

Университет ИТМО

Факультет программной инженерии и компьютерной техники

**Сети ЭВМ и телекоммуникации**

Учебно-исследовательская работа №3

Анализ трафика компьютерных сетей с помощью утилиты  
Wireshark

Лабушев Тимофей

Группа Р3302

Санкт-Петербург

2019

## Цель

Изучить структуру протокольных блоков данных, анализируя реальный трафик на компьютере студента с помощью бесплатно распространяемой утилиты Wireshark.

## Задание

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 требуется анализировать последовательности команд и назначение служебных данных, используемых для организации обмена данными в следующих протоколах: ARP, DNS, FTP, HTTP, DHCP.

## Исходные данные

В качестве адреса сайта в заданиях используются следующие URL:

- HTTP: <http://ltm.lt/>
- FTP: <ftp://ftp.deb-multimedia.org/>

## Ход работы

### Анализ трафика утилиты ping

Трафик, создаваемый утилитой ping, отслеживается с помощью Wireshark со следующими настройками:

- Захватывается интерфейс wlp2s0 (Wi-Fi)
- Фильтр host 185.5.53.9 (IP адрес выбранного HTTP сайта)

Для варьирования размера пакетов утилита ping запускалась следующим образом:

```
for s in 100 500 1000 1500 2000 3000 4000 5000 7000 10000; do ping -c 1 -s $s ltm.lt; done
```

Запрос утилиты выглядит следующим образом:

```
↳ Frame 1: 142 bytes on wire (1136 bits), 142 bytes captured (1136 bits) on interface wlp2s0, id 0
↳ Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
↳ Internet Protocol Version 4, Src: 192.168.0.101, Dst: 185.5.53.9
↳ Internet Control Message Protocol
```

Рис. 1. ping-запрос

## Frame

```
▼ Frame 1: 142 bytes on wire (1136 bits), 142 bytes captured (1136 bits) on interface wlp2s0, id 0
  ▼ Interface id: 0 (wlp2s0)
    └─ Interface name: wlp2s0
  └─ Encapsulation type: Ethernet (1)
  └─ Arrival Time: Apr 13, 2020 12:46:14.961908301 MSK
  └─ [Time shift for this packet: 0.000000000 seconds]
  └─ Epoch Time: 1586771174.961908301 seconds
  └─ [Time delta from previous captured frame: 0.000000000 seconds]
  └─ [Time delta from previous displayed frame: 0.000000000 seconds]
  └─ [Time since reference or first frame: 0.000000000 seconds]
  └─ Frame Number: 1
  └─ Frame Length: 142 bytes (1136 bits)
  └─ Capture Length: 142 bytes (1136 bits)
  └─ [Frame is marked: False]
  └─ [Frame is ignored: False]
  └─ [Protocols in frame: eth:ethertype:ip:icmp:data]
  └─ [Coloring Rule Name: ICMP]
  └─ [Coloring Rule String: icmp || icmpv6]
```

*Рис. 2. Frame*

В контексте Wireshark термин *Frame* применяется к метаданным, собранным при наблюдении пакета.

## Ethernet II

```
▼ Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
  ▼ Destination: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
    └─ Address: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
      └─ .... ..0. .... = LG bit: Globally unique address (factory default)
      └─ .... ..0. .... = IG bit: Individual address (unicast)
  ▼ Source: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
    └─ Address: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
      └─ .... ..0. .... = LG bit: Globally unique address (factory default)
      └─ .... ..0. .... = IG bit: Individual address (unicast)
  └─ Type: IPv4 (0x0800)
```

*Рис. 3. Ethernet II*

Заголовок Ethernet II содержит в себе управляющие данные канального уровня:

- тип протокола (EtherType), в данном случае IPv4
- MAC-адреса отправителя и получателя

## IPv4

```
Internet Protocol Version 4, Src: 192.168.0.101, Dst: 185.5.53.9
- 0100 .... = Version: 4
- .... 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  - 0000 00.. = Differentiated Services Codepoint: Default (0)
  - .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
- Total Length: 128
- Identification: 0x9b24 (39716)
- Flags: 0x4000, Don't fragment
  - 0... .... = Reserved bit: Not set
  - .1.. .... = Don't fragment: Set
  - ..0. .... = More fragments: Not set
- ...0 0000 0000 0000 = Fragment offset: 0
- Time to live: 64
- Protocol: ICMP (1)
- Header checksum: 0xf03c [validation disabled]
- [Header checksum status: Unverified]
- Source: 192.168.0.101
- Destination: 185.5.53.9
```

Рис. 4. IPv4

Заголовок IPv4 содержит в себе управляющие данные сетевого уровня:

- версию протокола (4)
- длину заголовка в 32-битных словах (5), изменяющуюся в случае указания опций пакета
- класс обслуживания, который *может* использоваться маршрутизаторами для определения порядка отбрасывания пакетов в случае сетевых проблем
- идентификационный номер, используемый при фрагментации пакета
- флаги
- время жизни — число маршрутизаторов, через которые пакет может пройти до того, как будет отброшен; предотвращает бесконечное циркулирование пакетов в сети в случае ошибок
- используемый протокол
- контрольную сумму заголовка
- IP-адреса отправителя и получателя

Отдельно рассмотрим флаги:

- первый бит зарезервирован, всегда установлен в ноль
- второй бит, don't fragment (DF), указывает на то, что пакет должен быть отброшен, если для его передачи требуется фрагментация, и используется для определения максимального размера пакета, который не будет фрагментирован (MTU)
- третий бит, more fragments (MF), устанавливается всеми фрагментированными пакетами кроме последнего

## ICMP

ICMP — протокол сетевого уровня, который используется для передачи диагностической информации и сообщениях об ошибках, возникших при передаче данных.

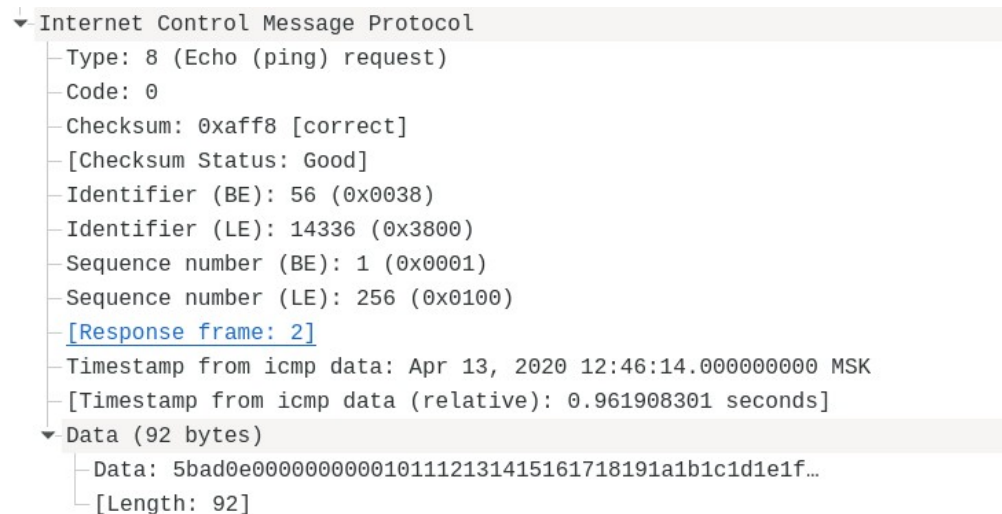


Рис. 5. ICMP

Заголовок IPv4 содержит в себе следующие управляющие данные:

- тип сообщения, в данном случае ping-запрос
- подтип сообщения, в случае ping-запроса всегда 0
- контрольная сумма заголовка и данных

### Вопросы к заданию

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

Фрагментация имеет место при превышении максимального размера пакета (MTU — Maximum Transmission Unit), на что указывает один из управляющих флагов протокола IPv4. MTU для Ethernet составляет 1500 байт.

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

Флаг MF устанавливается в 1, если пакет является промежуточным, и в 0, если он последний или единственный.

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

Определим максимальный размер данных, передаваемых в одном пакете: из MTU вычтем размер IPv4 заголовка (20 байт) и ICMP заголовка (8 байт), получим 1472 байта.

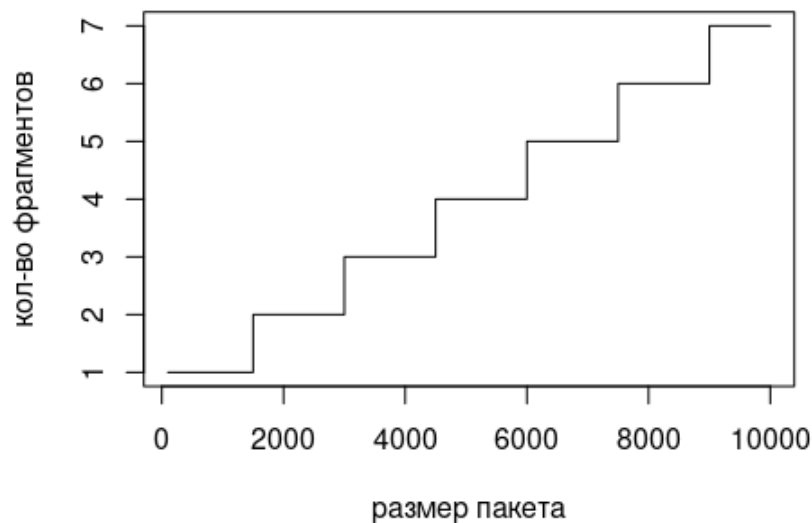
Количество пакетов можно аппроксимировать, как  $S/1472$ , где  $S$  — размер данных в ICMP пакете.

