

Университет ИТМО
Факультет программной инженерии и компьютерной техники

Лабораторная работа №4
«Анализ трафика компьютерных сетей с помощью утилиты
Wireshark»
по дисциплине “Компьютерные сети”

Выполнил:
Студент группы Р3334
Галиуллин Р.Д.

Преподаватель:
Алиев Т. И.

Санкт-Петербург
2025 г.

ОГЛАВЛЕНИЕ

ЗАДАНИЕ	3
Описание лабораторной работы	3
Вариант для выполнения.....	3
4.1. АНАЛИЗ ТРАФИКА УТИЛИТЫ PING.....	4
Ответы на контрольные вопросы	5
4.2. АНАЛИЗ ТРАФИКА УТИЛИТЫ TRACERT (TRACEROUTE).....	8
Ответы на контрольные вопросы	8
4.3. АНАЛИЗ HTTP-ТРАФИКА	12
4.8. АНАЛИЗ DNS-ТРАФИКА.....	14
Ответы на контрольные вопросы	16
ВЫВОД.....	18

ЗАДАНИЕ

Описание лабораторной работы

Цель работы – изучить структуру протокольных блоков данных, анализируя реальный трафик на компьютере студента с помощью бесплатно распространяемой утилиты Wireshark.

В процессе выполнения домашнего задания выполняются наблюдения за передаваемым трафиком с компьютера пользователя в Интернет и в обратном направлении. Применение специализированной утилиты Wireshark позволяет наблюдать структуру передаваемых кадров, пакетов и сегментов данных различных сетевых протоколов. При выполнении УИР рекомендуется выполнить анализ последовательности команд и определить назначение служебных данных, используемых для организации обмена данными в протоколах: ARP, DNS, FTP, HTTP, DHCP.

Вариант для выполнения

Сайт для анализа трафика - rgd.ca

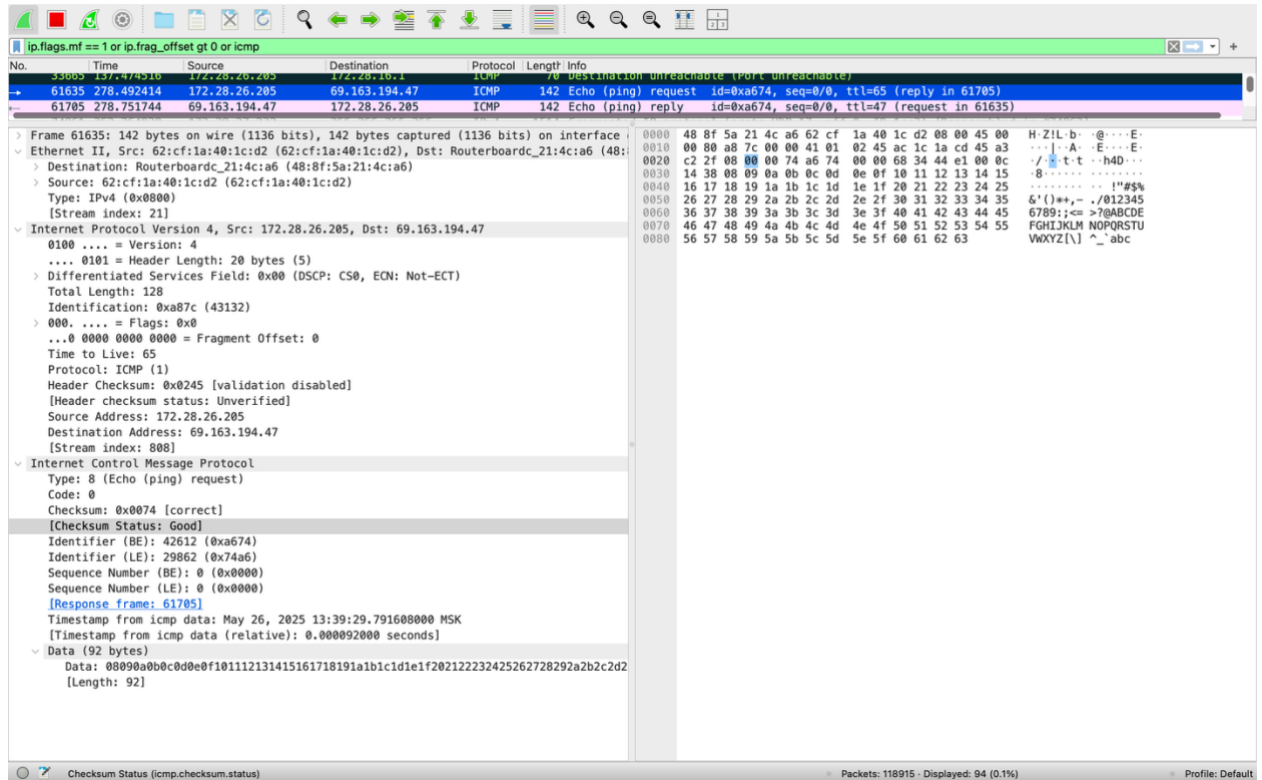
4.1. АНАЛИЗ ТРАФИКА УТИЛИТЫ PING

Запускаем из терминала (MacOS) команду для отправки пакетов на сайт rgd.ca:

ping -s “размер_пакета” -c “количество пакетов” rgd.ca

Чтобы отправлять один пакет, а не 4, как происходит по умолчанию, будем использовать флаг -n. Отправим пакеты размерами 100, 200, 500, 1000, 2000, 5000, 10000 байт.

