Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет безопасности информационных технологий

Дисциплина:

«Компьютерные сети»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

«Анализ трафика компьютерных сетей утилитой Wireshark»

Выполнил:

Арендаренко М. М., студент группы N3347

(подпись)

Проверил:

Есипов Д.А

_____

(подпись)

Санкт-Петербург

2024 г.

Содержание

Оглавление

Введение

Цель работы — изучение методов анализа сетевого трафика с использованием утилиты Wireshark, включая захват пакетов, их анализ и интерпретацию. В ходе работы будет рассмотрено несколько видов трафика (ping, tracert, HTTP, DNS, ARP, FTP и DHCP.), а также исследование структуры и значений ключевых полей заголовков сетевых пакетов.

Для выполнения лабораторной работы необходимо:

1. Настроить утилиту Wireshark для захвата сетевого трафика.
2. Запустить различные утилиты и программы (ping, tracert, nslookup и т.д.), и одновременно с этим начать захват трафика в Wireshark.
3. Проанализировать захваченные пакеты для каждого из протоколов, выявить ключевые параметры, такие как заголовки, типы сообщений, коды состояний.
4. Сделать выводы по каждому из этапов анализа, описав особенности работы протоколов, их структуру и возможные проблемы, выявленные при анализе трафика.

1.Анализ трафика утилиты PING

Ping — это сетевая утилита, используемая для проверки доступности узла в сети и измерения времени отклика от него. Название "ping" происходит от звука, который создается при эхолокации, что символизирует посылку и получение сигнала.

Утилита **ping** использует протокол **ICMP** (Internet Control Message Protocol) для отправки специального запроса на указанный IP-адрес и получения ответа. В процессе работы **ping** отправляет **ICMP Echo Request** (запрос эха) и получает в ответ **ICMP Echo Reply**

Для выполнения лабораторной работы будем использовать сайт google.com

```
C:\Users\maxim>ping -l 100 googl.com

C:\Users\maxim>ping google.com

Обмен пакетами с google.com [142.251.1.102] с 32 байтами данных:
Ответ от 142.251.1.102: число байт=32 время=9мс TTL=107
Ответ от 142.251.1.102: число байт=32 время=6мс TTL=107
Ответ от 142.251.1.102: число байт=32 время=5мс TTL=107
Ответ от 142.251.1.102: число байт=32 время=6мс TTL=107

Статистика Ping для 142.251.1.102:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 5мсек, Максимальное = 9 мсек, Среднее = 6 мсек
```

Рис. 1 Использование утилиты ping

Используя фильтра пакетов в Wireshark находим пакеты ICMP

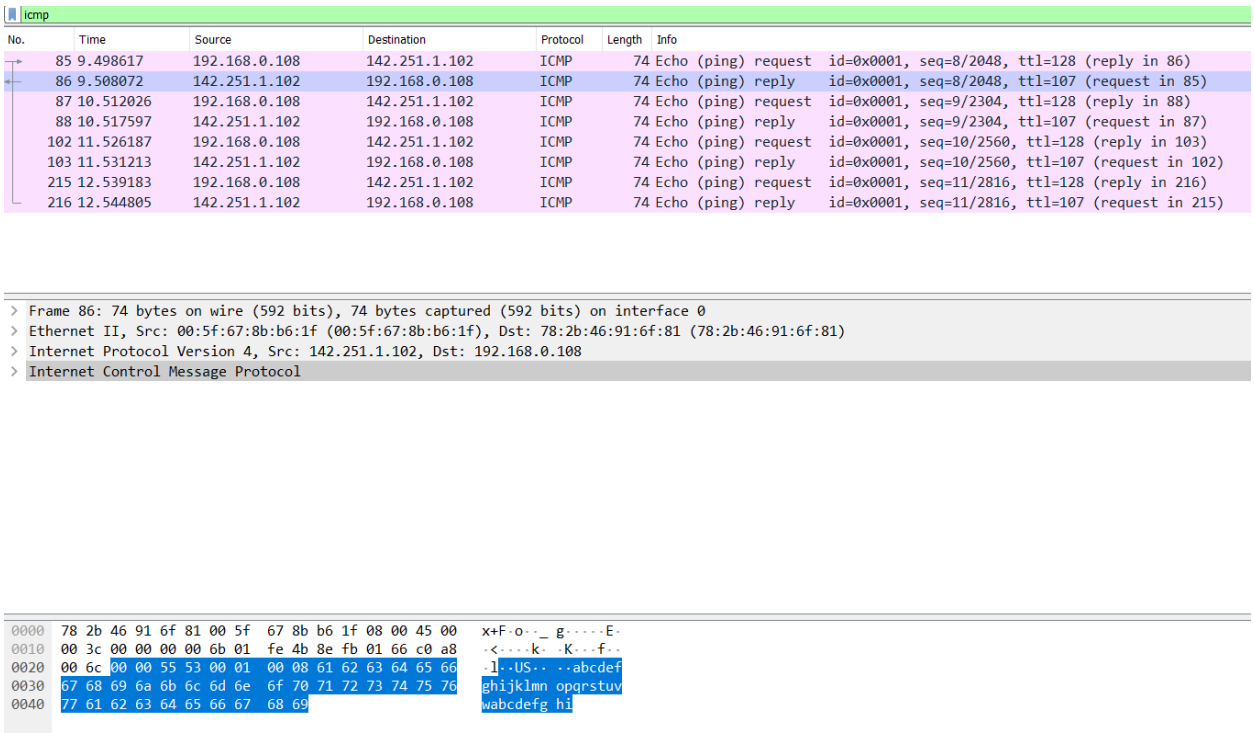

Рис. 2 Трафик утилиты ping

Дамп пакетов

ping.pcapng

1.2 Ответы на вопросы

1.2.1 Имеет ли место фрагментация исходного пакета, какое поле на это указывает?

Да, фрагментация пакета возможна, если размер пакета превышает максимальное значение для передачи в сети (MTU). Поле, указывающее на фрагментацию в IP-заголовке, — это **Flags** и **Fragment Offset**:

- **Flags** включает бит "Don't Fragment" (DF), который запрещает фрагментацию, и бит "More Fragments" (MF), указывающий, что этот фрагмент не является последним.
- **Fragment Offset** показывает позицию данного фрагмента в исходном пакете.

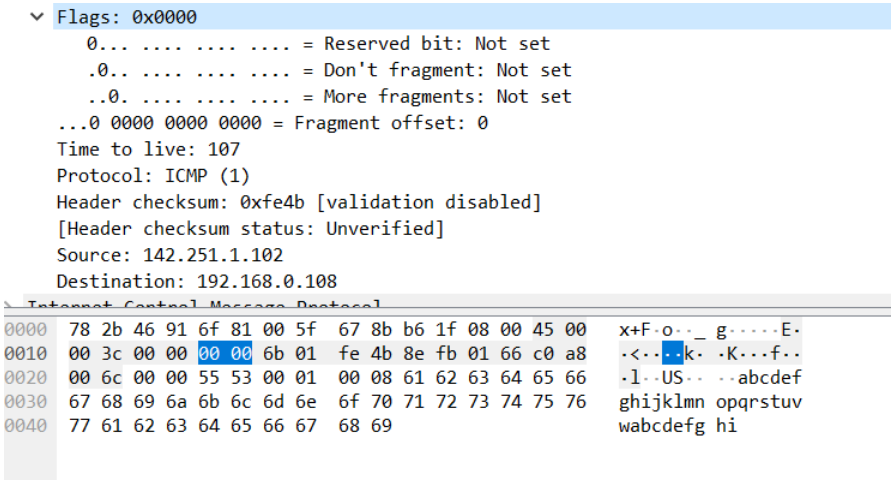

Рис. 3 Фрагментация

1.2.2 Какая информация указывает, является ли фрагмент пакета последним или промежуточным?

Поле **Flags**, а конкретно бит **MF (More Fragments)**, указывает, является ли пакет последним фрагментом. Если **MF=1**, значит это промежуточный фрагмент, и последующие части пакета ещё будут передаваться. Если **MF=0**, это означает, что пакет является последним фрагментом.

1.2.3 Чему равно количество фрагментов при передаче ping-пакетов?

Количество фрагментов зависит от размера ICMP пакета и **MTU** (максимально допустимой длины пакета) в сети. Если размер ICMP пакета превышает MTU (например, 1500 байт для Ethernet), то пакет будет разделён на несколько фрагментов. Количество фрагментов можно рассчитать по формуле:

$$\text{Количество фрагментов} = \frac{\text{Размер пакета}}{MTU} + 1$$

Размер MTU (Maximum Transmission Unit — максимальная длина передаваемого пакета

Формула расчета MTU:

$$MTU = \text{Размер данных} + \text{Размер Заголовков}$$

В нашем случае:

Total Length = 60 байт (общий размер пакета)

Заголовок IPv4 = 20 байт

Заголовок ICMP = 8 байт

Выполнив команду ping -l 100 google.com
Мы послали 100 байт полезной нагрузки, следовательно размер пакета MTU будет равен:
MTU = 20 байт + 8 байт + 100 байт = 128 байт