Количество фрагментов  $F$  в зависимости от размера отправленных пакетов  $S$ :

Таблица 1.

S	100	500	1000	1500	2000	3000	5000	7000	10000
F	1	1	1	2	2	3	4	5	7

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



5. Как изменить поле TTL с помощью утилиты ping?

Передать необходимое значение с ключом -t.

6. Что содержится в поле данных ping-пакета?

70	4f	57	ce	48	24	f8	63	3f	f4	1e	ab	08	00	45	00	p0w	H\$	c	?	.....E
02	10	9b	49	40	00	40	01	ee	87	c0	a8	00	65	b9	05	...	I@	..@	.....e	..
35	09	08	00	ec	a1	00	39	00	01	e7	34	94	5e	00	00	5	.....9	...	4	^
00	00	23	33	00	00	00	00	00	00	10	11	12	13	14	15	...	#3	.....	.....	.....
16	17	18	19	1a	1b	1c	1d	1e	1f	20	21	22	23	24	25	.....	..	!"#\$%	.....	.....
26	27	28	29	2a	2b	2c	2d	2e	2f	30	31	32	33	34	35	&'()	*+, -	./012345	.....	.....
36	37	38	39	3a	3b	3c	3d	3e	3f	40	41	42	43	44	45	6789;	<=	>?@ABCDE	.....	.....
46	47	48	49	4a	4b	4c	4d	4e	4f	50	51	52	53	54	55	FGHIJKLM	NOPQRSTU	.....	.....	.....
56	57	58	59	5a	5b	5c	5d	5e	5f	60	61	62	63	64	65	VWXYZ[\]	^_`abcde	.....	.....	.....
66	67	68	69	6a	6b	6c	6d	6e	6f	70	71	72	73	74	75	fghijklm	nopqrstu	.....	.....	.....
76	77	78	79	7a	7b	7c	7d	7e	7f	80	81	82	83	84	85	vwxyz{ }	~.....	.....	.....	.....
86	87	88	89	8a	8b	8c	8d	8e	8f	90	91	92	93	94	95	.....	.....	.....	.....	.....
96	97	98	99	9a	9b	9c	9d	9e	9f	a0	a1	a2	a3	a4	a5	.....	.....	.....	.....	.....
a6	a7	a8	a9	aa	ab	ac	ad	ae	af	b0	b1	b2	b3	b4	b5	.....	.....	.....	.....	.....
b6	b7	b8	b9	ba	bb	bc	bd	be	bf	c0	c1	c2	c3	c4	c5	.....	.....	.....	.....	.....
c6	c7	c8	c9	ca	cb	cc	cd	ce	cf	d0	d1	d2	d3	d4	d5	.....	.....	.....	.....	.....
d6	d7	d8	d9	da	db	dc	dd	de	df	e0	e1	e2	e3	e4	e5	.....	.....	.....	.....	.....
e6	e7	e8	e9	ea	eb	ec	ed	ee	ef	f0	f1	f2	f3	f4	f5	.....	.....	.....	.....	.....
f6	f7	f8	f9	fa	fb	fc	fd	fe	ff	00	01	02	03	04	05	.....	.....	.....	.....	.....
06	07	08	09	0a	0b	0c	0d	0e	0f	10	11	12	13	14	15	.....	.....	.....	.....	.....
16	17	18	19	1a	1b	1c	1d	1e	1f	20	21	22	23	24	25	.....	..	!"#\$%	.....	.....
26	27	28	29	2a	2b	2c	2d	2e	2f	30	31	32	33	34	35	&'()	*+, -	./012345	.....	.....
36	37	38	39	3a	3b	3c	3d	3e	3f	40	41	42	43	44	45	6789;	<=	>?@ABCDE	.....	.....
46	47	48	49	4a	4b	4c	4d	4e	4f	50	51	52	53	54	55	FGHIJKLM	NOPQRSTU	.....	.....	.....
56	57	58	59	5a	5b	5c	5d	5e	5f	60	61	62	63	64	65	VWXYZ[\]	^_`abcde	.....	.....	.....
66	67	68	69	6a	6b	6c	6d	6e	6f	70	71	72	73	74	75	fghijklm	nopqrstu	.....	.....	.....
76	77	78	79	7a	7b	7c	7d	7e	7f	80	81	82	83	84	85	vwxyz{ }	~.....	.....	.....	.....
86	87	88	89	8a	8b	8c	8d	8e	8f	90	91	92	93	94	95	.....	.....	.....	.....	.....
96	97	98	99	9a	9b	9c	9d	9e	9f	a0	a1	a2	a3	a4	a5	.....	.....	.....	.....	.....
a6	a7	a8	a9	aa	ab	ac	ad	ae	af	b0	b1	b2	b3	b4	b5	.....	.....	.....	.....	.....
b6	b7	b8	b9	ba	bb	bc	bd	be	bf	c0	c1	c2	c3	c4	c5	.....	.....	.....	.....	.....
c6	c7	c8	c9	ca	cb	cc	cd	ce	cf	d0	d1	d2	d3	d4	d5	.....	.....	.....	.....	.....
d6	d7	d8	d9	da	db	dc	dd	de	df	e0	e1	e2	e3	e4	e5	.....	.....	.....	.....	.....
e6	e7	e8	e9	ea	eb	ec	ed	ee	ef	f0	f1	f2	f3			.....	.....	.....	.....	.....

Рис. 6. Поле данных ping-пакета

Время отправки пакета, затем повторяющаяся последовательность байт 00...ff.

## Анализ трафика утилиты traceroute

Трафик, создаваемый утилитой traceroute, состоит из UDP пакетов с увеличивающимся значением TTL (начиная с 1), причем на каждое значение отправляется три пакета.

Рассмотрим один из запросов утилиты:

```

▶ Frame 17: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface wlp2s0, id 0
▶ Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
▼ Internet Protocol Version 4, Src: 192.168.0.101, Dst: 185.5.53.9
  - 0100 .... = Version: 4
  - .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  - Total Length: 60
  - Identification: 0xc615 (50709)
  ▶ Flags: 0x0000
  - ...0 0000 0000 0000 = Fragment offset: 0
  - Time to live: 5
  - Protocol: UDP (17)
  - Header checksum: 0x4080 [validation disabled]
  - [Header checksum status: Unverified]
  - Source: 192.168.0.101
  - Destination: 185.5.53.9
  ▶ [Destination GeoIP: LT]
▼ User Datagram Protocol, Src Port: 40974, Dst Port: 33446
  - Source Port: 40974
  ▶ Destination Port: 33446
  - Length: 40
  - Checksum: 0x38c8 [unverified]
  - [Checksum Status: Unverified]
  - [Stream index: 13]
  ▶ [Timestamps]
▼ Data (32 bytes)
  - Data: 404142434445464748494a4b4c4d4e4f5051525354555657...
  - [Length: 32]

```

Рис. 7. Запрос traceroute

Используемые протоколы были рассмотрены ранее, за исключением UDP.

## UDP

Заголовок UDP содержит в себе следующие управляющие данные:

- номера портов отправителя и получателя
- размер пакета (8 байт заголовка + размер передаваемых данных)
- контрольная сумма

## ICMP ответ

Если TTL пакета недостаточен, чтобы достигнуть получателя, пакет отбрасывается, а промежуточный маршрутизатор отправляет ICMP сообщение с ошибкой:



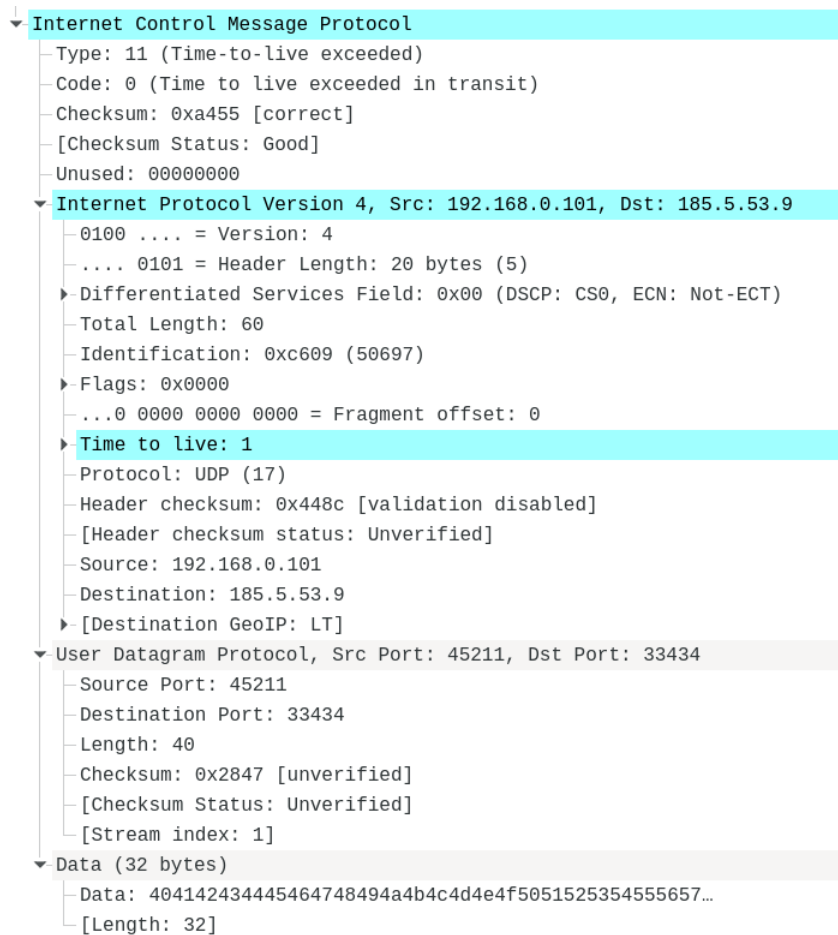


Рис. 8. ICMP TTL exceeded

Если пакет доходит до получателя, то отправляется иное сообщение — в данном случае, сообщение ICMP о том, что порт, к которому обращается traceroute, не найден:

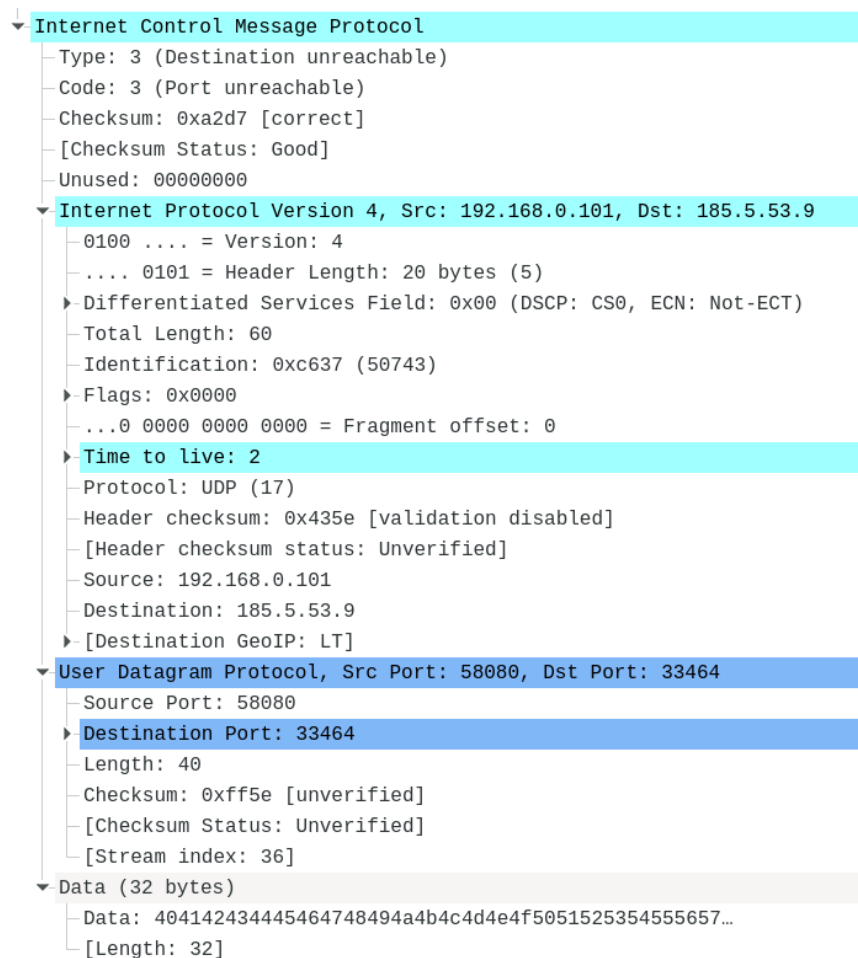


Рис. 9. ICMP port unreachable

### Вопросы к заданию

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

Заголовок IPv4 содержит 20 байт (см. разбор трафика утилиты ping). Поле данных содержит 40 байт и состоит из заголовка UDP (8 байт) и данных (32 байта).

2. Как и почему именно так изменяется поле TTL в следующих друг за другом ICMP-пакетах (проследить изменение TTL в как минимум пяти подряд идущих пакетах)?

Каждые три пакета значение TTL увеличивается на 1. Каждый промежуточный маршрутизатор декрементирует значение TTL и отправляет ошибку при достижении нуля. Сообщения об ошибке позволяют утилите отследить промежуточные маршрутизаторы, через которые пакет проходит до получателя.

3. Чем отличаются ICMP-пакеты, генерируемые утилитой, от ICMP-пакетов, генерируемых утилитой ping

При использовании ключа -I для отправки ICMP запросов вместо UDP, пакеты отличаются только значением TTL.

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

ICMP error пакеты приходят как при отбрасывании пакета промежуточным узлом, так и по достижении получателя. Они отличаются типом ошибки (TTL exceeded, port unreachable).

ICMP reply пакеты отправляются получателем в случае использования ICMP вместо UDP.

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

IP-адреса промежуточных адресов будут преобразованы в доменные имена, в связи с чем будут сгенерированы дополнительные DNS запросы. В traceroute данная функциональность включена по умолчанию и отключается ключом -n.

## Анализ HTTP-трафика

Трафик, создаваемый браузером при посещении сайта, отслеживается с помощью Wireshark. Необходимо сравнить GET-запрос браузера и ответ сервера при первичном посещении страницы и при вторичном запросе (условный GET).

При первичном посещении страницы запрос выглядит следующим образом:

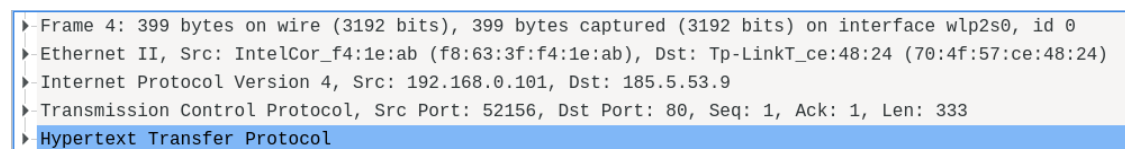


Рис. 10. HTTP GET при первичном посещении

## TCP

Рассмотрим содержимое пакета на уровне TCP, транспортного протокола, который отвечает за установление соединения и обмен байтами между двумя узлами в заданном порядке, с обработкой ошибок.

```

Transmission Control Protocol, Src Port: 52156, Dst Port: 80, Seq: 1, Ack: 1, Len: 333
-Source Port: 52156
-Destination Port: 80
-[Stream index: 0]
-[TCP Segment Len: 333]
-Sequence number: 1      (relative sequence number)
-Sequence number (raw): 261019588
-[Next sequence number: 334      (relative sequence number)]
-Acknowledgment number: 1      (relative ack number)
-Acknowledgment number (raw): 3093609212
-1000 .... = Header Length: 32 bytes (8)
-Flags: 0x018 (PSH, ACK)
- 000. .... = Reserved: Not set
- ...0 .... = Nonce: Not set
- .... 0... = Congestion Window Reduced (CWR): Not set
- .... .0.. = ECN-Echo: Not set
- .... ..0. = Urgent: Not set
- .... ...1 .... = Acknowledgment: Set
- .... .... 1... = Push: Set
- .... .... .0.. = Reset: Not set
- .... .... ..0. = Syn: Not set
- .... .... ...0 = Fin: Not set
- [TCP Flags: .....AP...]
-Window size value: 502
-[Calculated window size: 64256]
-[Window size scaling factor: 128]
-Checksum: 0x7eb9 [unverified]
-[Checksum Status: Unverified]
-Urgent pointer: 0
▶Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
▶[SEQ/ACK analysis]
▶[Timestamps]
-TCP payload (333 bytes)

```

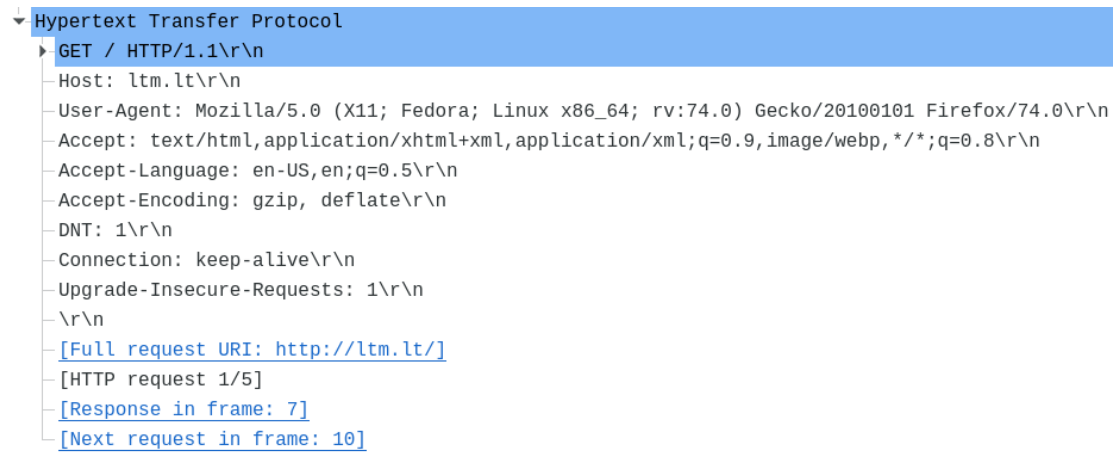
Рис. 11. TCP

Передаваемые метаданные включают в себя порт отправителя и получателя, контрольную сумму, а также следующие управляющие флаги:

- Urgent (URG): указывает, что в заголовке установлен Urgent Pointer (не используется в современных протоколах)
- Acknowledgment (ACK): указывает, что в заголовке установлено поле Acknowledgment, подтверждающее принятие данных
- Push (PSH): указывает получателю, что принятые данные должны быть сразу переданы в приложение, не дожидаясь заполнения буфера
- Reset (RST): обрыв соединения
- Syn: синхронизация номера последовательности, устанавливается в первом пакете, отправленном с каждой стороны
- Fin: помечает отправленный пакет как последний

## HTTP

Рассмотрим содержимое пакета на уровне прикладного протокола HTTP:



The image shows a network packet capture for an HTTP GET request. The packet is expanded to show its contents. The first line is the request line: GET / HTTP/1.1. This is followed by several header lines: Host: ltm.lt, User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86\_64; rv:74.0) Gecko/20100101 Firefox/74.0, Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8, Accept-Language: en-US,en;q=0.5, Accept-Encoding: gzip, deflate, DNT: 1, Connection: keep-alive, and Upgrade-Insecure-Requests: 1. The packet ends with a blank line. Below the headers, there are several informational lines: [Full request URI: http://ltm.lt/], [HTTP request 1/5], [Response in frame: 7], and [Next request in frame: 10].

```
▼ Hypertext Transfer Protocol
  ▶ GET / HTTP/1.1\r\n
  - Host: ltm.lt\r\n
  - User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:74.0) Gecko/20100101 Firefox/74.0\r\n
  - Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
  - Accept-Language: en-US,en;q=0.5\r\n
  - Accept-Encoding: gzip, deflate\r\n
  - DNT: 1\r\n
  - Connection: keep-alive\r\n
  - Upgrade-Insecure-Requests: 1\r\n
  - \r\n
  - [Full request URI: http://ltm.lt/]
  - [HTTP request 1/5]
  - [Response in frame: 7]
  - [Next request in frame: 10]
```

Рис. 12. HTTP GET запрос

Передаваемые данные разделены на следующие секции, разделенные переносом строки (`\r\n`):

- стартовую строку, определяющую метод запроса, запрашиваемый путь и версию протокола
- заголовки, содержащие информацию о клиенте, сервере, соединении и самих данных в виде пар `key: value`
- тело сообщения

Ответ сервера:

```

Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Sun, 12 Apr 2020 08:49:13 GMT\r\n
  Server: Apache\r\n
  Set-Cookie: PHPSESSID=cbaatadtaf0e9crim5q26laer5g; path=/\r\n
  Expires: Thu, 19 Nov 1981 08:52:00 GMT\r\n
  Cache-Control: no-store, no-cache, must-revalidate\r\n
  Pragma: no-cache\r\n
  Connection: Upgrade, Keep-Alive\r\n
  Vary: Accept-Encoding\r\n
  Content-Encoding: gzip\r\n
  Keep-Alive: timeout=2, max=100\r\n
  Transfer-Encoding: chunked\r\n
  Content-Type: text/html\r\n
  \r\n
  [HTTP response 1/5]
  [Time since request: 0.046263619 seconds]
  [Request in frame: 4]
  [Next request in frame: 10]
  [Next response in frame: 24]
  [Request URI: http://ltm.lt/template/styles/main.css]
  HTTP chunked response
  Content-encoded entity body (gzip): 2106 bytes -> 8626 bytes
  File Data: 8626 bytes
Line-based text data: text/html (266 lines)
  <!DOCTYPE html>\n
  <html>\n
  <head>\n
  \t<base href="/" />\n
  \t<meta name="viewport" content="initial-scale=1.0, user-scalable=no" />\n
  \t<meta charset="utf-8" />\n
  \t<!--[if lt IE 9]><script src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.min.js"></script><![endif]-->\n
  \t<title>LTM GARMENTS</title>\n

```

*Рис. 13. HTTP ответ, который не может быть кэширован*

Можно увидеть, что в ответе заголовок `Cache-Control`, отвечающий за кэширование, устанавливается в `no-store`. Это инструктирует браузер не сохранять данные и при повторном обращении выполнить такой же запрос, как и при первом.

Сравним с ответом сервера на запрос статического ресурса, который может быть сохранен в браузерном кэше:

```

Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Sun, 12 Apr 2020 08:49:14 GMT\r\n
  Server: Apache\r\n
  Last-Modified: Sun, 16 Jul 2017 18:06:52 GMT\r\n
  ETag: "6067-5547325a339e3"\r\n
  Accept-Ranges: bytes\r\n
  Content-Length: 24679\r\n
  Cache-Control: max-age=604800\r\n
  Expires: Sun, 19 Apr 2020 08:49:14 GMT\r\n
  Keep-Alive: timeout=2, max=97\r\n
  Connection: Keep-Alive\r\n
  Content-Type: image/png\r\n
  \r\n
  [HTTP response 4/5]
  [Time since request: 0.059388577 seconds]
  [Prev request in frame: 79]
  [Prev response in frame: 560]
  [Request in frame: 562]
  [Next request in frame: 637]
  [Request URI: http://ltm.lt/template/pictures/bg.png]
  File Data: 24679 bytes
  Portable Network Graphics

```

*Рис. 14. HTTP ответ, который может быть кэширован*

При повторной загрузке сайта браузер установит заголовки If-Modified-Since и If-None-Match.

```

Hypertext Transfer Protocol
  GET /template/pictures/bg.png HTTP/1.1\r\n
  Host: ltm.lt\r\n
  User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:74.0) Gecko/20100101 Firefox/74.0\r\n
  Accept: image/webp, */*\r\n
  Accept-Language: en-US,en;q=0.5\r\n
  Accept-Encoding: gzip, deflate\r\n
  DNT: 1\r\n
  Connection: keep-alive\r\n
  Referer: http://ltm.lt/template/styles/main.css\r\n
  Cookie: PHPSESSID=cbatadtaf0e9crim5q26laer5g\r\n
  If-Modified-Since: Sun, 16 Jul 2017 18:06:52 GMT\r\n
  If-None-Match: "6067-5547325a339e3"\r\n
  Cache-Control: max-age=0\r\n
  \r\n
  [Full request URI: http://ltm.lt/template/pictures/bg.png]
  [HTTP request 2/7]
  [Prev request in frame: 1784]
  [Response in frame: 1815]
  [Next request in frame: 1830]

```

*Рис. 15. Запрос кэшированного ресурса*

Поскольку статический ресурс не был изменен после указанной в заголовке даты, сервер отправит ответ со статусом 304 (Not Modified), без данных.

```

Hypertext Transfer Protocol
  HTTP/1.1 304 Not Modified\r\n
  Date: Sun, 12 Apr 2020 08:49:20 GMT\r\n
  Server: Apache\r\n
  Connection: Keep-Alive\r\n
  Keep-Alive: timeout=2, max=99\r\n
  ETag: "6067-5547325a339e3"\r\n
  Expires: Sun, 19 Apr 2020 08:49:20 GMT\r\n
  Cache-Control: max-age=604800\r\n
  \r\n
  [HTTP response 2/7]
  [Time since request: 0.081186810 seconds]
  [Prev request in frame: 1784]
  [Prev response in frame: 1807]
  [Request in frame: 1809]
  [Next request in frame: 1830]
  [Next response in frame: 1835]
  [Request URI: http://ltm.lt/template/pictures/bg.png]

```

Рис. 16. HTTP 304

## Анализ DNS-трафика

DNS — система трансляции между доменными именами и IP-адресами.

### DNS-запрос

```

Frame 53: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface wlp2s0, id 0
  Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
  Internet Protocol Version 4, Src: 192.168.0.101, Dst: 192.168.0.1
  User Datagram Protocol, Src Port: 51341, Dst Port: 53
  Domain Name System (query)
    Transaction ID: 0xcfdc
    Flags: 0x0100 Standard query
      0... .. = Response: Message is a query
      .000 0... .. = Opcode: Standard query (0)
      .... 0... .. = Truncated: Message is not truncated
      .... 1... .. = Recursion desired: Do query recursively
      .... .. 0... .. = Z: reserved (0)
      .... .. 0... .. = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
      ltm.lt: type A, class IN
        Name: ltm.lt
        [Name Length: 6]
        [Label Count: 2]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
      [Response In: 55]

```

Рис. 17. DNS-запрос



Сообщение DNS включает в себя следующие управляющие данные:

- однобитовый флаг Response (QR): является ли сообщение запросом (query = 0) или ответом (response = 1)
- четырехбитовое поле Opcode (OPCODE): тип запроса (стандартный = 0, обратный = 1, статус сервера = 2)
- однобитовый флаг Truncated (TC): было ли сообщение обрезано из-за превышения ограничения на размер
- однобитовый флаг Recursion desired (RD): является ли запрос рекурсивным
- трехбитовое зарезервированное поле
- однобитовый флаг Non-authenticated data: устанавливается в 1, если ответ содержит данные из достоверного источника