Утилита управляет ICMP запросами и ответами.



Структура пакета:

Ethernet 2 (канальный уровень)

Заголовок содержит:

- Destination – MAC адрес получателя
- Source – MAC-адрес отправителя
- Type – тип протокола

IP-заголовок (сетевой уровень)

Заголовок содержит:

- Version
- Header Length
- Identification – идентификатор фрагмента
- Protocol – тип вложенного протокола
- Flags – указывается DF и MF

- TTL – ограничение на кол-во хопов
- Fragment offset – смещение фрагмента (если пакет был фрагментирован)
- Header Checksum – контрольная сумма заголовка
- Source IP address
- Destination IP address

ICMP (сетевой протокол)

Заголовок содержит:

- Type – request или reply
- Checksum - CRC
- Identifier – уникальный ID запроса
- Seq number – номер последовательности запроса

Data (поле данных)

Ответы на контрольные вопросы

1. Имеет ли место фрагментация исходного пакета, какое поле на это указывает?
2. Какая информация указывает, является ли фрагмент пакета последним или промежуточным?

Фрагментация пакетов — это процесс разделения большого сетевого пакета на более мелкие части (фрагменты), чтобы они могли быть переданы через сеть с ограничениями на максимальный размер передаваемого блока данных (MTU, Maximum Transmission Unit). Этот механизм реализуется на сетевом уровне (IP) и обеспечивает совместимость между устройствами и сетями с разными MTU.

Да, имеет, на это указывают:

- флаг фрагментации More Fragments в IP-заголовке:

MF = 1: промежуточный фрагмент.

MF = 0: последний фрагмент.

- Fragment Offset (смещение фрагмента) – на позицию фрагмента в исходном пакете (в 8-байтовых блоках).

```

Internet Protocol Version 4, Src: 172.28.26.205, Dst: 69.163.194.47
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 548
  Identification: 0x5412 (21522)
  > 000. .... = Flags: 0x0
  ...0 0000 1011 1001 = Fragment Offset: 1480
  
```

✓ 001. = Flags: 0x1, More fragments
 0... = Reserved bit: Not set
 .0.. = Don't fragment: Not set
 ..1. = More fragments: Set

```

  ✓ Internet Protocol Version 4, Src: 172.28.26.205, Dst: 69.163.194.47
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 128
      Identification: 0xa49b (42139)
    > 000. .... = Flags: 0x0
      ...0 0000 0000 0000 = Fragment Offset: 0
  
```

✓ 000. = Flags: 0x0
 0... = Reserved bit: Not set
 .0.. = Don't fragment: Not set
 ..0. = More fragments: Not set

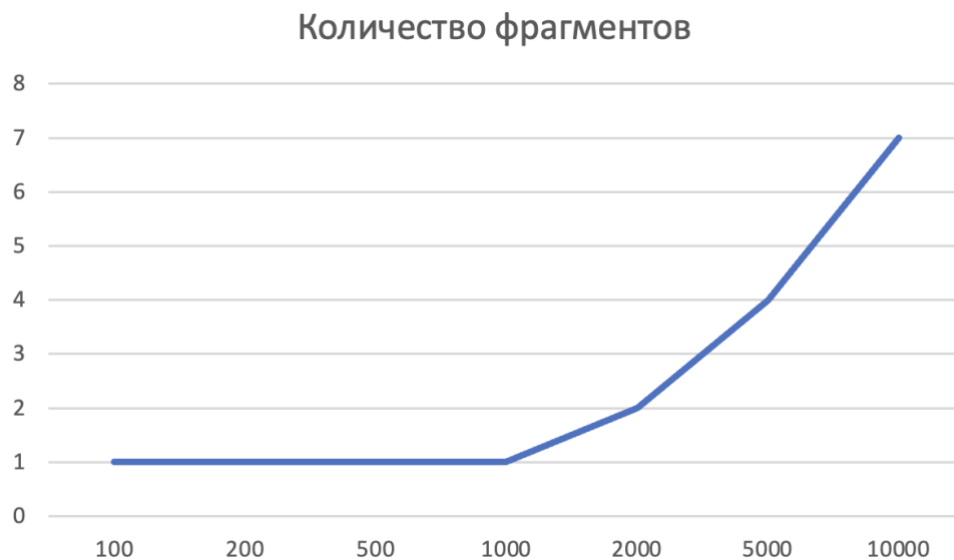
3. Чему равно количество фрагментов при передаче ping-пакетов?

100, 200, 500, 1000, 2000, 5000, 10000

Размер пакета (байт)	Количество фрагментов
100	1
200	1
500	1
1000	1
2000	2
5000	4
10000	7

Необходимо округлить вверх до целого числа деление размера пакета на MTU = 1480

4. Построить график, в котором на оси абсцисс находится размер_пакета, а по оси ординат – количество фрагментов, на которое был разделён каждый ping-пакет.