1.2.4

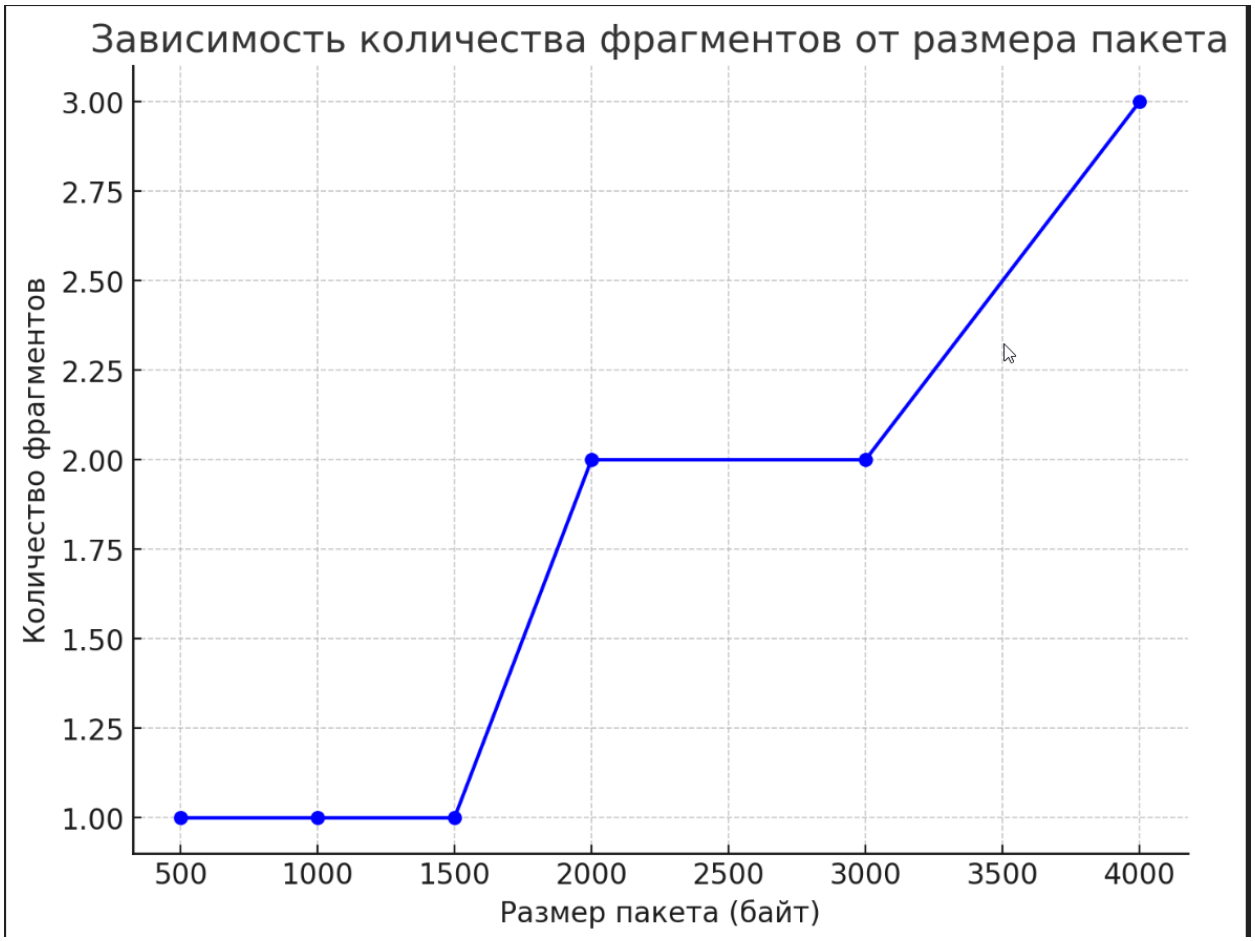

Рис. 4 График зависимости

Как видно, до определенного размера (1500 байт) пакет передается без фрагментации, после чего пакет начинает делиться на несколько фрагментов. Например, при размере пакета 2000 байт пакет разделяется на 2 фрагмента, а при размере 4000 байт — на 3 фрагмента.

1.2.5 Как изменить поле TTL с помощью утилиты ping?

Что такое TTL?

TTL (Time To Live) — это поле в заголовке IP-пакета, которое указывает, сколько маршрутизаторов может пройти пакет, прежде чем его удалят из сети. Это важный механизм для предотвращения бесконечной циркуляции пакетов по сети. Каждый раз, когда пакет проходит через маршрутизатор, значение TTL уменьшается на 1. Если TTL достигает нуля, пакет сбрасывается, а отправителю может быть отправлено сообщение об ошибке с помощью протокола ICMP (Internet Control Message Protocol).

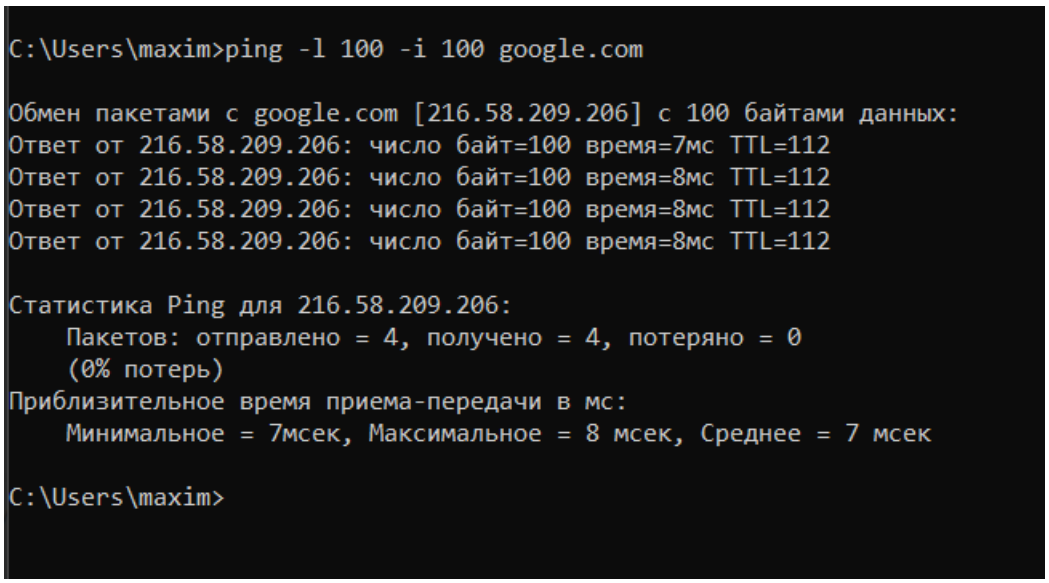

Рис. 5 Ping и измененным TTL

Дамп Pong TTL

1.2.6 Что содержится в поле данных ping-пакета?

Поле данных ping-пакета содержит произвольную информацию, которая используется для проверки целостности передачи данных. Обычно это:

- Последовательность байт, которая может включать ASCII-символы, случайные данные, или тестовые строки.
- Важно: часто это 56 байт (или 64 байта с учётом заголовков ICMP).

Кроме того, это поле может содержать отметки времени, если используется опция измерения времени передачи пакета.

2. АНАЛИЗ ТРАФИКА УТИЛИТЫ TRACERT (TRACEROUTE)

Traceroute — это команда, которая помогает вам увидеть, каким маршрутом данные проходят через интернет от вашего компьютера до какого-то другого устройства или сайта. Она показывает, через какие узлы (или "прыжки) проходят данные, чтобы достичь своего адресата.

```
C:\Users\maxim>tracert -d google.com

Трассировка маршрута к google.com [216.58.209.206]
с максимальным числом прыжков 30:

  1     1 ms     1 ms     1 ms  192.168.0.1
  2     *        *        *     Превышен интервал ожидания для запроса.
  3     9 ms     4 ms     3 ms  93.100.0.131
  4     3 ms     1 ms     2 ms  185.37.128.141
  5     5 ms     4 ms     4 ms  185.37.128.142
  6     *        *        *     Превышен интервал ожидания для запроса.
  7     4 ms     5 ms     5 ms  72.14.205.120
  8     5 ms     3 ms     2 ms  74.125.244.133
  9    13 ms    16 ms    16 ms  142.251.51.187
 10     8 ms    10 ms     8 ms  172.253.65.158
 11    14 ms    11 ms    10 ms  142.250.46.44
 12    12 ms    14 ms    13 ms  172.253.69.241
 13    11 ms     9 ms     9 ms  142.250.227.81
 14     7 ms     7 ms     7 ms  216.58.209.206

Трассировка завершена.
```

Рис. 6 Утилита tracert

icmp

No.	Time	Source	Destination	Protocol	Length	Info
86	6.892377	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=30/7680, ttl=1 (no response found!)
87	6.893541	192.168.0.1	192.168.0.108	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
88	6.894413	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=31/7936, ttl=1 (no response found!)
89	6.895411	192.168.0.1	192.168.0.108	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
90	6.896253	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=32/8192, ttl=1 (no response found!)
91	6.897308	192.168.0.1	192.168.0.108	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
96	7.911811	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=33/8448, ttl=2 (no response found!)
132	11.667622	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=34/8704, ttl=2 (no response found!)
175	15.664271	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=35/8960, ttl=2 (no response found!)
220	19.671123	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=36/9216, ttl=3 (no response found!)
222	19.680044	93.100.0.131	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
223	19.681297	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=37/9472, ttl=3 (no response found!)
224	19.685934	93.100.0.131	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
225	19.687622	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=38/9728, ttl=3 (no response found!)
226	19.690859	93.100.0.131	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
230	20.705871	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=39/9984, ttl=4 (no response found!)
231	20.708623	185.37.128.141	192.168.0.108	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
232	20.710674	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=40/10240, ttl=4 (no response found!)
233	20.712332	185.37.128.141	192.168.0.108	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
234	20.714322	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=41/10496, ttl=4 (no response found!)
235	20.716828	185.37.128.141	192.168.0.108	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
236	21.728195	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=42/10752, ttl=5 (no response found!)
237	21.732985	185.37.128.142	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
238	21.733903	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=43/11008, ttl=5 (no response found!)
239	21.737683	185.37.128.142	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
240	21.738765	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=44/11264, ttl=5 (no response found!)
241	21.743135	185.37.128.142	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
242	22.748671	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=45/11520, ttl=6 (no response found!)
263	26.674307	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=46/11776, ttl=6 (no response found!)
318	30.671753	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=47/12032, ttl=6 (no response found!)
354	34.674931	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=48/12288, ttl=7 (no response found!)
355	34.679203	72.14.205.120	192.168.0.108	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
356	34.680031	192.168.0.108	216.58.209.206	ICMP	106	Echo (ping) request id=0x0001, seq=49/12544, ttl=7 (no response found!)

Рис. 7 Трафик tracert

2.1 Ответы на вопросы

2.1.1 Сколько байт содержится в заголовке IP? Сколько байт содержится в поле данных?

Заголовок IPv4 имеет минимальный размер 20 байт. Однако, если используются дополнительные опции (например, для маршрутизации или диагностики), размер заголовка может увеличиться, максимум до 60 байт.

