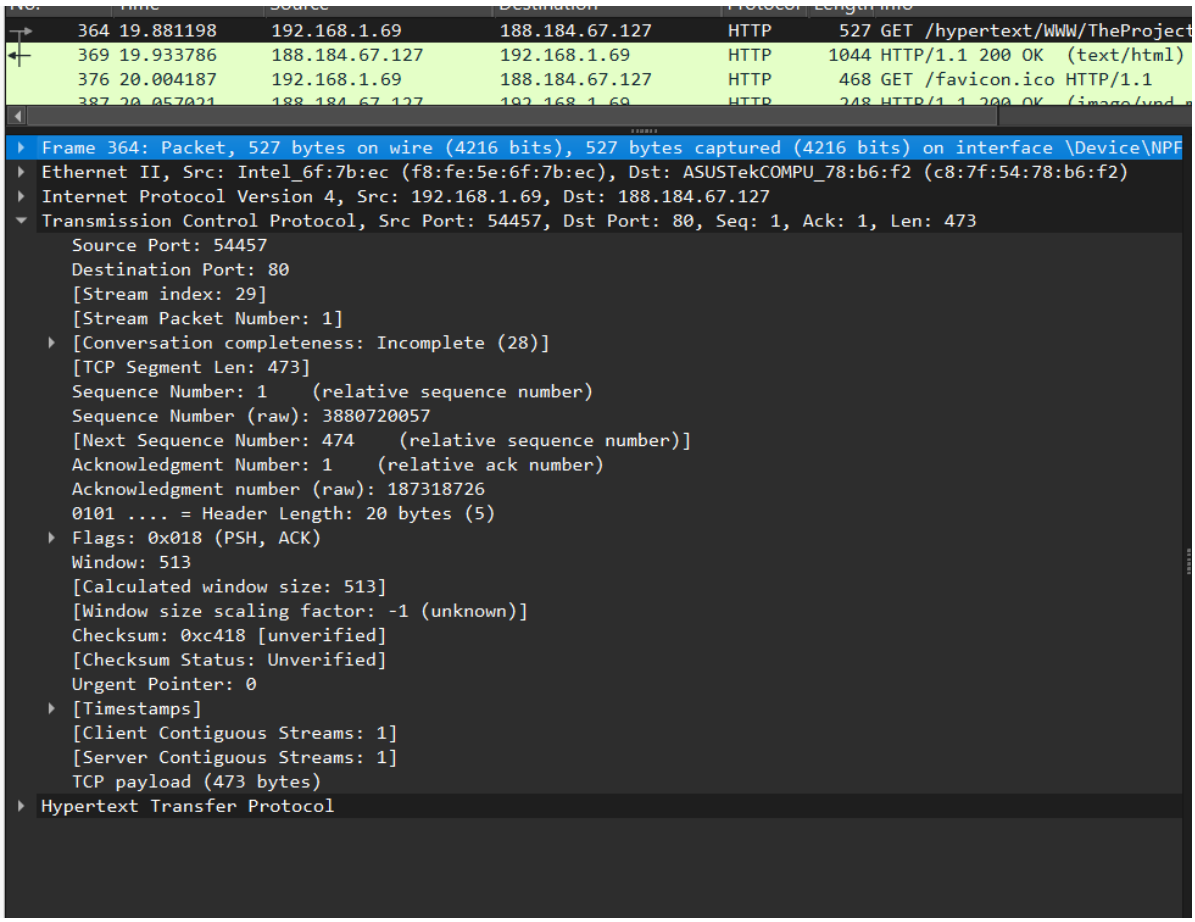

Рис. 2.10: HTTP GET запрос и параметры TCP-сегмента

4. Проанализирован HTTP-ответ сервера:

   - IP-адрес источника: `188.184.67.127`;
   - IP-адрес назначения: `192.168.1.69`;
   - Порт источника: `80`;
   - Порт назначения: `54457`;
   - Длина TCP-сегмента: 990 байт;
   - Зафиксирован статус ответа **HTTP/1.1 200 OK**;
   - Тип содержимого: `text/html`;
   - Данные передаются в нескольких TCP-сегментах с последующей сборкой (TCP reassembly).