5. Как изменить поле TTL с помощью утилиты ping?

С помощью флага -m "TTL"

Пример с командой ping -s 5000 -c 1 -m 5 rgd.ca

```
✓ 000. .... = Flags: 0x0
  0... .... = Reserved bit: Not set
  .0.. .... = Don't fragment: Not set
  ..0. .... = More fragments: Not set
  ...0 0010 0010 1011 = Fragment Offset: 4440
Time to Live: 5
Protocol: ICMP (1)
```

6. Что содержится в поле данных ping-пакета?

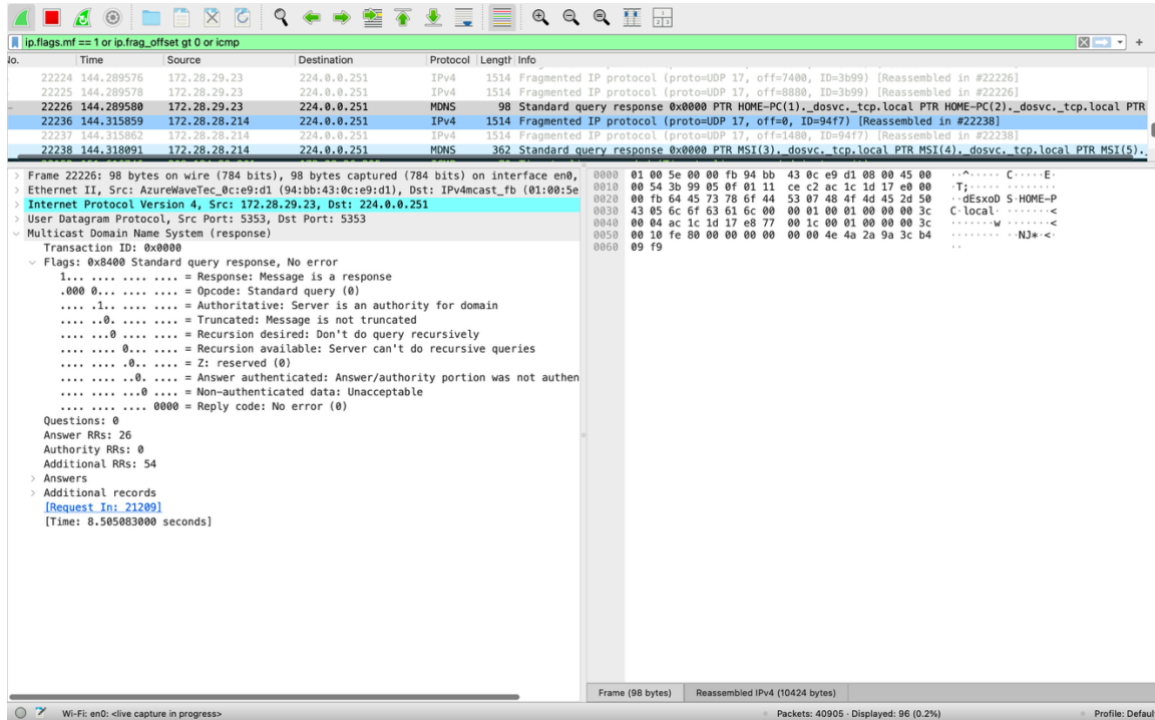
- Заголовок ICMP
- Идентификатор
- Номер последовательности
- Содержимое

4.2. АНАЛИЗ ТРАФИКА УТИЛИТЫ TRACERT (TRACEROUTE)

Запустим команду из терминала:

```
tracert -n rgd.ca
```

Флаг -n указывает на то, что DNS пакеты отправлялись после построения маршрута, так как в них нет важного функционала



Заголовок содержит:

- ID запроса
- Flags – ошибки, авторитетность, тип запроса/ответа.
- Questions – кол-во запросов
- Answer – кол-во ответов
- Authority – кол-во записей авторитетных серверов
- Additional – кол-во дополнительных записей

Ответы на контрольные вопросы

1. Сколько байт содержится в заголовке IP? Сколько байт содержится в поле данных?

```
Internet Protocol Version 4, Src: 172.28.29.23, Dst: 224.0.0.251
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
```

Заголовок IPv4: 20 байт (без опций).

Поле данных:

Для ICMP-пакетов traceroute:

Заголовок ICMP — 8 байт.

Полезная нагрузка (данные) — зависит от реализации. В macOS traceroute по умолчанию отправляет UDP-пакеты с пустой нагрузкой.

Пример: UDP-пакет с заголовком 8 байт и данными 0 байт.

У ICMP заголовок равен 8 байт, а сами данные 64 байта.