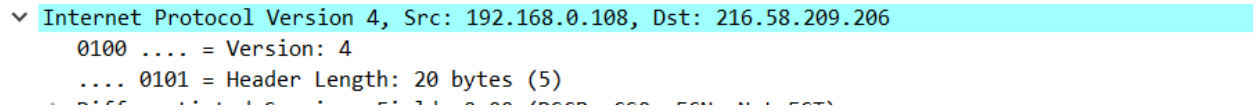

Рис. 8 Размер заголовка

Поле данных (или полезная нагрузка) содержит от 0 до 65,535 байт, включая заголовок. Таким образом, максимально допустимый объем полезной нагрузки без учета заголовка составляет 65,515 байт (если заголовок минимального размера – 20 байт).

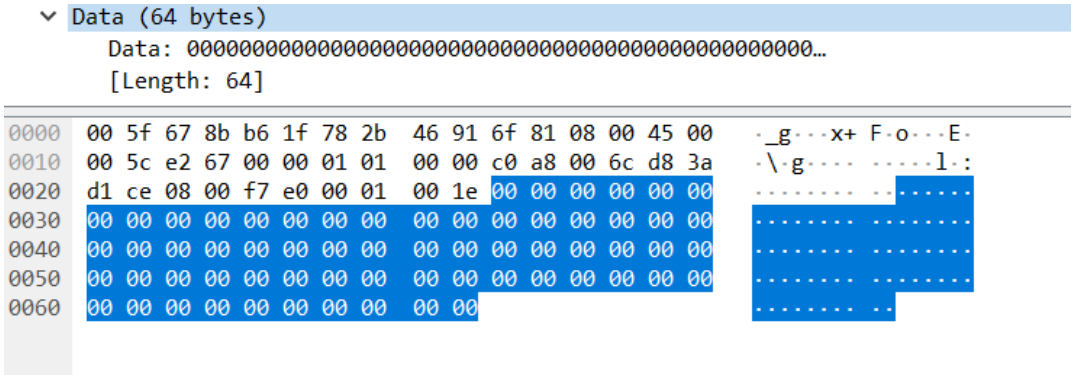

Рис. 9 Размер поля данных

2.1.2 Как и почему изменяется поле TTL в следующих друг за другом ICMP-пакетах tracert?

Поле TTL (Time to Live) используется для предотвращения бесконечной циркуляции пакетов в сети. Каждый маршрутизатор, через который проходит пакет, уменьшает значение TTL на 1. Когда TTL достигает 0, пакет отбрасывается, и отправляется ICMP-сообщение о «время жизни истекло».

Tracert использует этот механизм, отправляя последовательные ICMP-пакеты с увеличением значения TTL на 1 для каждого следующего пакета. Например, первый пакет имеет TTL = 1, второй – TTL = 2, и так далее. Это позволяет утилите определить каждый маршрутизатор на пути к конечному узлу

2.1.3 Чем отличаются ICMP-пакеты, генерируемые утилитой tracert, от ICMP-пакетов, генерируемых утилитой ping?

Ping отправляет ICMP Echo Request пакеты и ожидает ICMP Echo Reply от целевого узла. Ping напрямую взаимодействует только с конечной точкой, не отслеживая промежуточные узлы. **Tracert**, в свою очередь, отправляет ICMP Echo Request пакеты с постепенно увеличивающимся TTL, и получает ICMP Time Exceeded сообщения от промежуточных маршрутизаторов до тех пор, пока не дойдет до целевого узла, от которого будет получен ICMP Echo Reply.

2.1.4 Чем отличаются полученные пакеты «ICMP reply» от «ICMP error» и зачем нужны оба этих типа ответов?

ICMP Reply:

- **Что это за пакет**: ICMP Echo Reply — это сообщение, которое отправляется в ответ на ICMP Echo Request. Этот процесс часто используется утилитами, такими как `ping`, для проверки доступности сетевых узлов.
- **Зачем нужен**: Когда хост получает ICMP Echo Request, он отправляет ICMP Echo Reply, подтверждая, что пакет достиг адресата. Это полезно для измерения времени доставки пакетов и проверки работоспособности узлов в сети.

ICMP Error:

- Что это за пакет: ICMP Error — это категория ICMP-сообщений, которая включает различные типы ошибок, возникающих при попытке доставки пакетов. Примеры таких сообщений:
    - **Time Exceeded** (Время жизни истекло): Уведомление о том, что пакет был отброшен маршрутизатором, так как значение TTL (время жизни) пакета достигло нуля.
    - **Destination Unreachable** (Цель недоступна): Сообщение о том, что пакет не может быть доставлен по назначению по различным причинам, таким как отсутствие маршрута к узлу.
    - **Fragmentation Needed** (Требуется фрагментация): Сообщение о том, что пакет слишком велик и не может быть передан без фрагментации, а флаг Don't Fragment (DF) установлен, что запрещает фрагментацию.

Зачем нужны оба типа ICMP-сообщений:

1. **ICMP Reply**: Это сообщение подтверждает, что пакет успешно доставлен до целевого узла. Оно используется для базовой проверки доступности и измерения задержек в сети. Например, утилита ping использует ICMP Echo Request/Reply для диагностики.
2. **ICMP Error**: Сообщения об ошибках ICMP помогают выявлять и диагностировать проблемы с маршрутизацией и доставкой пакетов. Например, при использовании утилиты tracert ICMP Time Exceeded сообщения помогают отслеживать каждый хоп на маршруте, если TTL пакета истек на промежуточном маршрутизаторе.

Таким образом, оба типа сообщений критически важны:

- **ICMP Reply** используется для проверки успешности связи и оценки задержки.
- **ICMP Error** помогает выявить проблемы на маршруте (например, перегрузку маршрутизаторов, превышение TTL или недоступность хоста), что делает ICMP незаменимым инструментом для диагностики и устранения проблем в сети.

2.1.5 Что изменится в работе tracert, если убрать ключ “-d”? Какой дополнительный трафик при этом будет генерироваться?

Ключ "-d":

- Значение ключа: Ключ -d отключает разрешение IP-адресов в доменные имена. Это означает, что tracert будет отображать только IP-адреса узлов, через которые проходит пакет, без попытки преобразовать их в соответствующие доменные имена с использованием DNS.

Как работает tracert с ключом "-d":

- Без разрешения имен: когда используется ключ -d, tracert не отправляет DNS-запросы для разрешения IP-адресов узлов на маршруте в доменные имена. Это ускоряет работу команды и уменьшает сетевой трафик, так как DNS-запросы не выполняются.

Что изменится в работе tracert без ключа "-d":

- Разрешение доменных имен: без ключа -d, tracert будет пытаться разрешить каждый IP-адрес на маршруте в соответствующее доменное имя. Это означает, что для каждого узла, через который проходит пакет, утилита отправит DNS-запрос к DNS-серверу, что может занять больше времени и создать дополнительную задержку при отображении маршрута.

Дополнительный трафик:

- DNS-запросы: Каждый раз, когда tracert пытается разрешить IP-адрес в доменное имя, оно генерирует DNS-запросы. Эти запросы отправляются к DNS-серверам для каждого промежуточного маршрутизатора, что увеличивает количество передаваемого трафика.

- Влияние на производительность: без использования ключа -d, общее время выполнения команды может увеличиться из-за задержек, вызванных разрешением имен. Особенно это заметно при медленном DNS-сервере или наличии большого количества маршрутизаторов на пути.

Пример работы без ключа "-d":

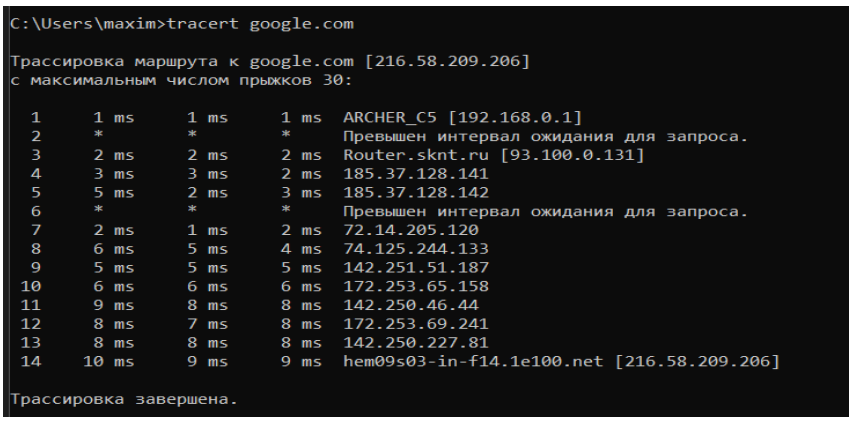

Рис. 10 Пример работы без ключа "-d"

Допустим, tracert проходит через 10 маршрутизаторов. Без ключа -d утилита выполнит по одному DNS-запросу для каждого маршрутизатора, чтобы разрешить его IP-адрес в доменное имя. Это может привести к дополнительным задержкам и

увеличению объема сетевого трафика, особенно если DNS-серверы находятся далеко или обрабатывают запросы медленно.

Итог: Ключ -d оптимизирует работу команды tracert, уменьшая время выполнения и объем генерируемого трафика за счет отключения DNS-запросов.

3. АНАЛИЗ HHTP-ТРАФИКА

HTTP (HyperText Transfer Protocol) — это протокол, который используется для передачи данных между веб-браузером (или другим клиентом) и веб-сервером. Когда вы открываете веб-страницу, ваш браузер отправляет запрос серверу по протоколу HTTP, а сервер возвращает запрашиваемую информацию (например, текст, изображения или видео) обратно.