В секции Queries содержится информация о запросах:

- имя домена
- тип запроса (A — IPv4 адрес, AAAA — IPv6 адрес)
- класс запроса, устанавливается в IN для хостов в сети Интернет

## DNS-ответ

```
▶ Frame 55: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface wlp2s0, id 0
▶ Ethernet II, Src: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
▶ Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.101
▶ User Datagram Protocol, Src Port: 53, Dst Port: 51341
▼ Domain Name System (response)
  Transaction ID: 0xcfdc
  Flags: 0x8180 Standard query response, No error
    1... .. = Response: Message is a response
    .000 0... .. = Opcode: Standard query (0)
    .... .0.. .. = Authoritative: Server is not an authority for domain
    .... ..0. .... = Truncated: Message is not truncated
    .... ..1 .... = Recursion desired: Do query recursively
    .... ..1... .. = Recursion available: Server can do recursive queries
    .... ..0.. .... = Z: reserved (0)
    .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
    .... ..0 .... = Non-authenticated data: Unacceptable
    .... ..0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 1
  Authority RRs: 0
  Additional RRs: 0
  Queries
    ▼ ltm.lt: type A, class IN
      Name: ltm.lt
      [Name Length: 6]
      [Label Count: 2]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
  Answers
    ▼ ltm.lt: type A, class IN, addr 185.5.53.9
      Name: ltm.lt
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 3600 (1 hour)
      Data length: 4
      Address: 185.5.53.9
    [Request In: 53]
    [Time: 0.088145566 seconds]
```

Рис. 18. DNS-ответ

В добавление к данным, передаваемым в запросе, добавляются:

- однобитовый флаг Authoritative (AA): является ли сервер авторитативным для данного домена
- однобитовый флаг Recursion available (RA): предоставляет ли сервер возможность выполнения рекурсивных запросов
- поле Answer authenticated: является ли ответ удостоверенным
- четырехбитовое поле Reply code (RCODE): 0 = успешный ответ, 1 = FORMERR (неверный запрос), 2 = SERVFAIL (ошибка сервера), 3 = NXDOMAIN (несуществующий домен).

В секции Answers содержится информация об ответах сервера:

- имя домена

- тип запроса
- класс запроса
- время, в течение которого информация остается валидной, в секундах
- запрошенная информация (в данном случае, IP-адрес)

### Вопросы к заданию

1. Почему адрес, на который отправлен DNS-запрос, не совпадает с адресом посещаемого сайта?

DNS-запрос отправляется на адрес DNS-сервера, чтобы узнать IP-адрес посещаемого сайта.

2. Какие бывают типы DNS-запросов?
  - итеративный: DNS-сервер не опрашивает другие серверы
  - рекурсивный: DNS-сервер может обращаться к другим серверам
  - обратный: запрос доменного имени по IP-адресу
3. В какой ситуации нужно выполнять независимые DNS-запросы для получения содержащихся на сайте изображений?

В случае, если изображения размещены на другом доменном имени (в том числе поддомене).

### Анализ ARP-трафика

ARP — протокол обнаружения MAC-адресов устройств (канальный уровень) по адресам сетевого уровня (IP).

#### ARP-запрос

```

▶ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface wlp2s0, id 0
▶ Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▼ Address Resolution Protocol (request)
  - Hardware type: Ethernet (1)
  - Protocol type: IPv4 (0x0800)
  - Hardware size: 6
  - Protocol size: 4
  - Opcode: request (1)
  - Sender MAC address: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
  - Sender IP address: 192.168.0.101
  - Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  - Target IP address: 192.168.0.1
  
```

Рис. 19. ARP-запрос

ARP-запрос содержит следующие управляющие данные:

- канальный протокол (hardware type)
- сетевой протокол (protocol type)
- длина адреса канального протокола в байтах (hardware size, 8 для MAC-адреса)

- длина адреса сетевого протокола в байтах (`protocol size`, 4 для IPv4-адреса)
- код операции (`opcode`, 1 = запрос, 2 = ответ)
- MAC/IP-адреса отправителя (`sender hardware/protocol address`)
- MAC/IP-адреса получателя (`target hardware/protocol address`, MAC-адрес в запросе игнорируется)

```

▶ Frame 2: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface wlp2s0, id 0
▶ Ethernet II, Src: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
▼ Address Resolution Protocol (reply)
  - Hardware type: Ethernet (1)
  - Protocol type: IPv4 (0x0800)
  - Hardware size: 6
  - Protocol size: 4
  - Opcode: reply (2)
  - Sender MAC address: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
  - Sender IP address: 192.168.0.1
  - Target MAC address: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
  - Target IP address: 192.168.0.101

```

Рис. 20. ARP-ответ

ARP-ответ посылается узлом с запрашиваемым IP-адресом отправителю запроса, MAC/IP-адреса полностью заполнены.

### Вопросы к заданию

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

MAC-адреса уникально идентифицируют сетевые устройства. Они присваиваются производителем (хотя устройство может включать возможность перезаписи своего MAC-адреса). В присвоенный производителем адрес включается его уникальный идентификатор, что можно увидеть в захваченных пакетах: они соответствуют встроенному в компьютер сетевому адаптеру Intel (f8:63:3f:f4:1e:ab) и маршрутизатору Tp-Link (70:4f:57:ce:48:24).

Также встречается широковещательный адрес ff:ff:ff:ff:ff:ff, запросы на который воспринимаются всеми узлами сети.

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

В HTTP-пакетах присутствуют MAC-адреса обозначенных выше устройств (компьютера и маршрутизатора) на канальном уровне (в заголовке Ethernet II). При запросе отправителем является компьютер, при ответе сервера — маршрутизатор.

3. Для чего ARP-запрос содержит IP-адрес источника?

Это позволяет принимающему узлу добавить отправителя в свою ARP-таблицу без необходимости повторных запросов.

## Анализ трафика утилиты nslookup

Трафик nslookup состоит из DNS-запросов A (записи, содержащей IPv4 адрес хоста) и AAAA (записи, содержащей IPv6 адрес хоста).

Запросы аналогичны рассмотренным в разделе *Анализ DNS-трафика*. Отдельно стоит рассмотреть ответ сервера об отсутствии записи, который возвращается для AAAA.

### Ответ об отсутствии информации

```
Frame 4: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits) on interface wlp2s0, id 0
Ethernet II, Src: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.101
User Datagram Protocol, Src Port: 53, Dst Port: 47044
Domain Name System (response)
  Transaction ID: 0x4f8f
  Flags: 0x8180 Standard query response, No error
    1... .. = Response: Message is a response
    .000 0... .. = Opcode: Standard query (0)
    .... 0... .. = Authoritative: Server is not an authority for domain
    .... ..0. .... = Truncated: Message is not truncated
    .... ..1. .... = Recursion desired: Do query recursively
    .... ..1... .. = Recursion available: Server can do recursive queries
    .... ..0... .. = Z: reserved (0)
    .... ..0... .. = Answer authenticated: Answer/authority portion was not authenticated by the server
    .... ..0... .. = Non-authenticated data: Unacceptable
    .... ..0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 0
  Authority RRs: 1
  Additional RRs: 0
  Queries
    ltm.lt: type AAAA, class IN
      Name: ltm.lt
      [Name Length: 6]
      [Label Count: 2]
      Type: AAAA (IPv6 Address) (28)
      Class: IN (0x0001)
  Authoritative nameservers
    ltm.lt: type SOA, class IN, mname ns1.serveriai.lt
      Name: ltm.lt
      Type: SOA (Start Of a zone of Authority) (6)
      Class: IN (0x0001)
      Time to live: 317 (5 minutes, 17 seconds)
      Data length: 56
      Primary name server: ns1.serveriai.lt
      Responsible authority's mailbox: hostmaster.iv.lt
      Serial Number: 2020041200
      Refresh Interval: 43200 (12 hours)
      Retry Interval: 3600 (1 hour)
      Expire limit: 1209600 (14 days)
      Minimum TTL: 3600 (1 hour)
  [Request In: 3]
  [Time: 0.005354432 seconds]
```

Рис. 21. Ответ об отсутствии записи AAAA

Код ответа (Reply code) равен 0 (No error), при этом количество ответов (Answer RRs) также равно нулю.

В ответе содержится запись SOA (Start of authority) о сервере, который отвечает за доменное имя. Согласно секции 2.2.1 RFC 2308, она отправляется для того, чтобы клиент мог кэшировать ответ, основываясь на Minimum TTL полученной записи.