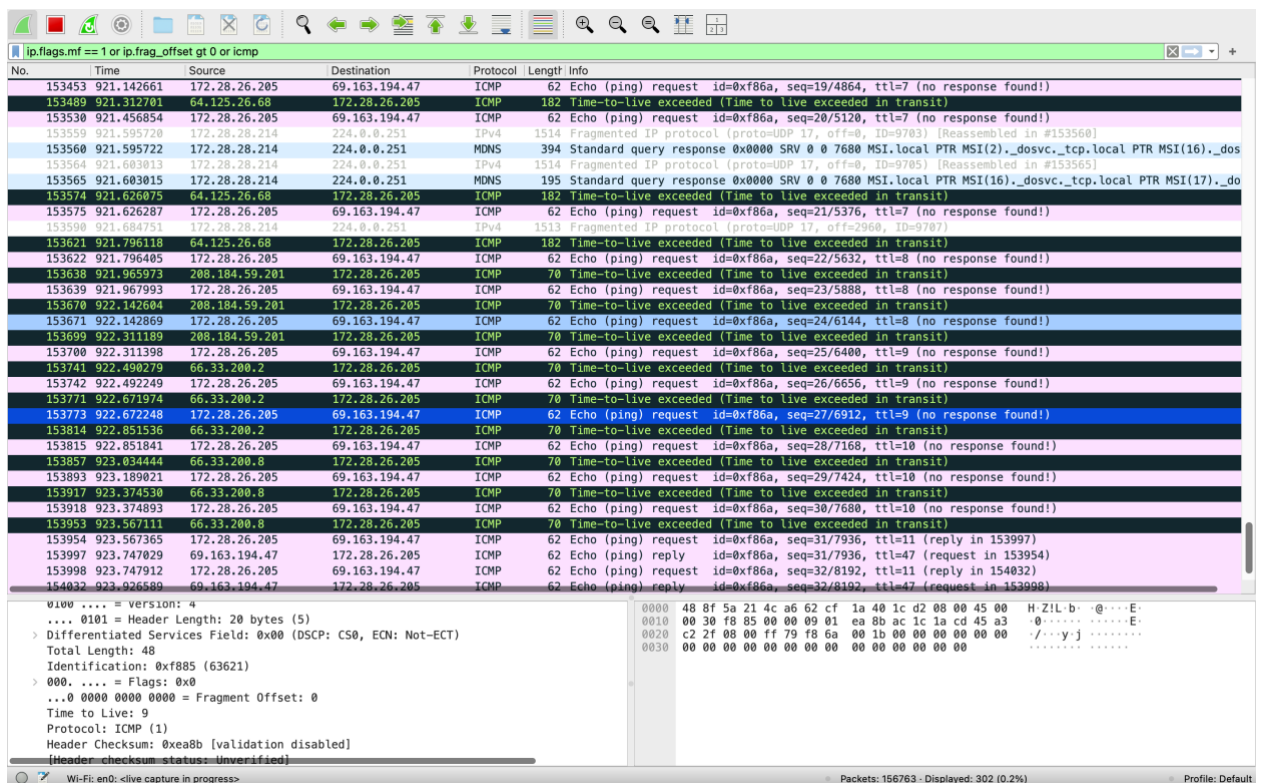
Тогда данные 72 байта. IP-заголовок 20 байт.

2. Как и почему изменяется поле TTL в следующих друг за другом ICMP- пакетах tracer? Для ответа на этот вопрос нужно проследить изменение TTL при передаче по маршруту, состоящему из более чем двух хопов.

Принцип работы:

tracertoute отправляет пакеты с TTL до достижения цели.

Каждый маршрутизатор уменьшает TTL на 1. При TTL=0 пакет отбрасывается, и отправитель получает ICMP-сообщение Time Exceeded.



3. Чем отличаются ICMP-пакеты, генерируемые утилитой `tracert`, от ICMP- пакетов, генерируемых утилитой `ping` (см. предыдущее задание).

ping:

Использует ICMP Echo Request (тип 8) и ICMP Echo Reply (тип 0).

Данные содержат временные метки и случайные байты.

tracert (macOS):

По умолчанию отправляет UDP-пакеты на случайный порт (обычно 33434+).

Получает ICMP Time Exceeded (тип 11) от маршрутизаторов.

Для финального хоста может получить ICMP Port Unreachable (тип 3), если порт закрыт.

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

Echo Reply подтверждает доступность узла.

ICMP Error помогает диагностировать проблемы на пути (например, перегрузку маршрутизаторов). Приходит, когда TTL истекает

ICMP Reply (Echo Reply):

Ответ на успешный запрос (например, ping).

Тип: 0.

ICMP Error (Time Exceeded, Port Unreachable):

Сообщает об ошибках:

Time Exceeded (тип 11): TTL достиг нуля.

Port Unreachable (тип 3): порт недоступен.

Содержит заголовок исходного пакета для диагностики.

5. Что изменится в работе tracert, если убрать ключ «-d»? Какой дополнительный трафик при этом будет генерироваться?

Без -n tracert будет выполнять обратные DNS-запросы для IP-адресов маршрутизаторов.

Дополнительный трафик:

DNS-запросы (порт 53) для разрешения IP-адресов в доменные имена.

В Wireshark будут видны пакеты с фильтром dns.

Wireshark packet capture showing ICMP and DNS traffic. The top pane displays a list of packets with columns for No., Time, Source, Destination, Protocol, Length, and Info. The bottom pane shows the packet details for the selected packet (No. 6595, Time 28.344803), including the ICMP Echo (ping) request and the DNS response.