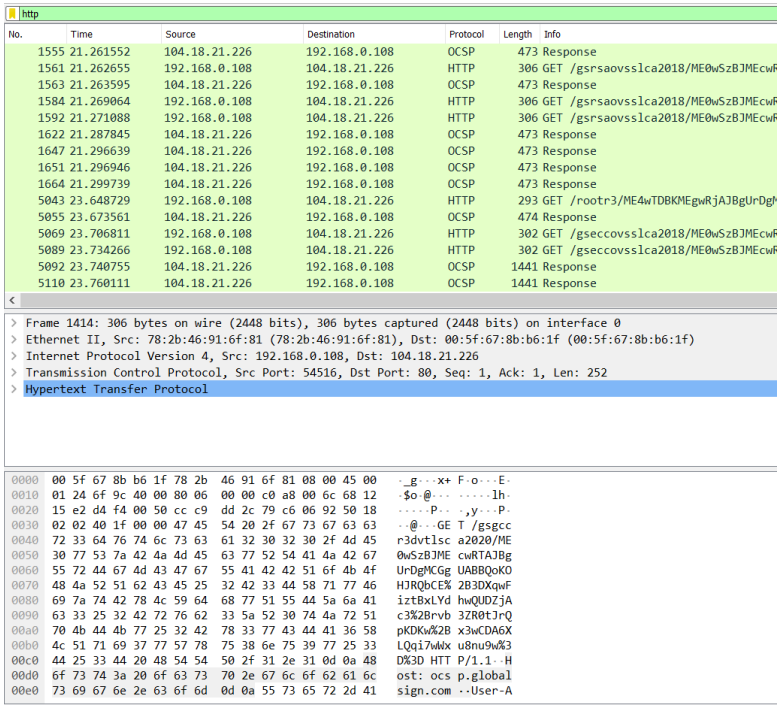

Рис. 11 HTTP трафик

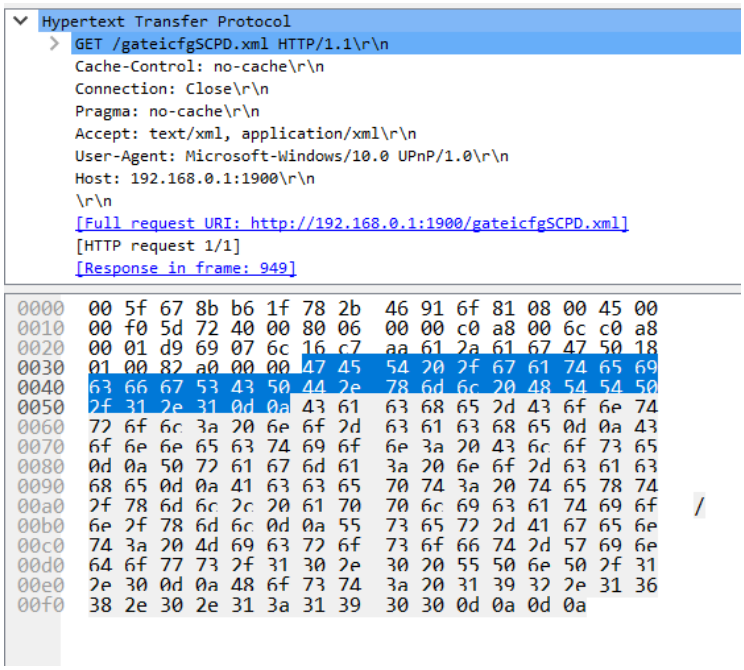

Рис. 12 Пример GET запроса

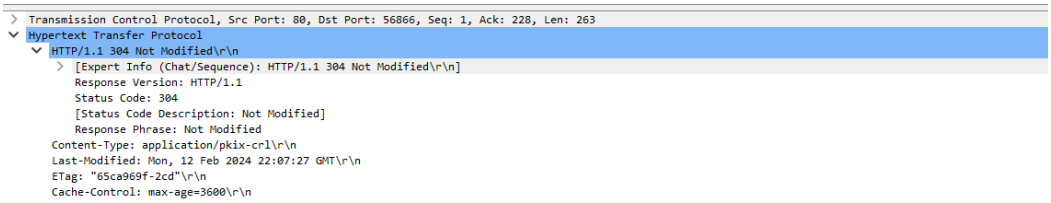

Рис. 13 CONDITIONAL GET

1. Первичный запрос на страницу (Initial GET Request):

Когда пользователь впервые посещает страницу, браузер отправляет HTTP GET-запрос серверу для получения всех необходимых ресурсов. В этом случае запрос и ответ включают следующие элементы:

- **Запрос**: Браузер посылает запрос на сервер с указанием URL страницы и дополнительной информацией (заголовки HTTP, включая информацию о поддерживаемых типах контента, языке и других параметрах).
- **Ответ**: Сервер возвращает полный ответ, включающий:
    - Статус код (обычно 200 OK).
    - Полное содержимое страницы (HTML-код).
    - Дополнительные ресурсы, такие как CSS, JavaScript, изображения и другие файлы, которые браузер загружает отдельно.

- Заголовки HTTP, в том числе заголовки, связанные с кэшированием (например, Cache-Control и Expires), которые указывают, как долго сохранять данные в локальном кэше браузера.

2. Вторичный запрос-обновление (Subsequent GET Request):

При повторном запросе или обновлении страницы браузер использует локально сохраненные ресурсы, если это возможно, чтобы уменьшить время загрузки и сетевую нагрузку. Этот процесс может разворачиваться несколькими способами:

- **Кэширование (HTTP 304 Not Modified)**: Если ресурсы были закэшированы при первичном запросе, браузер при вторичном запросе посылает заголовки с информацией о дате последней модификации файлов (If-Modified-Since). Если сервер видит, что данные не изменились, он возвращает статус **304 Not Modified** вместо полного содержимого страницы, указывая, что браузер может использовать кэшированные данные.
- **Обновление контента (HTTP 200 OK)**: Если ресурсы обновились с момента первичного запроса, сервер вернет полную версию обновленного ресурса с новым контентом и заголовками HTTP, такими как Last-Modified или ETag, для кэширования на будущее.

Основные различия:

- **Первичный запрос** всегда требует полной загрузки ресурсов с сервера.
- **Вторичный запрос** использует механизм кэширования, чтобы загрузить только те данные, которые были изменены, а неизмененные ресурсы предоставляются из кэша. Это оптимизирует работу браузера, снижая время загрузки и нагрузку на сеть.

4. АНАЛИЗ DNS трафика

DNS (Domain Name System) — это протокол, который используется для преобразования имён доменов (например, google.com) в IP-адреса (например, 142.251.214.142), которые понятны компьютерам и другим сетевым устройствам.

Основные характеристики DNS:

1. **Разрешение имён**: DNS позволяет пользователям вводить понятные имена доменов, а не сложные для запоминания IP-адреса. Когда вы вводите адрес сайта в браузере, DNS-протокол находит соответствующий IP-адрес, чтобы установить соединение с сервером, на котором находится этот сайт.
2. **Иерархическая структура**: DNS организован как иерархическая система, где доменные имена разбиваются на части, называемые доменами. Верхний уровень — это домены верхнего уровня (TLD), такие как .com, .org, .ru. Ниже следуют поддомены и сам домен.
3. **DNS-серверы**: В интернете существует множество DNS-серверов, которые хранят информацию о доменах и их IP-адресах. Запросы DNS от пользователя передаются от одного сервера к другому по цепочке, пока не будет найден нужный IP-адрес.
4. **Кэширование**: Для ускорения процесса разрешения имён DNS-серверы и локальные устройства (например, компьютеры и роутеры) могут сохранять результаты предыдущих запросов в кэше, чтобы не делать повторные запросы к серверу.

Пример работы DNS:

1. Пользователь вводит в браузер адрес сайта, например, google.com.
2. Браузер отправляет запрос на DNS-сервер для получения IP-адреса этого домена.
3. Если DNS-сервер не имеет нужной информации в своём кэше, он обращается к другим DNS-серверам, пока не найдёт IP-адрес.
4. Как только IP-адрес найден, он возвращается в браузер, который устанавливает соединение с сервером сайта по этому IP-адресу.

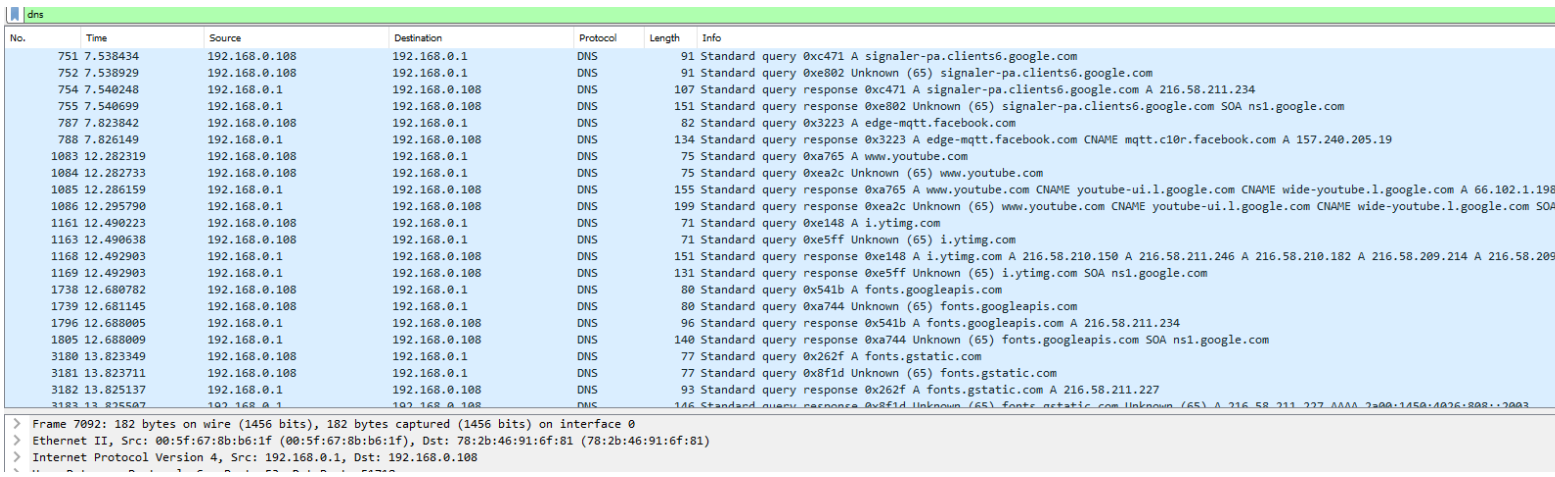

Рис. 14 DNS трафик

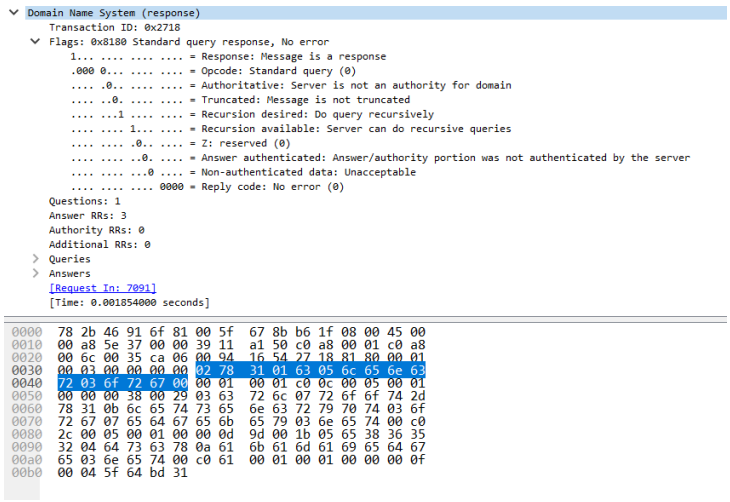

Рис. 15 Структура DNS

4.1 Ответы на вопросы

4.1.1 Почему адрес DNS-запроса отличается от адреса посещаемого сайта?