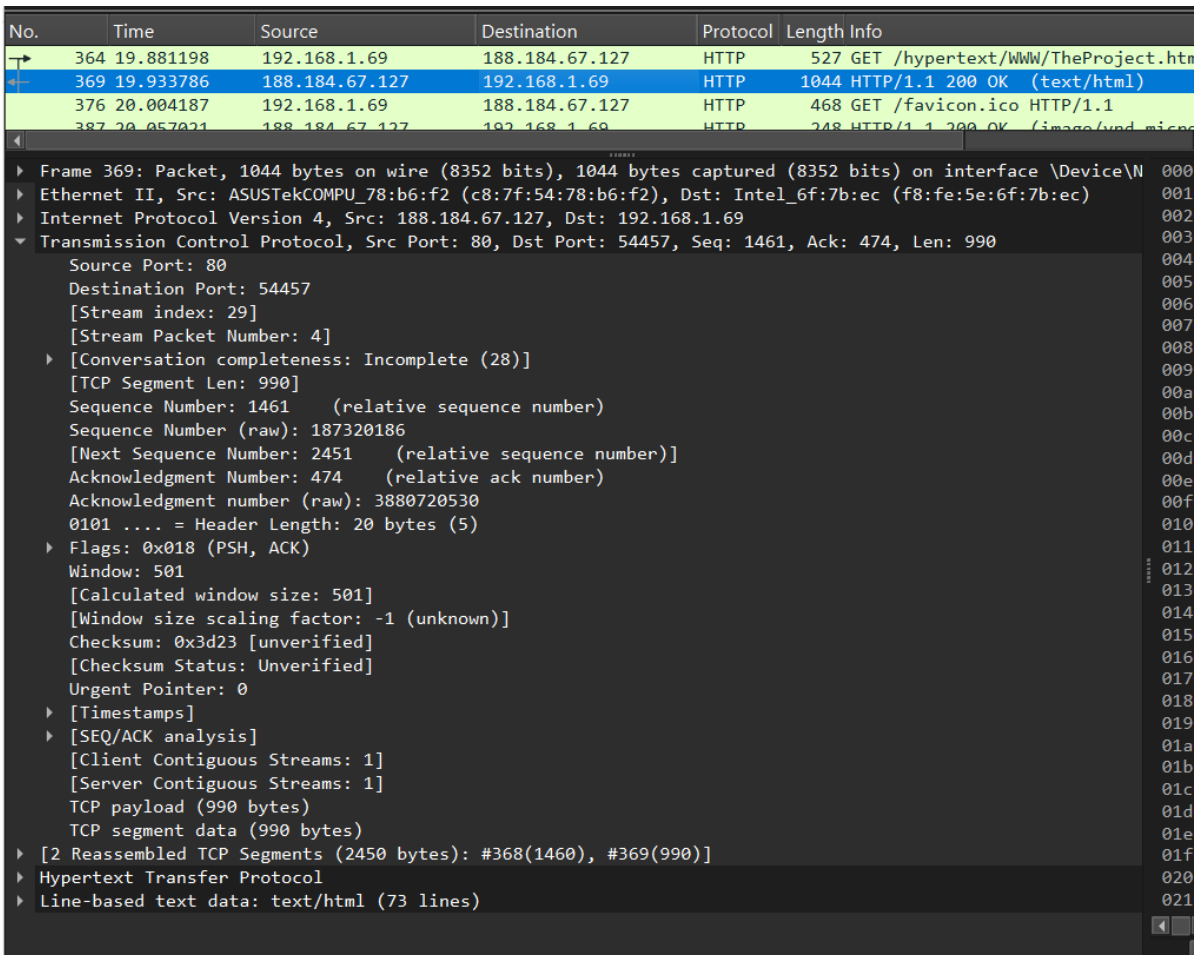

Рис. 2.11: HTTP ответ 200 OK и сборка TCP-сегментов

5. Для анализа доменных запросов в строке фильтра Wireshark указан фильтр `dns`. В результате отображены DNS-запросы и ответы, передаваемые по протоколу UDP.

   Проанализирован DNS-запрос:

   - IP-адрес источника: `192.168.1.69`;
   - IP-адрес назначения: `192.168.1.1` (DNS-сервер);
   - Протокол транспортного уровня: UDP;
   - Порт источника: `55855` (динамический);
   - Порт назначения: `53` (DNS);
   - Тип сообщения: Standard query;

- Тип запроса: A / HTTPS;
- Длина UDP-пакета: 48 байт.