No.	Time	Source	Destination	Protocol	Length	Info
359	11.412249	172.28.16.1	172.28.26.205	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
361	11.412250	172.28.16.1	172.28.26.205	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
366	11.438431	172.28.16.1	172.28.26.205	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
369	11.435856	77.234.199.66	172.28.26.205	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
373	11.446540	77.234.199.66	172.28.26.205	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
376	11.446548	77.234.199.66	172.28.26.205	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
388	11.451757	87.248.228.182	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2828	11.769589	139.45.238.84	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2824	11.781295	139.45.238.84	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2826	11.789532	139.45.238.84	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2832	11.834340	87.245.232.252	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2839	11.887223	87.245.232.252	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2849	11.926783	87.245.232.252	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2859	11.971184	208.184.12.152	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2869	12.017284	208.184.12.152	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2878	12.059426	208.184.12.152	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
3236	13.862748	172.28.26.205	172.28.16.1	DNS	86	Standard query 0x1737 A waa-pa.clients6.google.com
3237	13.862853	172.28.26.205	172.28.16.1	DNS	86	Standard query 0xe0de HTTPS waa-pa.clients6.google.com
3243	13.813981	172.28.16.1	172.28.26.205	DNS	238	Standard query response 0x1737 A waa-pa.clients6.google.com A 74.125.131.95 NS ns1.google.com NS ns2.
3248	13.818816	172.28.16.1	172.28.26.205	DNS	222	Standard query response 0xe0de HTTPS waa-pa.clients6.google.com NS ns2.google.com NS ns4.google.com N
3536	15.297569	172.28.26.205	172.28.16.1	DNS	91	Standard query 0x7e59 A signaler-pa.clients6.google.com
3537	15.297653	172.28.26.205	172.28.16.1	DNS	91	Standard query 0xa623 HTTPS signaler-pa.clients6.google.com
3548	15.385480	172.28.16.1	172.28.26.205	DNS	243	Standard query response 0x7e59 A signaler-pa.clients6.google.com A 216.58.211.234 NS ns1.google.com N
3543	15.387603	172.28.16.1	172.28.26.205	DNS	227	Standard query response 0xa623 HTTPS signaler-pa.clients6.google.com NS ns2.google.com NS ns4.google.
6263	27.242432	208.184.59.281	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
6303	27.417455	208.184.59.281	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
6356	27.586515	208.184.59.281	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
6398	27.783354	66.33.208.3	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
6483	27.786288	172.28.26.205	172.28.16.1	DNS	84	Standard query 0x3f68 PTR 3.208.33.66.in-addr.arpa
6437	27.981280	172.28.16.1	172.28.26.205	DNS	226	Standard query response 0x3f68 PTR 3.208.33.66.in-addr.arpa PTR pdx1-cr-2.sd.dreamhost.com NS ns3.dre
6483	28.163882	66.33.208.3	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
6505	28.344803	66.33.208.2	172.28.26.205	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)

Packet 6595 details:

..... = Answer authenticated: Answer/authority portion was not authen
..... = Non-authenticated data: Unacceptable
..... = Reply code: No error (0)

Questions: 1
Answer RRs: 0
Authority RRs: 4
Additional RRs: 4

0000 62 cf 1a 40 1c d2 48 8f 5a 21 4c a6 08 00 45 00 b 0 N ZIL...E-
0010 00 05 be 83 00 00 40 11 30 8e ac 1c 10 01 ac 1c ... 0 B ...
0020 1a cd 00 35 0b 78 00 c1 e5 a7 a6 23 81 80 00 01 ...5 X ...
0030 00 00 00 04 00 04 0b 73 69 67 6e 61 6c 65 72 d ... s ignaler-
0040 78 61 80 63 6c 69 65 6e 74 73 36 06 67 6f 67 ... pa clien ts6 goog
0050 6c 65 03 63 6f 6d 00 00 41 00 01 c0 21 00 02 00 ...e com A ...
0060 01 00 01 66 03 00 00 83 6e 73 32 c0 21 c0 21 00 ...f... ns2 i-
0070 02 00 01 00 01 66 03 00 00 83 6e 73 34 c0 21 c0 ...f... ns4 i-
0080 21 00 02 00 01 00 01 66 03 00 00 83 6e 73 33 c0 ...f... ns3 i-

Отправка без ключа -п

4.3. АНАЛИЗ HTTP-ТРАФИКА

Установим фильтр http для захвата http пакетов. К сожалению, с нашим сайтом не получается получить ответ 304 от сервера (Not Modified), поэтому воспользуемся распространенным сайтом example.com для отправки GET-запроса

The image shows a Wireshark packet capture of an HTTP GET request and response. The packet list on the left shows packets 189 through 293. Packet 241 is selected, showing the details of the HTTP response. The details pane shows the following information:

- Checksum: 0x4276 [unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
- [Timestamps]
- [SEQ/ACK analysis]
- TCP payload (978 bytes)
- Hypertext Transfer Protocol
 - HTTP/1.1 200 OK\r\n
 - Accept-Ranges: bytes\r\n
 - Content-Type: text/html\r\n
 - ETag: "84238dfc8092e5d9c0dac8ef93371a07:1736799080.121134"\r\n
 - Last-Modified: Mon, 13 Jan 2025 20:11:20 GMT\r\n
 - Vary: Accept-Encoding\r\n
 - Content-Encoding: gzip\r\n
 - Content-Length: 648\r\n
 - Cache-Control: max-age=2872\r\n
 - Date: Mon, 26 May 2025 13:10:13 GMT\r\n
 - Connection: keep-alive\r\n
 - \r\n
 - [Request in frame: 189]
 - [Time since request: 0.29016000 seconds]
 - [Request URI: /]
 - [Full request URI: http://example.com/]
 - Content-encoded entity body (gzip): 648 bytes -> 1256 bytes
 - File Data: 1256 bytes
- Line-based text data: text/html (46 lines)

The packet bytes pane on the right shows the raw data of the packet, including the HTTP response status line and headers.

Первичный GET-запрос

The image shows a Wireshark packet capture of an HTTP GET request and response. The packet list on the left shows packets 189 through 293. Packet 2369 is selected, showing the details of the HTTP response. The details pane shows the following information:

- Acknowledgment Number: 599 (relative ack number)
- Acknowledgment number (raw): 2495123463
- 1000 = Header Length: 32 bytes (8)
- Flags: 0x018 (PSH, ACK)
- Window: 505
- [Calculated window size: 64640]
- [Window size scaling factor: 128]
- Checksum: 0x35c6 [unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
- [Timestamps]
- [SEQ/ACK analysis]
- TCP payload (250 bytes)
- Hypertext Transfer Protocol
 - HTTP/1.1 304 Not Modified\r\n
 - Content-Type: text/html\r\n
 - Last-Modified: Mon, 13 Jan 2025 20:11:20 GMT\r\n
 - ETag: "84238dfc8092e5d9c0dac8ef93371a07:1736799080.121134"\r\n
 - Cache-Control: max-age=2851\r\n
 - Date: Mon, 26 May 2025 13:10:34 GMT\r\n
 - Connection: keep-alive\r\n
 - \r\n
 - [Request in frame: 2338]
 - [Time since request: 0.21690700 seconds]
 - [Request URI: /]
 - [Full request URI: http://example.com/]

The packet bytes pane on the right shows the raw data of the packet, including the HTTP response status line and headers.

Условный GET-запрос

```
Accept-Encoding: gzip, deflate\r\n
Accept-Language: ru-RU, ru;q=0.9, en-US;q=0.8, en;q=0.7\r\n
If-None-Match: "84238dfc8092e5d9c0dac8ef93371a07:1736799080.121134"\r\n
If-Modified-Since: Mon, 13 Jan 2025 20:11:20 GMT\r\n
\r\n
```

Заголовки:

If-Modified-Since — дата последнего изменения ресурса (из предыдущего ответа).

If-None-Match — уникальный идентификатор версии ресурса (ETag).

✓ Hypertext Transfer Protocol

> HTTP/1.1 304 Not Modified\r\n

Content-Type: text/html\r\n

Last-Modified: Mon, 13 Jan 2025 20:11:20 GMT\r\n

ETag: "84238dfc8092e5d9c0dac8ef93371a07:1736799080.121134"\r\n

Параметр	Первичный GET	Условный GET
Код ответа	200 OK	304 Not Modified / 200 OK
Тело ответа	Полный HTML	Отсутствует (при 304)
Заголовки запроса	Нет If-Modified-Since	Есть If-Modified-Since, ETag
Передача данных	Полная	Минимальная (кеширование)

4.8. АНАЛИЗ ДНСР-ТРАФИКА

Отследим и проанализируем трафик протокола ДНСР.

Проверим, что был использован ДНСР. На скриншоте мы видим yiaddr и другие ДНСР-поля — значит IP получен по ДНСР

```
sudo ipconfig set en0 DHCP

admin@MacBook-Pro-RushB ~ % sudo ipconfig set en0 BOOTP
sleep 2
sudo ipconfig set en0 DHCP

admin@MacBook-Pro-RushB ~ % ipconfig getpacket en0

op = BOOTREPLY
htype = 1
flags = 0x0
hlen = 6
hops = 0
xid = 0x581a5b53
secs = 0
ciaddr = 0.0.0.0
yiaddr = 172.28.26.205
siaddr = 172.28.16.1
giaddr = 0.0.0.0
chaddr = 62:cf:1a:40:1c:d2
sname =
file =
options:
Options count is 7
dhcp_message_type (uint8): ACK 0x5
server_identifier (ip): 172.28.16.1
lease_time (uint32): 0x15180
subnet_mask (ip): 255.255.240.0
router (ip_mult): {172.28.16.1}
domain_name_server (ip_mult): {172.28.16.1, 77.234.194.2, 77.234.216.3}
end (none):
admin@MacBook-Pro-RushB ~ % █
```

Настроим фильтр bootp - отображающий фильтр, который используется для анализа трафика протокола ДНСР.

BOOTP (Bootstrap Protocol) — это более старый протокол автоматической настройки IP-адресов, появившийся в 1980-х.

ДНСР (Dynamic Host Configuration Protocol) был создан как расширение BOOTP, чтобы добавить:

- аренду IP-адресов,
- повторное использование адресов,
- автоматическую настройку DNS и шлюзов,
- и другие функции.