### nslookup -type=NS

nslookup -type=NS выполняет запрос NS — записи, содержащей авторитативный DNS-сервер для доменного имени:

```
Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface wlp2s0, id 0
Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
Internet Protocol Version 4, Src: 192.168.0.101, Dst: 192.168.0.1
User Datagram Protocol, Src Port: 38298, Dst Port: 53
Domain Name System (query)
  Transaction ID: 0x621d
  Flags: 0x0100 Standard query
    0... .. = Response: Message is a query
    .000 0... .. = Opcode: Standard query (0)
    .... 0... .. = Truncated: Message is not truncated
    .... 1... .. = Recursion desired: Do query recursively
    .... .. 0... .. = Z: reserved (0)
    .... .. 0... .. = Non-authenticated data: Unacceptable
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 0
  Queries
    ltm.lt: type NS, class IN
      Name: ltm.lt
      [Name Length: 6]
      [Label Count: 2]
      Type: NS (authoritative Name Server) (2)
      Class: IN (0x0001)
  [Response In: 2]
```

Рис. 22. Запрос nslookup -type=NS

Ответ может содержать несколько серверов — основной и запасные:

```

Frame 2: 270 bytes on wire (2160 bits), 270 bytes captured (2160 bits) on interface wlp2s0, id 0
Ethernet II, Src: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.101
User Datagram Protocol, Src Port: 53, Dst Port: 38298
Domain Name System (response)
  Transaction ID: 0x621d
  Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 4
  Authority RRs: 0
  Additional RRs: 6
  Queries
    ltm.lt: type NS, class IN
      Name: ltm.lt
      [Name Length: 6]
      [Label Count: 2]
      Type: NS (authoritative Name Server) (2)
      Class: IN (0x0001)
  Answers
    ltm.lt: type NS, class IN, ns ns2.serveriai.lt
      Name: ltm.lt
      Type: NS (authoritative Name Server) (2)
      Class: IN (0x0001)
      Time to live: 7109 (1 hour, 58 minutes, 29 seconds)
      Data length: 18
      Name Server: ns2.serveriai.lt
    ltm.lt: type NS, class IN, ns ns4.serveriai.lt
    ltm.lt: type NS, class IN, ns ns1.serveriai.lt
    ltm.lt: type NS, class IN, ns ns3.serveriai.lt
  Additional records
    ns1.serveriai.lt: type A, class IN, addr 79.98.25.142
      Name: ns1.serveriai.lt
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 6429 (1 hour, 47 minutes, 9 seconds)
      Data length: 4
      Address: 79.98.25.142
    ns2.serveriai.lt: type A, class IN, addr 79.98.29.142
    ns3.serveriai.lt: type A, class IN, addr 162.159.24.19
    ns4.serveriai.lt: type A, class IN, addr 162.159.25.253
    ns3.serveriai.lt: type AAAA, class IN, addr 2400:cb00:2049:1::a29f:1813
    ns4.serveriai.lt: type AAAA, class IN, addr 2400:cb00:2049:1::a29f:19fd
  [Request In: 1]
  [Time: 0.010052961 seconds]

```

Рис. 23. Ответ nslookup -type=NS

В ответ также включена дополнительная информация — IP-адреса серверов.

### Вопросы к заданию

1. Чем различается трасса трафика в п.2 и п.4, указанных выше?

nslookup запрашивает IPv4 и IPv6 адреса сайта, а nslookup -type=NS — информацию об авторитативных DNS-серверах для сайта.

2. Что содержится в поле Answers DNS-ответа?

Содержание запрошенных DNS-записей.

3. Каковы имена серверов, возвращающих авторитативный (authoritative) отклик?

Для рассмотренного сайта — `ns1.serveriai.lt`, `ns2.serveriai.lt`, `ns3.serveriai.lt`, `ns4.serveriai.lt`.

## Анализ FTP-трафика

FTP — протокол прикладного уровня для передачи файлов по сети.

Рассмотрим пример запроса авторизации:

```
Frame 49: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface wlp2s0, id 0
Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24)
Internet Protocol Version 4, Src: 192.168.0.101, Dst: 91.121.146.196
Transmission Control Protocol, Src Port: 60854, Dst Port: 21, Seq: 1, Ack: 65, Len: 16
File Transfer Protocol (FTP)
  USER anonymous\r\n
    Request command: USER
    Request arg: anonymous
  [Current working directory: ]
```

Рис. 24. FTP-запрос *USER anonymous*

Клиент отправляет команду `USER` с аргументом `anonymous`, запрашивая анонимный доступ.

Для подтверждения авторизации сервер отправляет следующий ответ:

```
Frame 51: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface wlp2s0, id 0
Ethernet II, Src: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
Internet Protocol Version 4, Src: 91.121.146.196, Dst: 192.168.0.101
Transmission Control Protocol, Src Port: 21, Dst Port: 60854, Seq: 65, Ack: 17, Len: 75
File Transfer Protocol (FTP)
  331 Anonymous login ok, send your complete email address as your password\r\n
    Response code: User name okay, need password (331)
    Response arg: Anonymous login ok, send your complete email address as your password
  [Current working directory: ]
```

Рис. 25. FTP-ответ *331*

Трехзначный `Response code` кодируется тремя символами ASCII. За ним через пробел следует предназначенное для пользователя информационное сообщение.

Для скачивания файла клиент и сервер обменялись следующими сообщениями:

1. К: `USER anonymous` (запрос анонимного доступа)
2. С: `331 ...` (подтверждение анонимного логина и запрос пароля)
3. К: `PASS mozilla@example.com` (в качестве пароля при анонимном доступе передается любой email-адрес)



4. C: 230 ... (подтверждение доступа)
5. K: SYST (запрос типа системы)
6. C: 215 UNIX Type: L8 (тип, возвращаемый большинством серверов)
7. K: FEAT (запрос поддерживаемых сервером возможностей)
8. C: 211 Features:\r\n211 ... \r\n211 ... \r\n211 End (перечисление поддерживаемых возможностей)
9. K: PWD (запрос текущей директории)
10. C: "/" is the current directory
11. K: TYPE I (установить бинарную передачу файлов, альтернативно TYPE A — текстовую)
12. C: 200 ...
13. K: PASV (перейти в пассивный режим, при котором отдельное соединение для передачи данных открывает клиент, а не сервер)
14. C: 227 Entering Passive Mode (91,121,146,196,175,145) . (сервер ожидает подключения по IP-адресу 91.121.146.196 и порту  $175 \ll 8 \vee 145 = 44945$ )
15. K: SIZE /dists/buster/Release.gpg (запрос размера скачиваемого файла)
16. C: 213 833 (размер равен 833 байта)
17. K: MDTM /dists/buster/Release.gpg (запрос время последнего изменения файла)
18. C: 213 20200415080226 (2020.04.15 08:02:26)
19. K: RETR /dists/buster/Release.gpg (запрос на скачивание файла)
20. C: 150 ... (открытие data соединения)

FTP-DATA выглядит следующим образом:

```

> Frame 168: 899 bytes on wire (7192 bits), 899 bytes captured (7192 bits) on interface wlp2s0, id 0
> Ethernet II, Src: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab)
> Internet Protocol Version 4, Src: 91.121.146.196, Dst: 192.168.0.101
> Transmission Control Protocol, Src Port: 41173, Dst Port: 44222, Seq: 1, Ack: 1, Len: 833
  FTP Data (833 bytes data)
    [Setup frame: 158]
    [Setup method: PASV]
    [Command: SIZE /dists/buster/Release.gpg]
    Command frame: 159
    [Current working directory: /]
  Line-based text data (16 lines)
    -----BEGIN PGP SIGNATURE-----\n
    \n
    iQIzBAABCAAdFiEEpAH/mTaPofmBUT51XICMK2VVgRcFAl6wsxsACgkQXICMK2VV\n
    gRexuQ/7BBG9+9B17cIEhUY4DuU0eT+IE5dWM+nzS9UszNZ7gUh7l/oyntaLfQUB\n
    edV4+WykM376SYmapHMYfnKq1UD+ZQEjZsRuuS3S2a4lVoKvQHk99l4pu+2xFiPK\n
    mm7ZXRJnpGCTabZrRjWbc000eBnqQ6iGZauPHLUJ71/VimFD/oZQ0mPG3EDPe0J2\n
    Ag0FNrV+x0anSJNY28gdkidUM2EnRY5u/N96mAgVULehS5Q3ZgdC5oxCTg6QUA\n
    dAi9k5ZjyjnsHNo3yDLsle9hDKhLPyahmYSFNkRtCR4vrNjBG4t1nIyEvD7aurrm\n
    LTRMBYrpMJLAgb7NlHvpcIzCZSqe5f4i1C0t2q+VrYqKsSDn4gIIqLKPYPiE4bAP7\n
    MMA0UkSI566WNk3ZX4cn2UieLYoAXct7g7u0YQScpqhZvHMjNMnjcKCgsbuRYZd3\n
    jfKhsmqINEA+59l9cuDsvv8i/aY8l3LTye/+r5LuDCw0tvawQ52BxTTd1rgq7QQx\n
    ZfbanB2d8elGg+xYmfhF+xVqApnFfh4pfKUYOqRVHZLqzlwWdV0dAoZCeCvr1yDK\n
    x2pYxPZ+VrWrx9kG5b8hN0AtJvFrKUx4dXLD/kq7sfZNBVBNT0cmE5PkuV9fltjM\n
    x9EPfRfWRHJ0r5/RHKsiOXUjrPZFyv0BI/2hgwtIoKkDQfLIvrc=\n
    =AVHN\n
    -----END PGP SIGNATURE-----\n

```

Рис. 26. FTP-DATA

### Вопросы к заданию

1. Сколько байт данных содержится в пакете FTP-DATA?

Один пакет может содержать до 1460 байт данных, поскольку MTU = 1500 байт, из которых 20 байт занимает заголовок IP и 20 байт заголовок TCP.

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

На клиенте выбирается случайный порт в диапазоне 1024 до 65535 (0-1023 зарезервированы системой). На сервере для FTP используется порт 21.