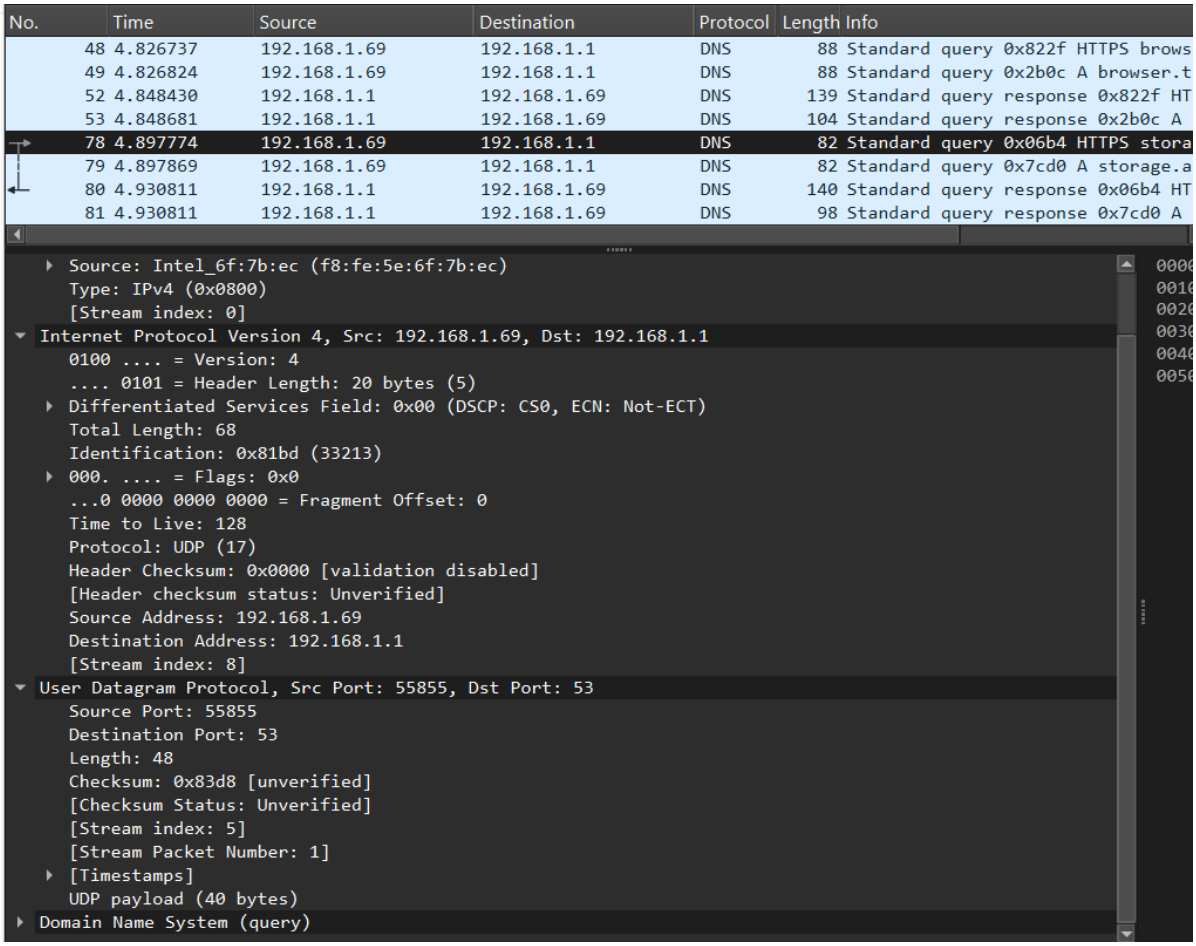

Рис. 2.12: DNS запрос по протоколу UDP

6. Проанализирован DNS-ответ сервера:

- IP-адрес источника: `192.168.1.1`;
- IP-адрес назначения: `192.168.1.69`;
- Протокол транспортного уровня: UDP;
- Порт источника: 53;
- Порт назначения: 55855;
- Тип сообщения: Standard query response;
- В ответе содержится сопоставление доменного имени с IP-адресом;

- Установлен флаг **Don’t fragment**, значение TTL равно 64.