Когда пользователь вводит URL (например, www.google.com) в браузере, компьютеру нужно узнать IP-адрес, который связан с этим доменом. Для этого отправляется DNS-запрос на DNS-сервер (например, сервер провайдера или Google DNS).

DNS-сервер и сервер сайта — это разные устройства. DNS-сервер отвечает за преобразование доменного имени в IP-адрес, а затем браузер уже подключается к нужному серверу по полученному IP.

4.1.2 Виды DNS-запросов

Существует несколько типов DNS-запросов:

- **Рекурсивный запрос**: DNS-сервер сам находит окончательный ответ, обращаясь к другим серверам.
- **Итеративный запрос**: Если DNS-сервер не знает ответа, он отправляет информацию о другом сервере, и браузер продолжает искать нужные данные.
- **Обратный запрос**: Этот запрос используется для получения доменного имени по IP-адресу.
- **Кэш-запрос**: Если DNS-сервер уже знает ответ, он выдаёт его из кэша, не запрашивая информацию у других серверов.

4.1.3 Когда нужно делать отдельные DNS-запросы для изображений на сайте?

Если ресурсы сайта (например, изображения) хранятся на других доменах, браузеру нужно выполнять дополнительные DNS-запросы для их загрузки. Это происходит в следующих случаях:

- **Использование CDN**: Если изображения хранятся на серверах CDN (например, images.google-cdn.com), требуется отдельный запрос к этому домену.

- **Реклама и трекеры**: Элементы рекламы или отслеживания могут загружаться с других доменов.
- **Сторонние мультимедийные ресурсы**: Если изображения размещены на внешних хостингах, браузеру также потребуется выполнить независимые DNS-запросы.

5. АНАЛИЗ ARP-ТРАФИКА

ARP (Address Resolution Protocol) — это сетевой протокол, используемый для определения MAC-адреса устройства в локальной сети на основе его IP-адреса. Поскольку устройства в сети взаимодействуют через IP-адреса, а передача данных в локальной сети осуществляется через MAC-адреса, ARP обеспечивает связь между этими двумя типами адресов.

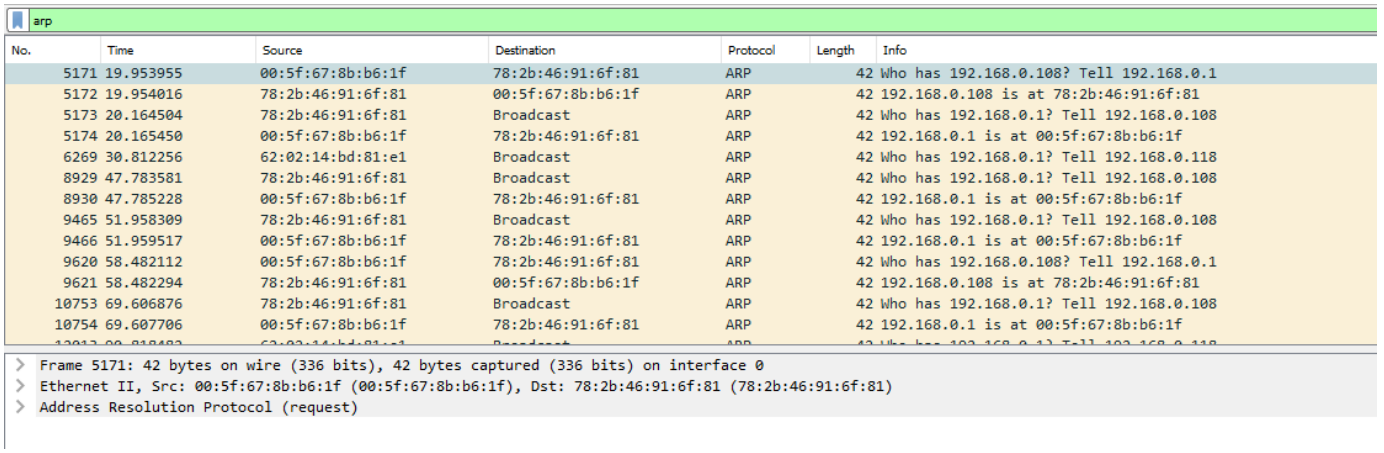

Рис. 16 ARP трафик

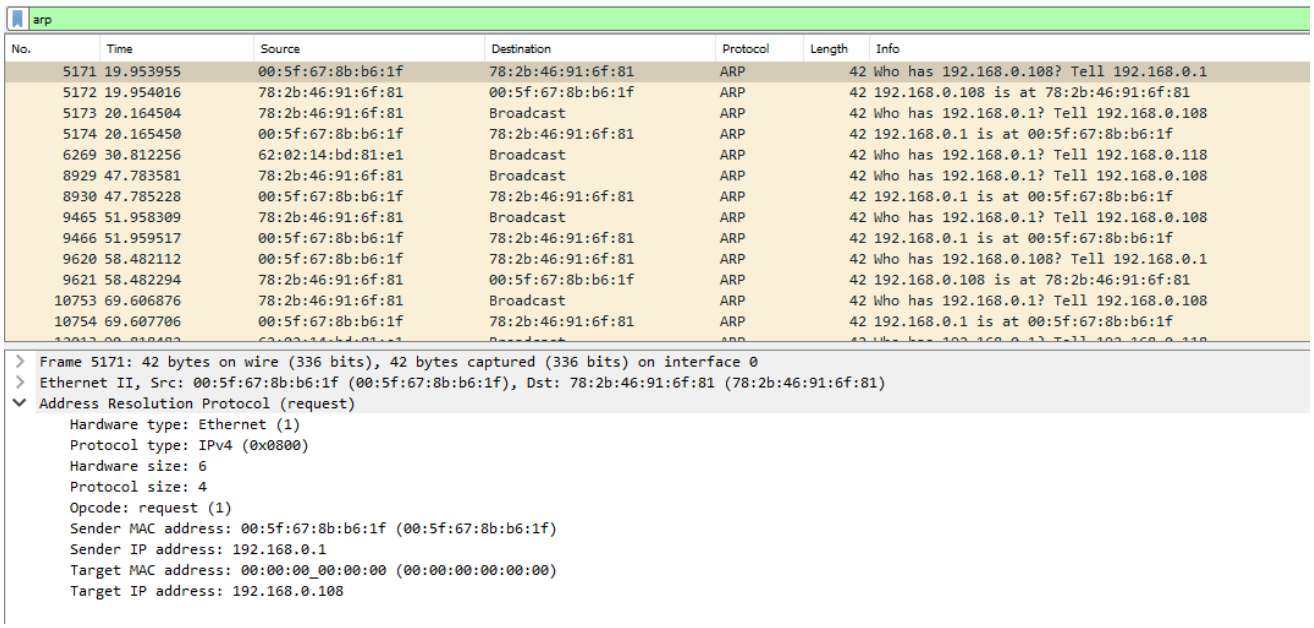

Рис. 17 ARP запрос

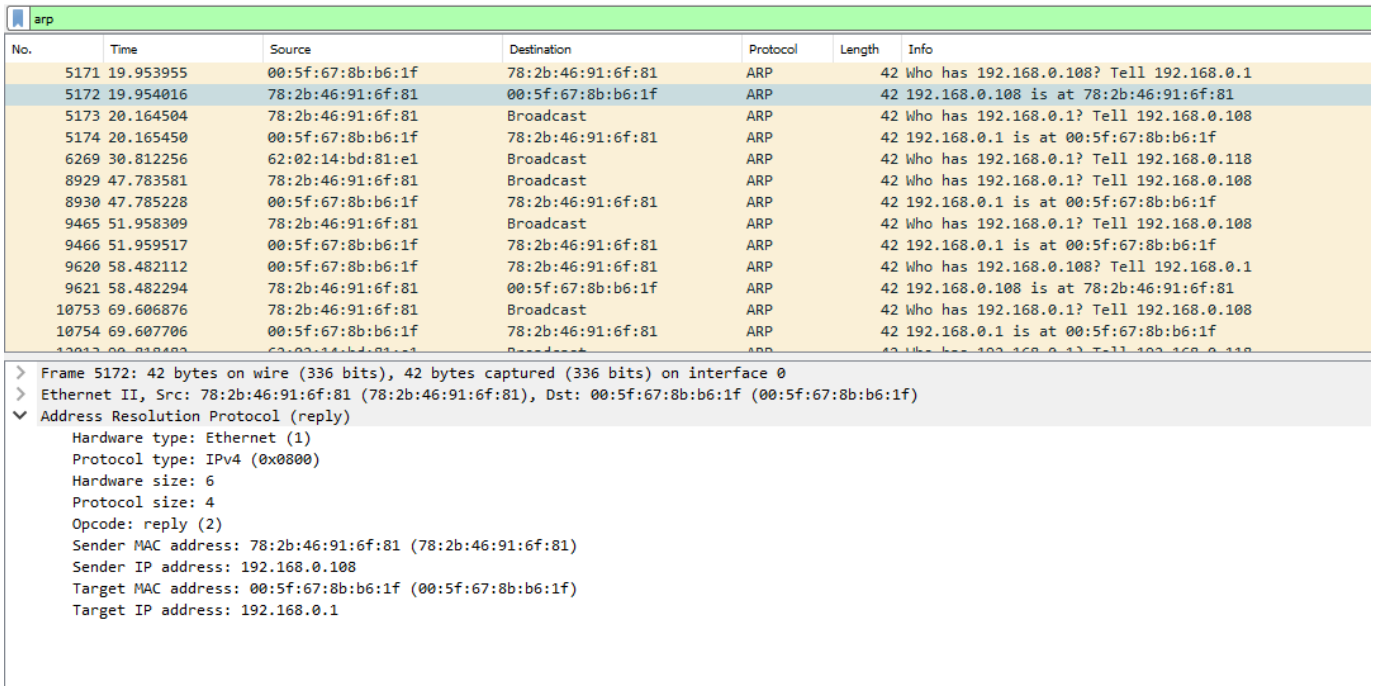

Рис. 18 ARP ответ

5.1 Ответы на вопросы

5.1.1 Какие MAC-адреса присутствуют в захваченных пакетах ARP-протокола? Что означают эти адреса? Какие устройства они идентифицируют?