ДНСР использует тот же формат сообщений, что и BOOTP, и те же порты:

- UDP порт 67 (сервер)
- UDP порт 68 (клиент)

Далее несколько раз сбросим и запросим новый IP-адрес:


```

xid = 0x581a5b53
secs = 0
ciaddr = 0.0.0.0
yiaddr = 172.28.26.205
siaddr = 172.28.16.1
giaddr = 0.0.0.0
chaddr = 62:cf:1a:40:1c:d2
sname =
file =
options:
Options count is 7
dhcp_message_type (uint8): ACK 0x5
server_identifier (ip): 172.28.16.1
lease_time (uint32): 0x15180
subnet_mask (ip): 255.255.240.0
router (ip_mult): {172.28.16.1}
domain_name_server (ip_mult): {172.28.16.1, 77.234.194.2, 77.234.216.3}
end (none):
admin@MacBook-Pro-RushB ~ % sudo ipconfig set en0 BOOTP
sleep 2
sudo ipconfig set en0 DHCP

Password:
admin@MacBook-Pro-RushB ~ % sudo ipconfig set en0 BOOTP
sleep 2
sudo ipconfig set en0 DHCP

admin@MacBook-Pro-RushB ~ % sudo ipconfig set en0 BOOTP
sleep 2
sudo ipconfig set en0 DHCP

admin@MacBook-Pro-RushB ~ %

```

Получим в WireShark:

The screenshot shows a Wireshark capture of network traffic on the 'bootp' interface. The packet list displays several DHCP-related packets:

- 957: DHCP Discover (Transaction ID 0xefa4328b) from 0.0.0.0 to 255.255.255.255.
- 1008: DHCP Offer (Transaction ID 0xefa4328b) from 172.28.16.1 to 0.0.0.0.
- 1018: DHCP Request (Transaction ID 0x18b5c1) from 0.0.0.0 to 255.255.255.255.
- 1184: DHCP Request (Transaction ID 0xefa4328b) from 0.0.0.0 to 255.255.255.255.
- 1185: DHCP ACK (Transaction ID 0xefa4328b) from 172.28.16.1 to 0.0.0.0.

The packet details for the selected DHCP ACK (Frame 957) show the following information:

- Section number: 1
- Interface id: 0 (en0)
- Encapsulation type: Ethernet (1)
- Arrival Time: May 26, 2025 16:36:20.657760000 MSK
- UTC Arrival Time: May 26, 2025 13:36:20.657760000 UTC
- Epoch Arrival Time: 1748266580.657760000
- [Time shift for this packet: 0.000000000 seconds]
- [Time delta from previous captured frame: 0.003341000 seconds]
- [Time delta from previous displayed frame: 0.003341000 seconds]
- [Time since reference or first frame: 5.436346000 seconds]
- Frame Number: 957
- Frame Length: 342 bytes (2736 bits)
- Capture Length: 342 bytes (2736 bits)
- [Frame is marked: False]
- [Frame is ignored: False]
- [Protocols in frame: eth:ethertype:ip:udp:dhcp]
- [Coloring Rule Name: UDP]
- [Coloring Rule String: udp]