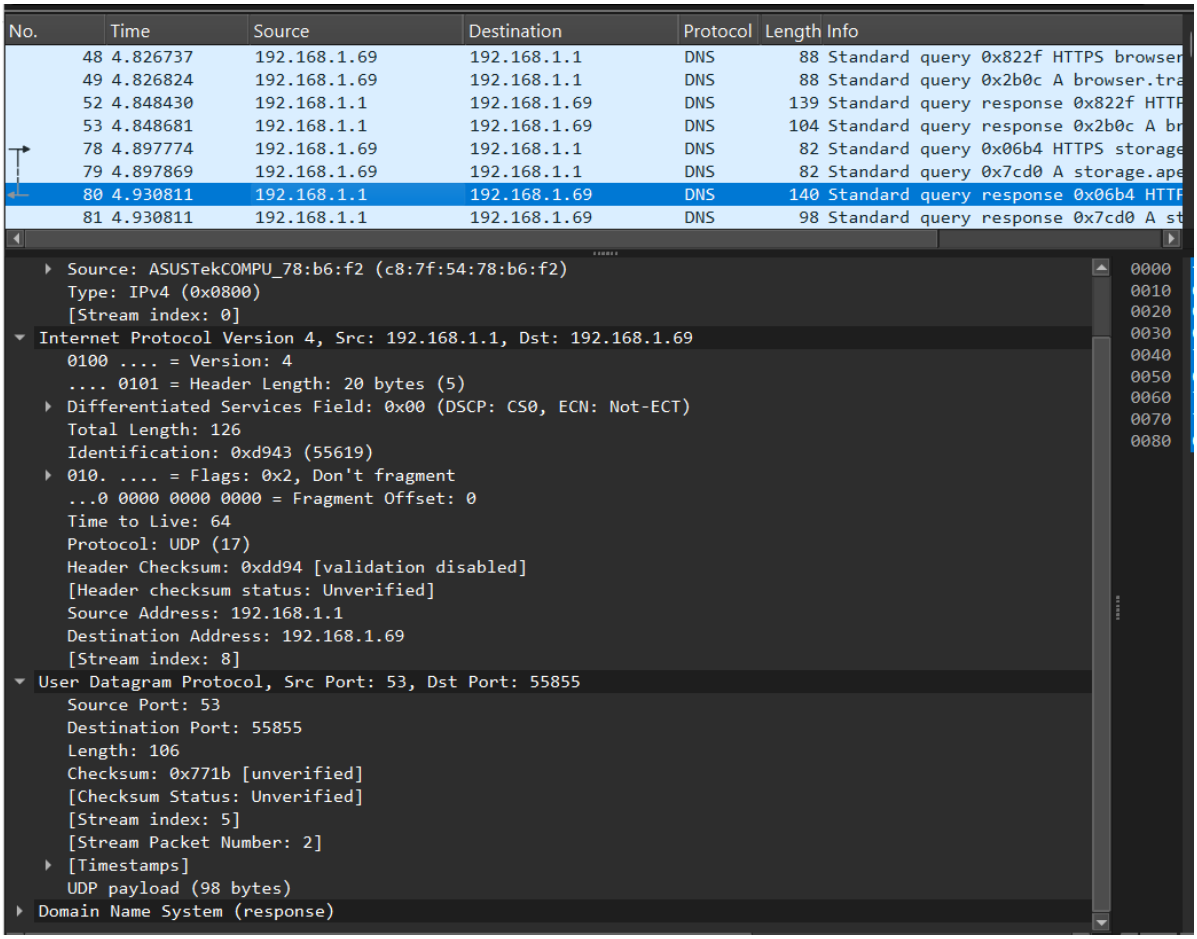

Рис. 2.13: DNS ответ с результатом разрешения имени

7. Для анализа современного транспортного протокола в строке фильтра Wireshark указан фильтр `quic`. Зафиксирован обмен данными между клиентом и внешним сервером по протоколу QUIC, использующему UDP.

   Проанализирован пакет QUIC Initial:

   - IP-адрес источника: `173.194.220.94`;
   - IP-адрес назначения: `192.168.1.69`;
   - Протокол транспортного уровня: UDP;
   - Порт источника: `443`;
   - Порт назначения: `50215`;

- Тип заголовка QUIC: Long Header;
- Тип пакета: Initial;
- Версия QUIC: 1;
- Используются идентификаторы соединения (Connection ID);
- Передаётся криптографическая информация для установки защищённого соединения.