В захваченных ARP-пакетах могут присутствовать два ключевых MAC-адреса:

- **MAC-адрес отправителя (Source MAC address)**: Это MAC-адрес устройства, отправляющего ARP-запрос или ARP-ответ. Этот адрес уникально идентифицирует сетевую карту (NIC) устройства, с которого исходит запрос или ответ.
- **MAC-адрес получателя (Destination MAC address)**: Это либо широковещательный адрес **FF:FF:FF:FF:FF**

(если это ARP-запрос, который рассылается всем устройствам в сети), либо MAC-адрес устройства, на которое направлен ARP-ответ.

5.1.2 Какие MAC-адреса присутствуют в захваченных HTTP-пакетах и что означают эти адреса? Какие устройства они идентифицируют?

В захваченных HTTP-пакетах также будут присутствовать два MAC-адреса:

- **MAC-адрес отправителя (Source MAC address): Это MAC-адрес устройства, инициирующего HTTP-запрос (например, вашего компьютера или маршрутизатора). Этот адрес указывает, откуда был отправлен запрос.**
- **MAC-адрес получателя (Destination MAC address): Это MAC-адрес следующего сетевого устройства на пути к серверу (например, роутера или шлюза). Если HTTP-сервер находится в локальной сети, это может быть MAC-адрес сервера, на который идёт запрос. В случае маршрутизации через интернет, это будет адрес промежуточного маршрутизатора.**

5.1.3 Для чего ARP-запрос содержит IP-адрес источника?

IP-адрес источника в ARP-запросе необходим для того, чтобы получатель ARP-ответа знал, к какому IP-адресу нужно привязать MAC-адрес, и чтобы отправить обратно ARP-ответ. Когда устройство отправляет ARP-запрос, оно сообщает свой IP-адрес, чтобы другой участник сети мог связать его с соответствующим MAC-адресом.

Эта информация позволяет устройству, получающему ARP-запрос, знать не только чей MAC-адрес требуется, но и кому следует отправить ответ.

6. АНАЛИЗ ТРАФИКА NSLOOKUP

nslookup — это сетевая утилита, которая используется для проверки и диагностики работы DNS (Domain Name System). Она позволяет узнать, какой IP-адрес соответствует доменному имени, или, наоборот, какое доменное имя соответствует IP-адресу.

```
C:\Users\maxim>nslookup google.com
╤xËтxË:  UnKnown
Address:  192.168.0.1

Не заслуживающий доверия ответ:
╚ь :     google.com
Addresses:  2a00:1450:4026:803::200e
          216.58.209.206


C:\Users\maxim>
```

Рис. 19 Использование nslookup

```
C:\Users\maxim>nslookup -type=NS google.com
╤xËтxË:  UnKnown
Address:  192.168.0.1

Не заслуживающий доверия ответ:
google.com      nameserver = ns1.google.com
google.com      nameserver = ns2.google.com
google.com      nameserver = ns3.google.com
google.com      nameserver = ns4.google.com

ns1.google.com  internet address = 216.239.32.10
ns2.google.com  internet address = 216.239.34.10
ns3.google.com  internet address = 216.239.36.10
ns4.google.com  internet address = 216.239.38.10
ns1.google.com  AAAA IPv6 address = 2001:4860:4802:32::a
ns2.google.com  AAAA IPv6 address = 2001:4860:4802:34::a
ns3.google.com  AAAA IPv6 address = 2001:4860:4802:36::a
ns4.google.com  AAAA IPv6 address = 2001:4860:4802:38::a

C:\Users\maxim>
```

Рис. 20 Использование nslookup с параметром

(dns) && (ip.addr == 192.168.0.108)

No.	Time	Source	Destination	Protocol	Length	Info
241	4.039599	192.168.0.108	192.168.0.1	DNS	84	Standard query 0x0001 PTR 1.0.168.192.in-addr.arpa
242	4.041320	192.168.0.1	192.168.0.108	DNS	154	Standard query response 0x0001 No such name PTR 1.0.168.192.in-addr.arp
243	4.043534	192.168.0.108	192.168.0.1	DNS	70	Standard query 0x0002 A google.com
244	4.044942	192.168.0.1	192.168.0.108	DNS	86	Standard query response 0x0002 A google.com A 216.58.209.206
245	4.047559	192.168.0.108	192.168.0.1	DNS	70	Standard query 0x0003 AAAA google.com
246	4.049152	192.168.0.1	192.168.0.108	DNS	98	Standard query response 0x0003 AAAA google.com AAAA 2a00:1450:4026:803:
618	9.821176	192.168.0.108	192.168.0.1	DNS	84	Standard query 0x0001 PTR 1.0.168.192.in-addr.arpa
619	9.823044	192.168.0.1	192.168.0.108	DNS	154	Standard query response 0x0001 No such name PTR 1.0.168.192.in-addr.arp
620	9.824970	192.168.0.108	192.168.0.1	DNS	70	Standard query 0x0002 NS google.com
621	9.826740	192.168.0.1	192.168.0.108	DNS	318	Standard query response 0x0002 NS google.com NS ns1.google.com NS ns3.g
638	11.811986	192.168.0.108	192.168.0.1	DNS	94	Standard query 0x5ad3 A p2p-sto1.discovery.steamserver.net
639	11.814751	192.168.0.1	192.168.0.108	DNS	142	Standard query response 0x5ad3 A p2p-sto1.discovery.steamserver.net A 1
2090	46.365470	192.168.0.108	192.168.0.1	DNS	82	Standard query 0xdd78 A edge-mqtt.facebook.com

Рис. 21 Трафик nslookup

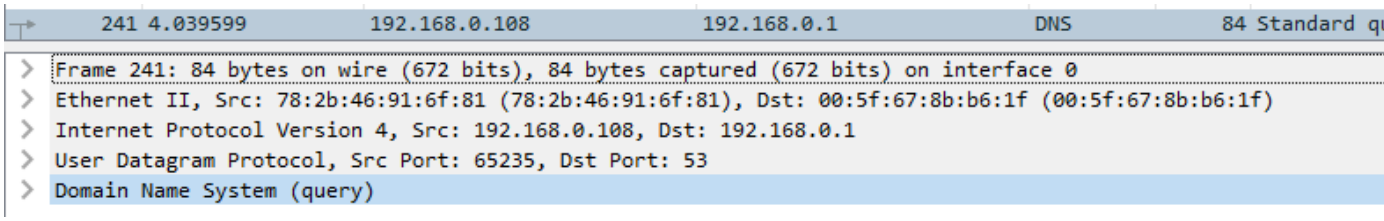

Рис. 22 Запрос DNS при работе nslookup

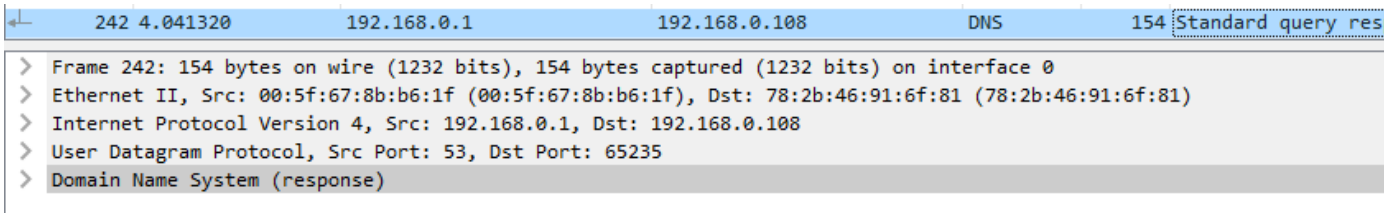

Рис. 23 Ответ DNS при работе nslookup

6.1 Ответы на вопросы

6.1.1 Чем различается трасса трафика в п.2 и п.4?

В пункте 2 утилита nslookup выполняет стандартный DNS-запрос для получения A-записи, которая возвращает IP-адрес хоста, связанного с указанным доменным именем. В пункте 4 команда nslookup -type=NS запрашивает NS-записи, которые предоставляют имена DNS-серверов, ответственных за управление зоной домена. Эти записи не содержат IP-адреса самого сайта, а указывают на серверы, управляющие его DNS-зоной.

6.1.2 Что содержится в поле «Answers» DNS-ответа?

Поле Answers в DNS-ответе содержит информацию о запрашиваемом ресурсе. Для стандартного DNS-запроса (пункт 2) это будет IP-адрес, указанный в A-записи домена. Для запроса на NS-записи (пункт 4) в поле Answers будут перечислены имена серверов, ответственных за управление зоной указанного домена.

6.1.3 Каковы имена серверов, возвращающих авторитативный отклик?

Авторитативный DNS-ответ — это ответ, который приходит от DNS-сервера, официально отвечающего за управление конкретной зоной домена. Такой сервер содержит самую актуальную и достоверную информацию о домене, поскольку он непосредственно обслуживает эту зону.

Имена серверов, предоставляющих авторитативные ответы, находятся в поле Authority. Эти серверы являются ответственными за домен, к которому относится запрашиваемый ресурс. Имена таких серверов можно увидеть в ответах на запросы NS-записей, а также в случае с A-запросами, если они обрабатываются авторитативным DNS-сервером.

7. АНАЛИЗ FTP ТРАФИКА

FTP (File Transfer Protocol) — это сетевой протокол, используемый для передачи файлов между устройствами через сеть, например, через интернет. FTP позволяет пользователям загружать файлы на сервер (upload) или скачивать файлы с сервера (download), а также управлять файлами на удалённом сервере.