3. Чем отличаются пакеты FTP от FTP-DATA?

FTP предназначен для передачи команд и ответов сервера, в то время как FTP-DATA — для передачи данных.

### Анализ DHCP-трафика

DHCP — протокол динамической настройки узла сети, позволяющий получать IP-адрес и другие опции (например, маску подсети) от сервера. DHCP работает поверх протокола UDP.

Цикл сброса-запроса адреса состоит из следующих обращений:

1. DHCP Release — клиент информирует сервер об освобождении адреса
2. DHCP Discover — клиент отправляет широковещательное сообщение с запросом информации о сети
3. DHCP Offer — один или несколько серверов отправляют клиенту информацию о доступном адресе и других параметрах сети
4. DHCP Request — клиент запрашивает предложенный IP у конкретного сервера, остальные сервера освобождают адрес, зарезервированный на этапе DHCP Offer
5. DHCP ACK — сервер подтверждает конфигурацию и сообщает клиенту о времени действия адреса, по истечении которого клиент должен повторить процесс конфигурации

No.	Time	Source	Destination	Protocol	Length	Info
102	0.892443607	192.168.0.101	192.168.0.1	DHCP	342	DHCP Release - Transaction ID 0x4aa5f84d
120	1.087517712	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x72e8359
121	1.096571622	192.168.0.1	192.168.0.101	DHCP	590	DHCP Offer - Transaction ID 0x72e8359
123	1.096710257	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x72e8359
151	1.608960071	192.168.0.1	192.168.0.101	DHCP	590	DHCP ACK - Transaction ID 0x72e8359
390	4.823685133	192.168.0.101	192.168.0.1	DHCP	342	DHCP Release - Transaction ID 0x7a54fe0d
404	5.057640750	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0xccb2bb48
406	5.076486118	192.168.0.1	192.168.0.101	DHCP	590	DHCP Offer - Transaction ID 0xccb2bb48
407	5.076746057	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0xccb2bb48
457	5.595777451	192.168.0.1	192.168.0.101	DHCP	590	DHCP ACK - Transaction ID 0xccb2bb48

▶ Frame 102: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface wlp2s0, id 0 ▶ Ethernet II, Src: IntelCor_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT_ce:48:24 (70:4f:57:ce:48:24) ▶ Internet Protocol Version 4, Src: 192.168.0.101, Dst: 192.168.0.1 ▶ User Datagram Protocol, Src Port: 68, Dst Port: 67 ▶ Dynamic Host Configuration Protocol (Release)
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Рис. 27. Цикл сброса-запроса адреса

## DHCP-запрос

В качестве примера запроса рассмотрим отправляемое клиентом сообщение DHCP Discover:

[illegible]

### *Рис. 28. Запрос DHCP Discover*

Сообщение отправляется всем узлам, поэтому IP-адрес (на транспортном уровне) выставляется в 255.255.255.255.

Сообщение содержит в себе следующие поля:

- тип сообщения, ор (1 = запрос, 2 = ответ сервера)
- тип канального протокола, htype (Ethernet)
- длина адреса на уровне канального протокола, hlen (6)

- количество промежуточных маршрутизаторов, через которые прошло сообщение, hops (выставлено клиентом в 0)
- уникальный идентификатор транзакции, xid (генерируется клиентом для DHCP Discover и сохраняется до получения DHCP ACK)
- время с момента начала транзакции, secs (может не использоваться)
- флаги BOOTP, flags (может не использоваться)
- IP-адрес клиента, ciaddr (выставляется клиентом в 0)
- IP-адрес, предложенный сервером, yiaddr (выставляется клиентом в 0)
- IP-адрес сервера, siaddr (выставляется в DHCP Offer сервером и в DHCP Request клиентом)
- IP-адрес промежуточного узла, giaddr
- адрес клиента на уровне канального протокола, chaddr (MAC)
- имя хоста сервера, sname (может не использоваться)
- имя загрузочного файла, file (использовалось в BOOTP)
- magic cookie, обозначающее начало опций (выставляется в 0x63825363)

Затем следуют DHCP-опции:

- запрашиваемый адрес (выставлен клиентом в выданный ранее)
- запрашиваемые параметры (маска подсети, DNS-сервер и т.д.)

#### DHCP-ответ

В качестве примера ответа рассмотрим отправляемое сервером сообщение DHCP Offer:

[illegible]

## Последовательность обмена

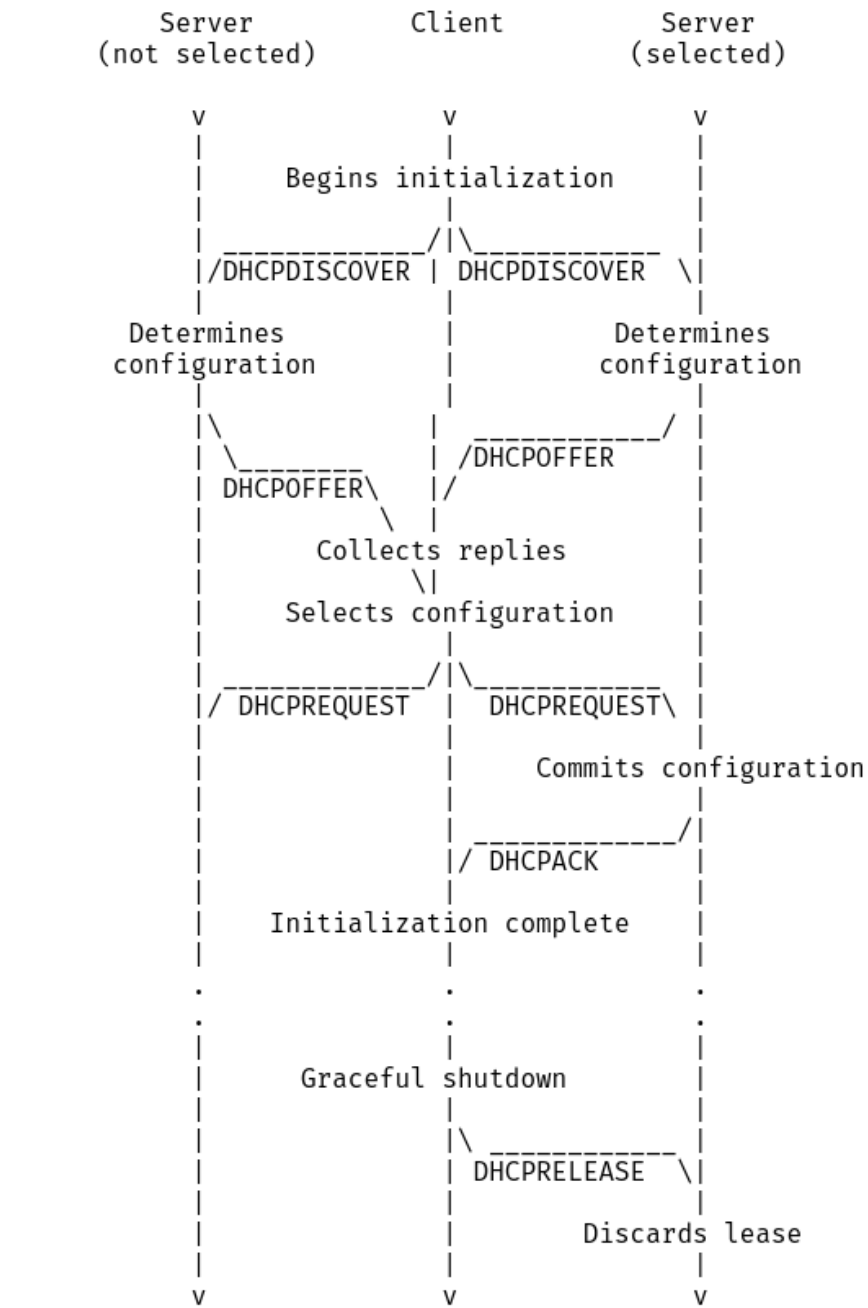


Рис. 30. Диаграмма запроса адреса

### Вопросы к заданию

1. Чем различаются пакеты DHCP Discover и DHCP Request?

DHCP Discover посылается для обнаружения серверов и запроса конфигурации, DHCP Request используется для выбора сервера.

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

В DHCP Discover и DHCP Request пакетах IP-адрес источника устанавливается в 0.0.0.0, поскольку адрес ему еще не присвоен.

3. Каков IP-адрес DHCP-сервера?

192.168.0.1 (см. DHCP Offer)

4. Что произойдет, если очистить использованный фильтр "bootp"?

Будут отображены все пакеты, захваченные во время анализа.