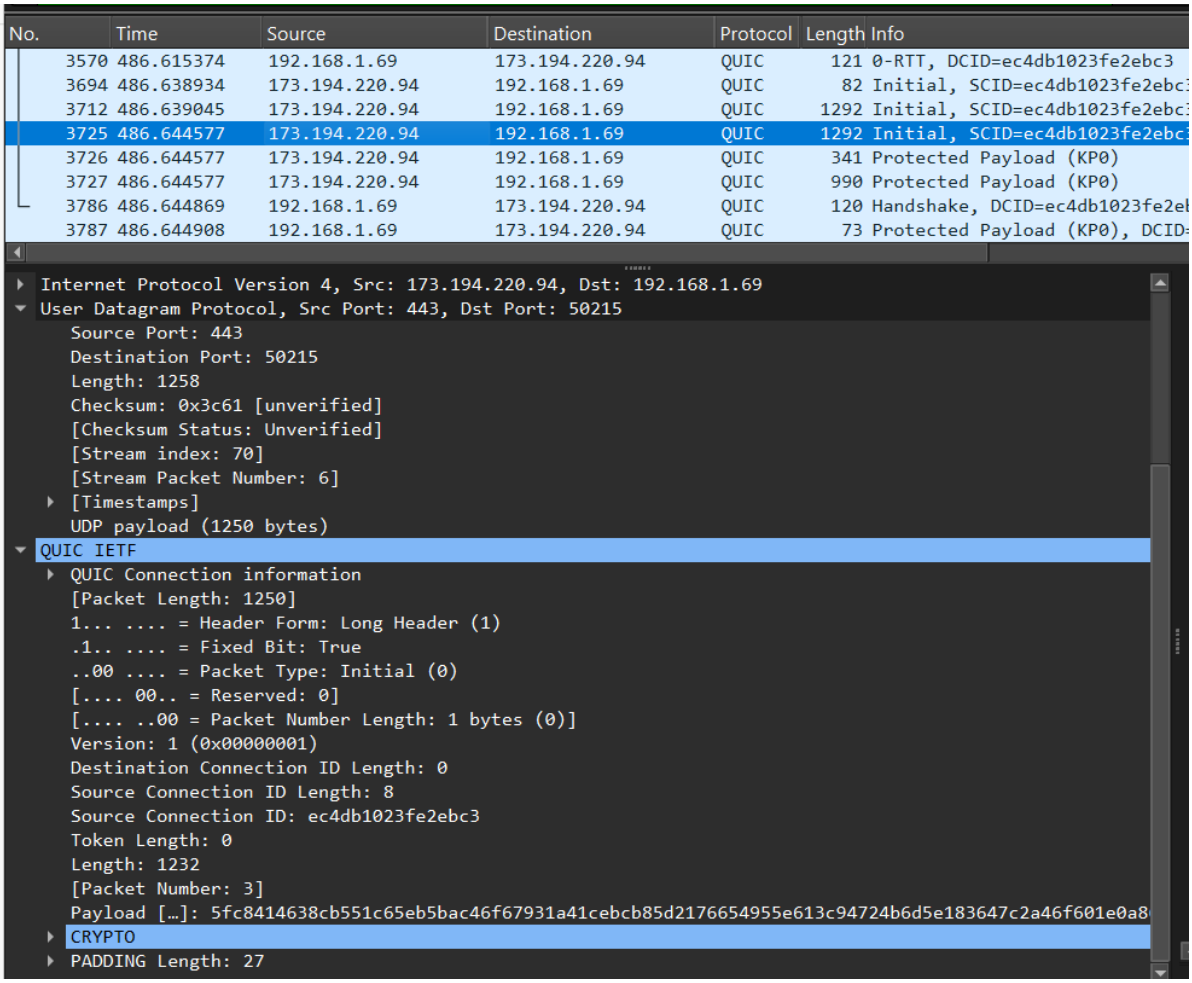

Рис. 2.14: QUIC Initial пакет и параметры соединения

8. Проанализирован пакет QUIC Handshake:

- Тип пакета: Handshake;
- Длина пакета: 78 байт;
- Передача осуществляется поверх UDP;

- Часть полезной нагрузки зашифрована, что подтверждается невозможностью расшифровки без ключей;
- Протокол QUIC обеспечивает встроенное шифрование и установку защищённого канала без использования TCP.