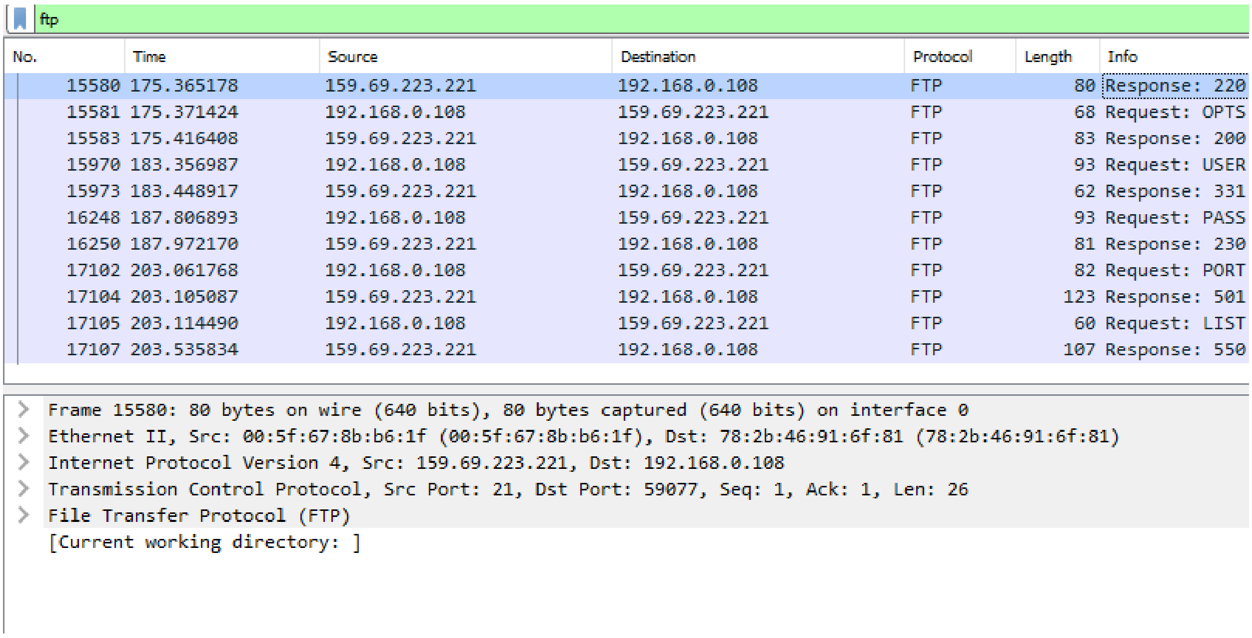

Рис. 24 FTP трафик

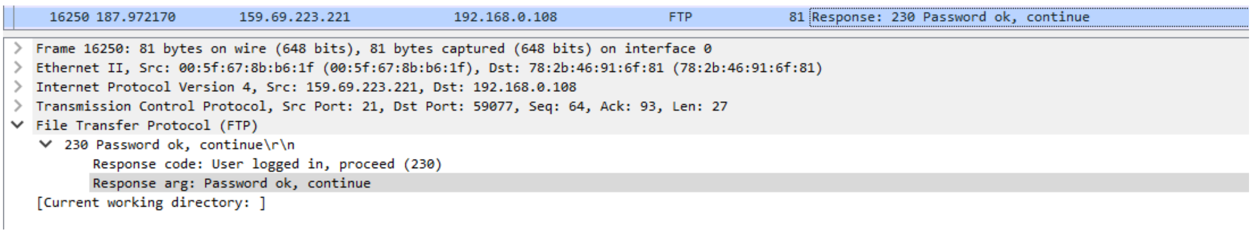

Рис. 25 Ответ FTP сервера

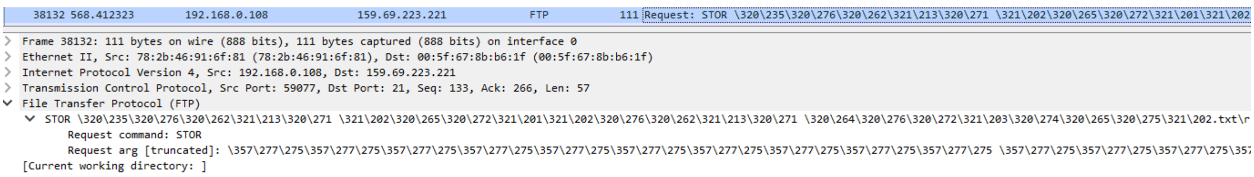

Рис. 26 Загрузка данных на FTP сервер

7.1 Ответы на вопросы

7.1.1 Сколько байт данных содержится в пакете FTP-DATA?

Размер данных, передаваемых в пакете **FTP-DATA**, варьируется в зависимости от того, сколько данных передаётся в одном сегменте TCP. Обычно размер пакета ограничен **MTU (Maximum Transmission Unit)** сети, которая в стандартных сетях Ethernet составляет 1500 байт. Однако из этих 1500 байт часть используется для заголовков TCP и IP, поэтому в одном пакете может передаваться до **1460 байт данных**. Для точного определения размера данных в пакете FTP-DATA нужно анализировать конкретные захваченные пакеты.

7.1.2 Как выбирается порт транспортного уровня, который используется для передачи FTP-пакетов?

Для передачи FTP-пакетов используются порты на транспортном уровне (уровень TCP):

- **Порт сервера**: По умолчанию, **21** порт используется для командного соединения FTP (управляющий канал), а **20** порт — для передачи данных (в активном режиме FTP).
- **Порт клиента**: Когда клиент устанавливает соединение с сервером, он использует случайный незарезервированный порт (порт с номером выше 1023), который выбирается динамически операционной системой. В пассивном режиме FTP сервер также выбирает случайный порт для передачи данных, и клиент подключается к этому порту.

7.1.3 Чем отличаются пакеты FTP от FTP-DATA?

Пакеты **FTP** и **FTP-DATA** выполняют разные функции:

- **FTP-пакеты (управляющий канал)**: Эти пакеты передают команды и ответы между клиентом и сервером. Например, клиент отправляет команду для входа на сервер (USER, PASS), получения списка файлов (LIST) или загрузки файла (RETR). Эти команды передаются через **порт 21** (по умолчанию). Содержимое таких пакетов — это текстовые команды и ответы.
- **FTP-DATA-пакеты (канал передачи данных)**: Эти пакеты содержат непосредственно передаваемые данные — файлы, которые загружаются или скачиваются с сервера. Передача данных происходит по отдельному соединению, которое может использовать **порт 20** (в активном режиме) или другой порт, выбранный сервером (в пассивном режиме). Содержимое таких пакетов — это не команды, а сами данные файлов.

Основные отличия:

- **FTP**: Управляет соединением и выполняет команды. Пакеты содержат команды и ответы.
- **FTP-DATA**: Передаёт файлы и другие данные. Пакеты содержат саму информацию, передаваемую между клиентом и сервером.

8. АНАЛИЗ DHCP ТРАФИКА

DHCP (Dynamic Host Configuration Protocol) — это сетевой протокол, который автоматически распределяет IP-адреса и другую сетевую информацию (например, адреса шлюзов, DNS-серверов) для устройств в сети. Благодаря DHCP, устройства, такие как компьютеры, смартфоны и принтеры, могут подключаться к сети без необходимости ручной настройки IP-адресов.

```
Адаптер беспроводной локальной сети Беспроводная сеть:

   DNS-суффикс подключения . . . . . :
   Описание. . . . . . . . . . . . . : Intel(R) Wi-Fi 6 AX201 160MHz
   Физический адрес. . . . . . . . . : 78-2B-46-91-6F-81
   DHCP включен. . . . . . . . . . . : Да
   Автонастройка включена. . . . . . : Да
   Локальный IPv6-адрес канала . . . : fe80::437e:4d5b:a423:2fe4%20(Основной)
   IPv4-адрес. . . . . . . . . . . . : 192.168.0.108(Основной)
   Маска подсети . . . . . . . . . . : 255.255.255.0
   Аренда получена. . . . . . . . . . : 6 октября 2024 г. 18:40:19
   Срок аренды истекает. . . . . . . . . . : 8 октября 2024 г. 0:14:26
   Основной шлюз. . . . . . . . . : 192.168.0.1
   DHCP-сервер. . . . . . . . . . . : 192.168.0.1
   IAID DHCPv6 . . . . . . . . . . . : 561523526
   DUID клиента DHCPv6 . . . . . . . : 00-01-00-01-27-47-6C-97-3C-7C-3F-5D-8F-6A
   DNS-серверы. . . . . . . . . . . : 192.168.0.1
                                       0.0.0.0
   NetBios через TCP/IP. . . . . . . : Включен
```

Рис. 27 IP адрес компьютера, выданный DHCP сервером

dhcp

No.	Time	Source	Destination	Protocol	Length	Info
81495	1216.620817	192.168.0.108	192.168.0.1	DHCP	342	DHCP Release - Transaction ID 0x2f8b766b
81627	1223.005110	0.0.0.0	255.255.255.255	DHCP	344	DHCP Discover - Transaction ID 0xe6238ba5
81628	1223.008654	192.168.0.1	192.168.0.108	DHCP	590	DHCP Offer - Transaction ID 0xe6238ba5
81629	1223.009455	0.0.0.0	255.255.255.255	DHCP	370	DHCP Request - Transaction ID 0xe6238ba5
81630	1223.009573	192.168.0.1	192.168.0.108	DHCP	590	DHCP Offer - Transaction ID 0xe6238ba5
81640	1223.515936	192.168.0.1	192.168.0.108	DHCP	590	DHCP ACK - Transaction ID 0xe6238ba5
81641	1223.516939	192.168.0.1	192.168.0.108	DHCP	590	DHCP ACK - Transaction ID 0xe6238ba5

Рис. 28 DHCP трафик

```
> Ethernet II, Src: 00:5f:67:8b:b6:1f (00:5f:67:8b:b6:1f), Dst: 78:2b:46:91:6f:81 (78:2b:46:
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.108
> User Datagram Protocol, Src Port: 67, Dst Port: 68
v Dynamic Host Configuration Protocol (Offer)
    Message type: Boot Reply (2)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0xe6238ba5
    Seconds elapsed: 0
  > Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0
    Your (client) IP address: 192.168.0.108
    Next server IP address: 0.0.0.0
    Relay agent IP address: 0.0.0.0
    Client MAC address: 78:2b:46:91:6f:81 (78:2b:46:91:6f:81)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  > Option: (53) DHCP Message Type (Offer)
  > Option: (54) DHCP Server Identifier (192.168.0.1)
  > Option: (51) IP Address Lease Time
  > Option: (1) Subnet Mask (255.255.255.0)
```

Рис. 29 Строение Offer

Рис. 30 Принцип работы DHCP

Описание пакетов:

1. **DHCP DISCOVER**:
    - Исходит от клиента к серверу.
    - Порты: Источник — **68** (клиент), Назначение — **67** (сервер).
2. **DHCP OFFER**:
    - Отправляется от DHCP-сервера к клиенту.
    - Порты: Источник — **67** (сервер), Назначение — **68** (клиент).
3. **DHCP REQUEST**:
    - Исходит от клиента к серверу с запросом предложенного IP.
    - Порты: Источник — **68** (клиент), Назначение — **67** (сервер).
4. **DHCP ACK**:
    - Отправляется от сервера к клиенту с подтверждением.
    - Порты: Источник — **67** (сервер), Назначение — **68** (клиент).