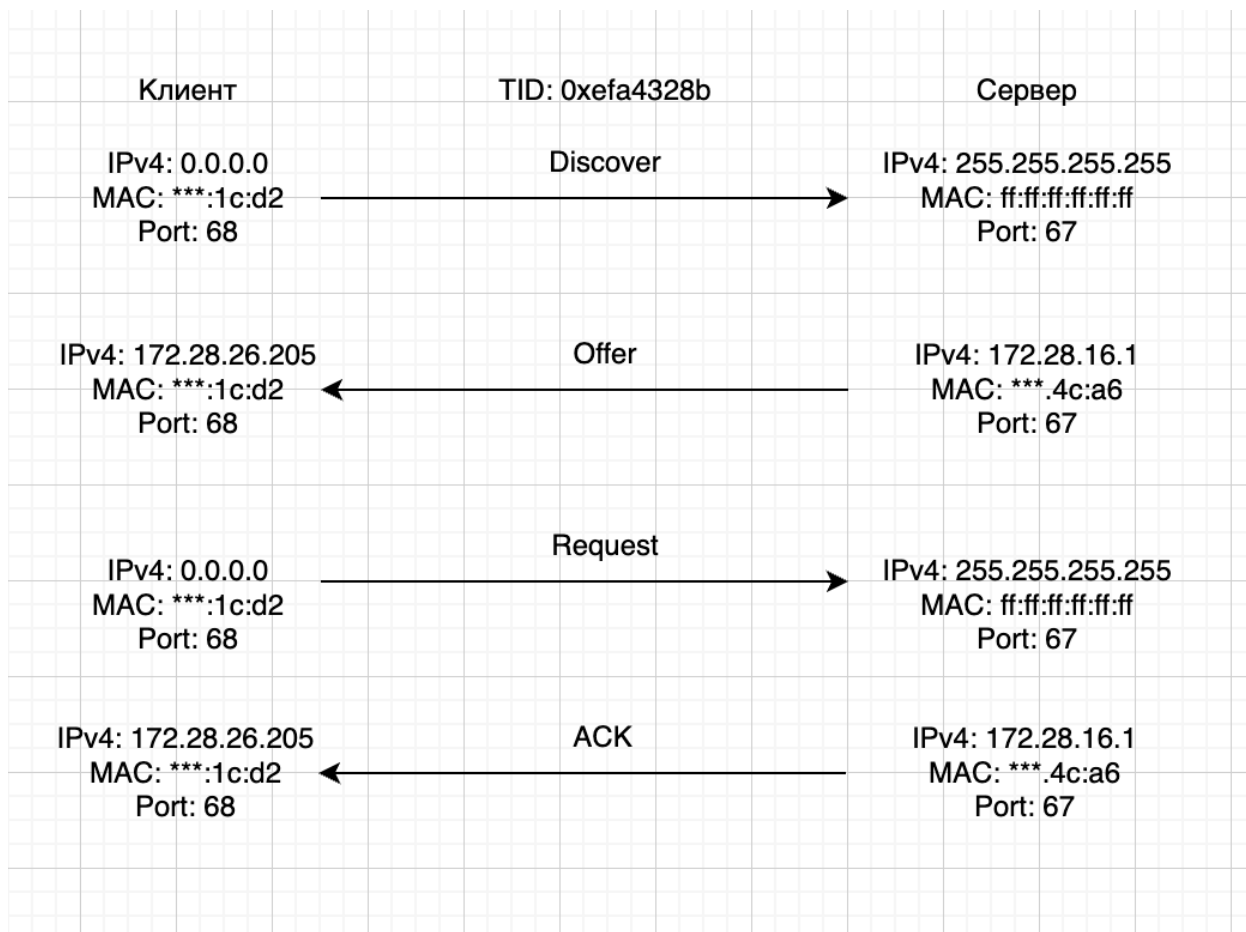
The packet bytes section shows the raw data in hexadecimal and ASCII:

```

0000 ff ff ff ff ff ff ff 62 cf 1a 40 1c d2 08 00 45 00 .....b...@....E:
0010 01 48 a2 3e 00 00 ff 11 18 67 00 00 00 00 ff ff ..H>....g.....
0020 ff ff 00 44 00 43 01 34 08 ef 01 01 06 00 ef a4 ...D.C.4.....
0030 32 8b 00 00 00 00 00 00 00 00 00 00 00 00 00 2.....
0040 00 00 00 00 00 62 cf 1a 40 1c d2 00 00 00 00 .....b...@.....
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0110 00 00 00 00 00 00 63 82 53 63 35 01 01 37 0d 01 .....c.ScS...7...
0120 79 03 06 0f 72 77 fc 64 65 5f 2c 2e 39 02 05 dc y...nd e...9...
0130 3d 07 01 62 cf 1a 40 1c d2 33 04 00 76 a7 00 0c =.b@...3.v...
0140 0a 4d 61 63 42 6f 6f 6b 50 72 6f ff 00 00 00 00 -MacBook Pro:....
0150 00 00 00 00 00 00 00 00 .....

```

Временная диаграмма:



Ответы на контрольные вопросы

1. Чем различаются пакеты «DHCP Discover» и «DHCP Request»?

Параметр	DHCP Discover	DHCP Request
Назначение	Найти DHCP-сервер	Запросить IP-адрес от сервера
Тип	Исходная широковещательная заявка	Ответ на предложение или обновление аренды
Ответы	Получает предложения от серверов	Подтверждает выбор конкретного сервера

2. Как и почему менялись MAC- и IP-адреса источника и назначения в переданных DHCP-пакетах.

В DHCP Discover и Request:

MAC-адрес клиента используется, так как у клиента ещё нет IP.

IP-адрес источника: 0.0.0.0

IP-адрес назначения: 255.255.255.255 (широковещательный)

В ответах сервера (Offer, ACK):

MAC-адрес назначения — MAC клиента

IP-адрес источника — IP сервера

IP назначения — IP клиента

3. Каков IP-адрес DHCP-сервера?

172.28.16.1

4. Что произойдёт, если очистить использованный фильтр «bootp»?

Wireshark отобразит весь трафик на интерфейсе, включая ARP, DNS, HTTP и т.д.

Это затруднит анализ DHCP-пакетов.

ВЫВОД

В ходе данной лабораторной работы было изучено строение протокольных блоков данных посредством анализа сетевого трафика с использованием утилиты Wireshark.

В частности, были проанализированы следующие протоколы:

- ICMP (ping): Определены структура и назначение заголовков Ethernet, IP и ICMP, а также изучен процесс фрагментации IP-пакетов.
- ICMP (traceroute): Установлены отличия ICMP-пакетов, генерируемых утилитами ping и traceroute, а также особенности изменения поля TTL. о
- DNS: Проанализированы DNS-запросы и ответы, а также определены основные поля DNS-заголовков.
- DHCP: Изучены этапы получения IP-адреса с использованием протокола DHCP, а также выявлены различия между пакетами DHCP Discover и DHCP Request.

Полученные результаты позволили углубить понимание принципов передачи данных в компьютерных сетях и роли каждого уровня OSI-модели в этом процессе.