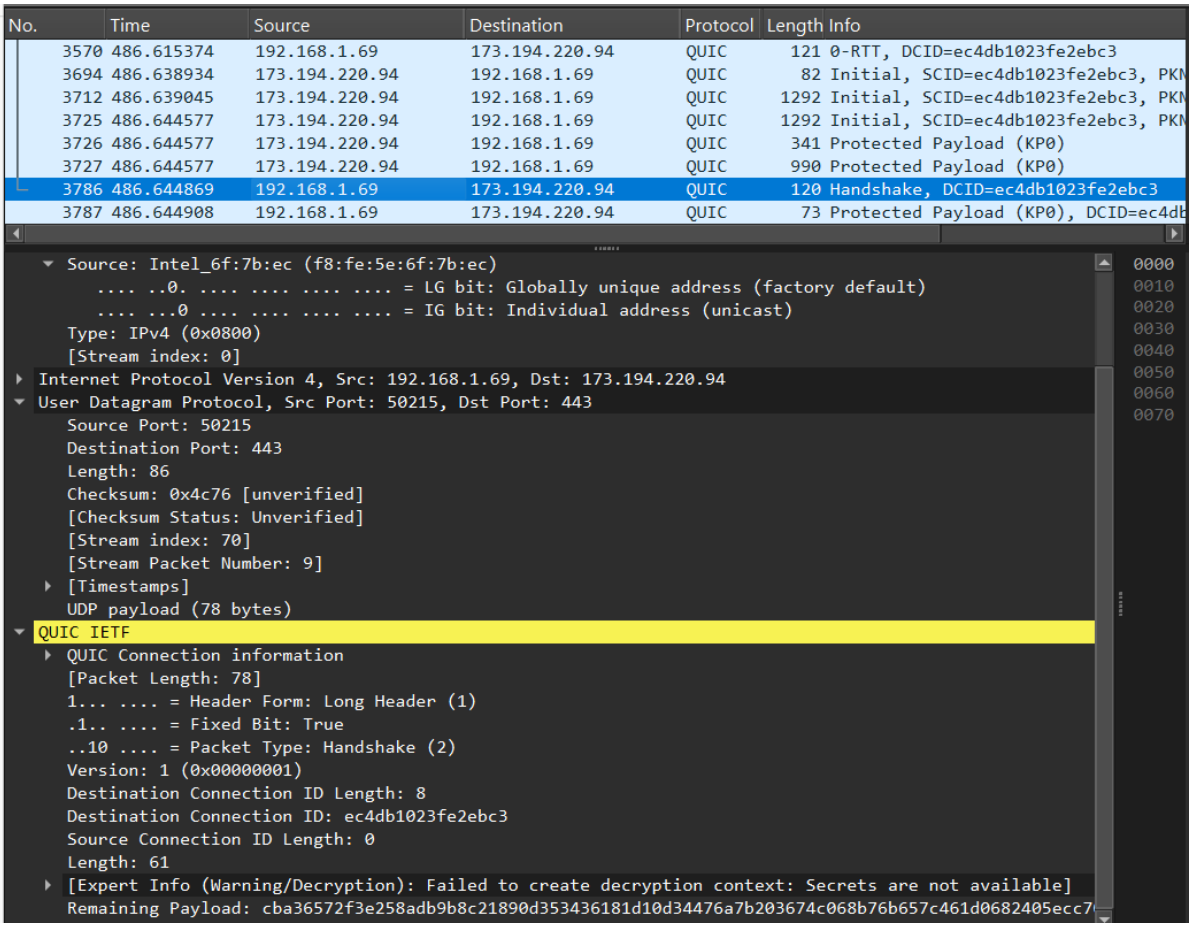

Рис. 2.15: QUIC Handshake пакет

9. По завершении анализа захват трафика в Wireshark был остановлен. Полученные данные подтверждают особенности работы транспортных протоколов:

   - HTTP использует TCP с установлением соединения и подтверждением доставки;
   - DNS преимущественно использует UDP для быстрого обмена запросами и ответами;

- QUIC работает поверх UDP, объединяя функции транспортного уровня и криптографической защиты, обеспечивая низкие задержки и защищённую передачу данных.

2.4 Анализ handshake протокола TCP в Wireshark

1. На рабочем устройстве запущен Wireshark. В качестве источника захвата выбран активный беспроводной сетевой интерфейс. После выбора интерфейса подтверждено начало процесса захвата сетевого трафика.

2. Для генерации TCP-трафика выполнено подключение к удалённому веб-серверу по протоколу HTTP. В результате инициировано установление TCP-соединения между локальным узлом с IP-адресом `192.168.1.69` и удалённым сервером `188.184.67.127` по порту `80`.