## Анализ Skype-трафика

### Текстовые сообщения

При передаче сообщений Skype использует протокол TCP с шифрованием (TLSv1.2):



9	0.701224074	52.149.21.60	192.168.0.101	TLSv1.2	538 Application Data
13	1.293622506	192.168.0.101	52.149.21.60	TLSv1.2	517 Application Data, Application Data
15	1.571400648	52.149.21.60	192.168.0.101	TLSv1.2	844 Application Data
19	1.641491624	192.168.0.101	52.149.21.60	TLSv1.2	240 Application Data
20	1.671603987	52.149.21.60	192.168.0.101	TLSv1.2	538 Application Data
24	1.792537870	192.168.0.101	52.149.21.60	TLSv1.2	388 Application Data, Application Data
29	2.060856850	52.149.21.60	192.168.0.101	TLSv1.2	980 Application Data
32	2.068814571	192.168.0.101	52.149.21.60	TLSv1.2	240 Application Data
33	2.073456661	52.149.21.60	192.168.0.101	TLSv1.2	320 Application Data
42	6.180214730	184.28.181.6	192.168.0.101	TLSv1.2	97 Encrypted Alert
47	9.699518233	192.168.0.101	52.114.75.149	TLSv1.2	835 Application Data
49	9.699573081	192.168.0.101	52.114.75.149	TLSv1.2	748 Application Data
52	10.279546260	52.114.75.149	192.168.0.101	TLSv1.2	492 Application Data
54	10.353793413	52.149.21.60	192.168.0.101	TLSv1.2	964 Application Data

▶ Frame 9: 538 bytes on wire (4304 bits), 538 bytes captured (4304 bits) on interface wlp2s0, id 0  
 ▶ Ethernet II, Src: Tp-LinkT\_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor\_f4:1e:ab (f8:63:3f:f4:1e:ab)  
 ▶ Internet Protocol Version 4, Src: 52.149.21.60, Dst: 192.168.0.101  
 ▼ Transmission Control Protocol, Src Port: 443, Dst Port: 52480, Seq: 1, Ack: 3242, Len: 484

- Source Port: 443
- Destination Port: 52480
- [Stream index: 1]
- [TCP Segment Len: 484]
- Sequence number: 1 (relative sequence number)
- Sequence number (raw): 891207257
- [Next sequence number: 485 (relative sequence number)]
- Acknowledgment number: 3242 (relative ack number)
- Acknowledgment number (raw): 563800236
- 0101 ... = Header Length: 20 bytes (5)
- ▶ Flags: 0x018 (PSH, ACK)
- Window size value: 2051
- [Calculated window size: 2051]
- [Window size scaling factor: -1 (unknown)]
- Checksum: 0x8c64 [unverified]
- [Checksum Status: Unverified]
- Urgent pointer: 0
- ▶ [SEQ/ACK analysis]
- ▶ [Timestamps]
- TCP payload (484 bytes)

▼ Transport Layer Security

- ▼ TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
  - Content Type: Application Data (23)
  - Version: TLS 1.2 (0x0303)
  - Length: 479
  - Encrypted Application Data: 000000000000000029d8e2e661b252279031eb91520b7a5c3...

Рис. 31. Передача текстовых сообщений в Skype

## Аудио

Во время звонка Skype передает запись голоса с использованием протокола UDP:

No.	Time	Source	Destination	Protocol	Length	Info
2373	36.270595586	192.168.0.101	52.169.248.48	UDP	152	38494 → 3480 Len=110
2374	36.279978921	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2375	36.292189087	192.168.0.101	52.169.248.48	UDP	162	38494 → 3480 Len=120
2376	36.298756119	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2377	36.313103944	192.168.0.101	52.169.248.48	UDP	148	38494 → 3480 Len=106
2378	36.318328756	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2379	36.329061443	192.168.0.101	52.169.248.48	UDP	148	38494 → 3480 Len=106
2380	36.337872929	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2381	36.350305841	192.168.0.101	52.169.248.48	UDP	159	38494 → 3480 Len=117
2382	36.357963051	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2383	36.371320285	192.168.0.101	52.169.248.48	UDP	159	38494 → 3480 Len=117
2384	36.378206283	52.169.248.48	192.168.0.101	UDP	255	3480 → 38494 Len=213
2385	36.392494216	192.168.0.101	52.169.248.48	UDP	164	38494 → 3480 Len=122
2386	36.399862363	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2387	36.408108478	192.168.0.101	52.169.248.48	UDP	141	38494 → 3480 Len=99
2388	36.419380219	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2389	36.429083267	192.168.0.101	52.169.248.48	UDP	139	38494 → 3480 Len=97
2390	36.438369383	52.169.248.48	192.168.0.101	UDP	247	3480 → 38494 Len=205
2391	36.449987724	192.168.0.101	52.169.248.48	UDP	144	38494 → 3480 Len=102

▶ Frame 2373: 152 bytes on wire (1216 bits), 152 bytes captured (1216 bits) on interface wlp2s0, id 0

▶ Ethernet II, Src: IntelCor\_f4:1e:ab (f8:63:3f:f4:1e:ab), Dst: Tp-LinkT\_ce:48:24 (70:4f:57:ce:48:24)

▶ Internet Protocol Version 4, Src: 192.168.0.101, Dst: 52.169.248.48

▼ User Datagram Protocol, Src Port: 38494, Dst Port: 3480

- Source Port: 38494
- Destination Port: 3480
- Length: 118
- Checksum: 0x452b [unverified]
- [Checksum Status: Unverified]
- [Stream index: 10]
- ▶ [Timestamps]

▼ Data (110 bytes)

- Data: ff10006a316517b50e8c2c389068340d147576ac0000b869...
- [Length: 110]

Рис. 32. Передача аудио в Skype

При рассмотрении захваченного трафика без видео было отмечено, что размер пакетов не превышает 600 байт.

## Видео

Видеоданные также передаются через UDP:

No.	Time	Source	Destination	Protocol	Length	Info
1753	24.777247623	13.80.154.3	192.168.0.101	UDP	153	3480 → 21952 Len=111
1754	24.795178803	13.80.154.3	192.168.0.101	UDP	215	3480 → 21952 Len=173
1755	24.795178993	13.80.154.3	192.168.0.101	UDP	231	3480 → 21952 Len=189
1756	24.795179043	13.80.154.3	192.168.0.101	UDP	151	3480 → 21952 Len=109
1757	24.804024919	13.80.154.3	192.168.0.101	UDP	116	3480 → 21952 Len=74
1758	24.804025032	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1759	24.804025069	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1760	24.804025106	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1761	24.804078443	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1762	24.804078493	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1763	24.804078530	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1764	24.804078567	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1765	24.804078605	13.80.154.3	192.168.0.101	UDP	1097	3480 → 21952 Len=1055
1766	24.804078641	13.80.154.3	192.168.0.101	UDP	1093	3480 → 21952 Len=1051
1767	24.804078679	13.80.154.3	192.168.0.101	UDP	1113	3480 → 21952 Len=1071
1768	24.812985358	13.80.154.3	192.168.0.101	UDP	158	3480 → 21952 Len=116
1769	24.833196693	13.80.154.3	192.168.0.101	UDP	145	3480 → 21952 Len=103
1770	24.854383526	13.80.154.3	192.168.0.101	UDP	149	3480 → 21952 Len=107
1771	24.865728178	192.168.0.101	13.80.154.3	UDP	368	21952 → 3480 Len=326
1772	24.865756614	192.168.0.101	13.80.154.3	UDP	118	21952 → 3480 Len=76
1773	24.865765387	192.168.0.101	13.80.154.3	UDP	1093	21952 → 3480 Len=1051

▶ Frame 1758: 1097 bytes on wire (8776 bits), 1097 bytes captured (8776 bits) on interface wlp2s0, id 0

▶ Ethernet II, Src: Tp-LinkT\_ce:48:24 (70:4f:57:ce:48:24), Dst: IntelCor\_f4:1e:ab (f8:63:3f:f4:1e:ab)

▶ Internet Protocol Version 4, Src: 13.80.154.3, Dst: 192.168.0.101

▼ User Datagram Protocol, Src Port: 3480, Dst Port: 21952

- Source Port: 3480
- Destination Port: 21952
- Length: 1063
- Checksum: 0x695d [unverified]
- [Checksum Status: Unverified]
- [Stream index: 10]
- ▶ [Timestamps]

▼ Data (1055 bytes)

- Data: 917a5ed8052e19b2000007d600000002bede000112d90a72...
- Length: 1055

Рис. 33. Передача видео и аудио в Skype

### Вопросы к заданию

1. Чем различаются пакеты разных видов Skype-трафика (текст, аудио, видео)?

Текст передается TCP-пакетами с шифрованием. Аудио и видео передаются через UDP, причем один и тот же порт используется для обоих видов трафика.

Согласно справке Skype, программа может использовать следующие порты:

- 443/TCP
- 3478-3481/UDP
- 50000-60000/UDP

В захваченных пакетах для порт 443 использовался для текста, 3480 — для аудио и видео.

2. Какой Wireshark-фильтр следует использовать для независимой идентификации Skype-трафика разных видов (текст, аудио, видео)?

Для идентификации текста можно использовать фильтр `tls && tls.record.content_type == "Application Data"`. Он отделяет пакеты, переданные с шифрованием TLS, при этом отбрасывает пакеты, управляющие соединением (обмен ключами шифрования, handshake). Стоит отметить, что Skype использует стандартный порт 443, поэтому среди пакетов могут встретиться данные других приложений.

Для идентификации аудио и видео достаточно фильтра `udp`. Порт выбирается приложением случайно, в захваченных пакетах использовался порт 3480, поэтому можно использовать более точное выражение `udp.port == 3480`.

Разделить аудио и видео данные можно по размеру пакета (см. разбор аудио трафика). Таким образом получим следующие фильтры:

- аудио: `udp.port == 3480 && udp.length < 600`
- видео: `udp.port == 3480 && udp.length > 600`