8.1 Ответы на вопросы

8.1.1 Чем различаются пакеты «DHCP Discover» и «DHCP Request»?

- **DHCP Discover** — это первый пакет, который отправляет клиент для поиска DHCP-сервера в сети. В этом пакете клиент указывает, что он ищет доступные серверы и просит назначить ему IP-адрес. Клиент обычно использует **широковещательный** (broadcast) запрос, и в поле IP-адреса источника стоит **0.0.0.0**, так как у клиента ещё нет IP-адреса.
- **DHCP Request** — это пакет, который клиент отправляет после получения предложения от DHCP-сервера (DHCPOFFER). В этом пакете клиент запрашивает конкретный IP-адрес, предложенный сервером, подтверждая, что он хочет его использовать. Пакет также может использоваться для продления существующей аренды IP-адреса.

8.1.2 Как и почему менялись MAC- и IP-адреса источника и назначения в переданных DHCP-пакетах?

- **MAC-адрес клиента остаётся неизменным на протяжении всего обмена, так как он идентифицирует физическое устройство в сети. Этот адрес передаётся в каждом DHCP-пакете, чтобы сервер мог связать IP-адрес с конкретным устройством.**
- **IP-адрес источника:**
    - **В DHCP Discover клиент не имеет IP-адреса, поэтому в поле источника указан 0.0.0.0.**
    - **В DHCP Request, если клиент принимает IP-адрес, предложенный сервером, то этот IP-адрес может быть использован в запросе (если клиент уже получил его) или остаётся 0.0.0.0, если адрес ещё не подтверждён.**
- **IP-адрес назначения:**
    - **В DHCP Discover запрос отправляется на широковещательный адрес (255.255.255.255), чтобы все DHCP-серверы в локальной сети могли получить запрос.**
    - **В DHCP Request запрос отправляется на адрес DHCP-сервера, чтобы подтвердить получение IP-адреса.**

8.1.3 Каков IP-адрес DHCP-сервера?

IP-адрес DHCP-сервера обычно можно увидеть в ответе **DHCPOFFER** или **DHCPACK**. Этот адрес — это IP-адрес, который сервер использует для взаимодействия с клиентами и предоставления им IP-адресов. Для анализа захваченного трафика, вы должны посмотреть поле "Server Identifier" в пакете DHCP Offer или DHCP ACK.

8.1.4 Что произойдёт, если очистить использованный фильтр “bootp”?

Фильтр **"bootp"** (в Wireshark) используется для отображения только DHCP/BOOTP-пакетов в захваченной сетевой трассе. Если вы очистите этот фильтр:

- Вы будете видеть **весь захваченный трафик**, включая пакеты других протоколов, таких как TCP, HTTP, ARP и другие.
- Это может затруднить поиск нужных DHCP-пакетов в большом количестве данных, так как все пакеты, захваченные во время анализа сети, будут отображены.

9. Анализ Discord-трафика

Для анализа трафика, генерируемого приложениями для обмена сообщениями и аудио/видео-общения (например, Discord), используются несколько протоколов. В зависимости от действия (текстовые сообщения, аудио или видеозвонки), различаются используемые протоколы

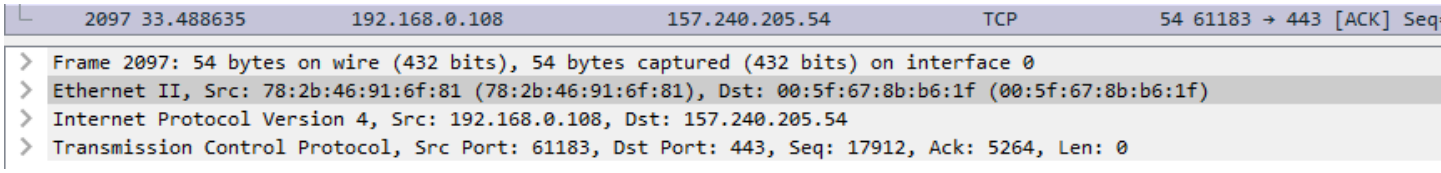

Рис. 31 Трафик тестовых сообщений

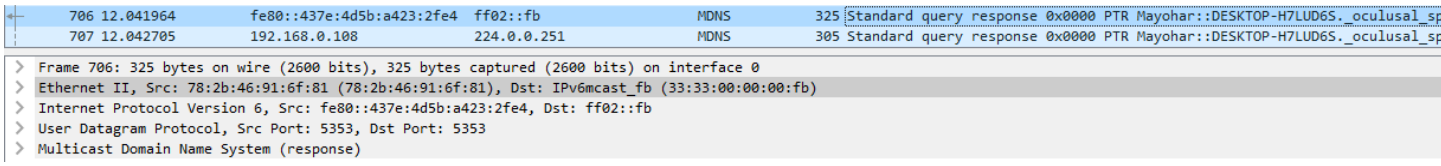

Рис. 32 Трафик Аудио

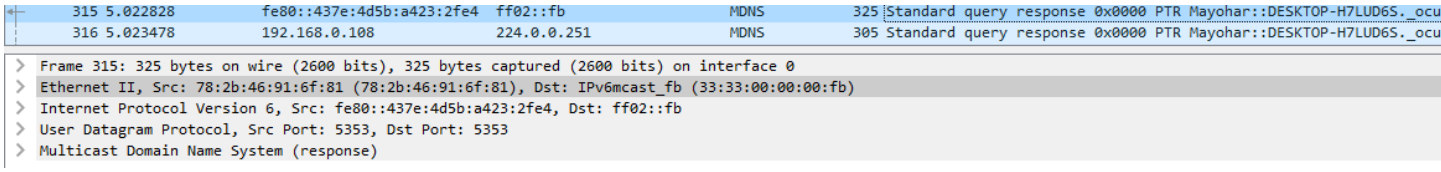

Рис. 33 Трафик видео

9.1 Ответы на вопросы

9.1.1 Чем различаются пакеты разных видов Discord-трафика (текст, аудио, видео)?

- Текстовые сообщения:
    - Discord отправляет текстовые сообщения через защищённые каналы HTTPS. Эти пакеты проходят через порт 443 и используют TCP для передачи данных.
    - Пакеты содержат зашифрованные данные, и их основное назначение — обмен текстовыми сообщениями и другой информацией через постоянное соединение с серверами Discord.
    - Обмен текстом происходит через WebSocket поверх HTTPS, что позволяет поддерживать постоянное соединение для передачи данных в реальном времени.
- Аудио-трафик:
    - Аудио в Discord передаётся через UDP, так как этот протокол лучше подходит для данных, передаваемых в реальном времени (меньше задержек, хотя возможна потеря пакетов).

    - Discord использует Opus в качестве кодека для сжатия и передачи аудио, а данные передаются с минимальной задержкой для обеспечения чёткого звука.
    - Аудио-пакеты имеют меньший размер по сравнению с видео и представляют собой поток данных, закодированных кодеком Opus.
- Видео-трафик:
    - Видео передаётся также через UDP, используя протокол RTP. Видео в Discord может быть закодировано с использованием кодеков, таких как H.264.
    - Видеопакеты крупнее, чем аудиопакеты, так как содержат данные не только о звуке, но и о видеопотоке. Также в них могут содержаться временные метки и другая служебная информация для синхронизации аудио и видео.
    - Видеотрафик потребляет больше полосы пропускания, чем аудио, что можно заметить по большему количеству передаваемых пакетов в единицу времени.

9.1.2 Какой Wireshark-фильтр следует использовать для независимой идентификации Discord-трафика разных видов (текст, аудио, видео)?

Текстовые сообщения:

- Используем фильтр для HTTPS-трафика: tcp.port == 443

Это отобразит трафик, проходящий через зашифрованное соединение (TCP/443), используемое Discord для текстовых сообщений.

Аудио-трафик:

- Аудио-трафик передаётся через **UDP** и кодируется с помощью **Opus**. Используйте следующие фильтры для захвата аудиопотоков: udp && frame contains "Opus"

Видео-трафик:

- Видео в Discord передаётся через **UDP** с использованием **RTP** и кодека **H.264**. Для фильтрации видеопакетов используйте фильтр: udp && rtp && frame contains "H264"

Заключение

В ходе лабораторной работы была изучена структура протокольных блоков данных, а также проведён детальный анализ реального сетевого трафика, генерируемого на компьютере студента. Для этого использовалась утилита **Wireshark**, которая свободно распространяется и широко используется для мониторинга и анализа сетевых пакетов.

Основной целью было освоение работы с Wireshark, начиная с установки и настройки программы, заканчивая захватом и анализом сетевых пакетов. В процессе работы были собраны дампы реального сетевого трафика, и особое внимание было уделено исследованию специфических пакетов, относящихся к выбранным протоколам.

Каждый пакет был детально разобран, начиная от его заголовков и заканчивая содержимым полезной нагрузки, что позволило понять, как различные протоколы обрабатывают и передают данные в сети. В частности, был проведён анализ ключевых протоколов, таких как TCP, UDP, DNS, HTTP, и другие, что позволило изучить их поведение в реальных условиях работы сети.