3. В процессе анализа списка пакетов Wireshark рассмотрен пример установки TCP-соединения (three-way handshake):

   Этап 1 — SYN

   Клиент инициирует соединение, отправляя TCP-сегмент:

   - Источник: `192.168.1.69`, порт `58579`;
   - Назначение: `188.184.67.127`, порт `80`;
   - Установлен флаг `SYN`;
   - Номер последовательности (Sequence Number): `0`.

   Этап 2 — SYN, ACK

   Сервер подтверждает получение запроса и предлагает собственный номер последовательности:

   - Источник: `188.184.67.127`, порт `80`;
   - Назначение: `192.168.1.69`, порт `58579`;
   - Установлены флаги `SYN, ACK`;

- Sequence Number: `0`;
- Acknowledgment Number: `1`, что подтверждает получение SYN от клиента.

Этап 3 — ACK

Клиент завершает установление соединения:

- Источник: `192.168.1.69`, порт `58579`;
- Назначение: `188.184.67.127`, порт `80`;
- Установлен флаг `ACK`;
- Sequence Number: `1`;
- Acknowledgment Number: `1`.

После выполнения этих трёх шагов TCP-соединение считается установленным, и стороны переходят к обмену данными прикладного уровня (HTTP).

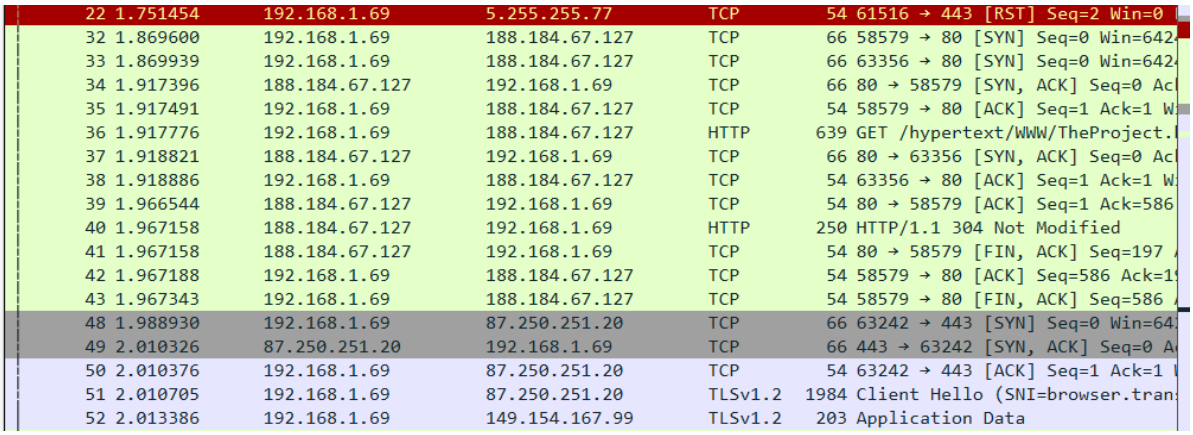

Рис. 2.16: Последовательность TCP handshake в списке пакетов Wireshark

4. После установления соединения зафиксирована передача данных:
   - HTTP GET-запрос передаётся с флагами `PSH, ACK`;
   - Сервер отправляет подтверждения (`ACK`) и ответ HTTP;
   - В завершении соединения стороны обмениваются сегментами с флагами `FIN, ACK`, после чего соединение корректно закрывается.

Также зафиксированы TCP-сегменты с флагом `RST`, что указывает на принудительное завершение соединения для отдельных попыток подключения (например, при обращении к закрытому порту).

5. Для наглядного анализа последовательности обмена TCP-сегментами в Wireshark использован инструмент **«Статистика → График потока»**. На графике отчётливо отображается:

   - направление передачи сегментов между клиентом и сервером;
   - этапы `SYN → SYN, ACK → ACK`;
   - дальнейшая передача данных с флагами `PSH, ACK`;
   - корректное завершение соединения с использованием `FIN, ACK`.

   Изменение значений `Seq` и `Ack` на графике подтверждает корректную работу механизма нумерации байтов и подтверждения доставки данных в TCP.

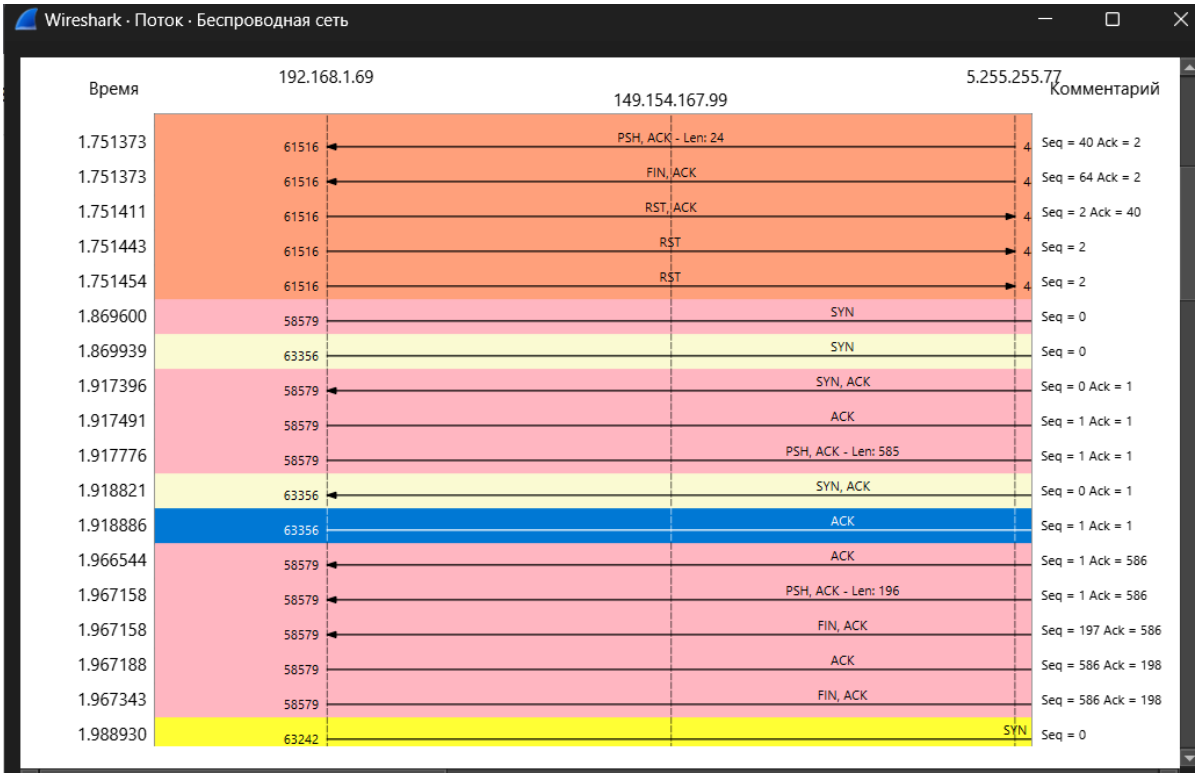

Рис. 2.17: График потока TCP-соединения в Wireshark

6. По завершении анализа захват трафика в Wireshark был остановлен. Полу-

ченные результаты подтверждают стандартный механизм установки TCP-соединения, состоящий из трёх этапов, а также демонстрируют процесс передачи данных и корректного завершения сеанса связи.

3 Заключение

В ходе выполнения лабораторной работы был выполнен захват и анализ сетевого трафика с использованием программы Wireshark. Изучены параметры сетевого подключения рабочей станции, определены IP- и MAC-адреса, а также принципы их использования на канальном и сетевом уровнях модели OSI.

Проведён анализ кадров протоколов ARP и ICMP, что позволило проследить процесс разрешения IP-адресов в MAC-адреса и подтвердить корректную работу механизма проверки доступности узлов в локальной и внешней сетях. Установлено, что при взаимодействии с внешними ресурсами на канальном уровне кадры адресуются MAC-адресу шлюза по умолчанию.

Дополнительно изучены протоколы транспортного и прикладного уровней HTTP, DNS и QUIC. Показаны различия в использовании протоколов TCP и UDP, а также особенности протокола QUIC, совмещающего функции транспортного уровня и криптографической защиты.

Отдельное внимание уделено анализу процесса установления TCP-соединения. На практике подтверждена работа трёхстороннего рукопожатия TCP, а также корректная передача и завершение соединения с использованием соответствующих флагов.

В результате выполненной работы закреплены практические навыки анализа сетевого трафика, интерпретации заголовков пакетов и понимания принципов взаимодействия протоколов различных уровней сетевой модели.