

Федеральное государственное автономное образовательное учреждение высшего  
образования "Национальный Исследовательский Университет ИТМО"  
Мегафакультет Компьютерных Технологий и Управления  
Факультет Программной Инженерии и Компьютерной Техники



**Лабораторная работа 4**  
*"Локальные сети"*  
по дисциплине  
**Компьютерные сети**

Выполнил Студент группы Р33102  
**Лапин Алексей Александрович**  
Преподаватель:  
**Авксентьева Елена Юрьевна**

г. Санкт-Петербург  
2024г.

# Содержание

Цель работы	3
1 Анализ трафика утилиты ping	3
2 Анализ трафика утилиты tracert (traceroute)	6
3 Анализ HTTP-трафика	9
4 Анализ DNS-трафика	10
5 Анализ ARP-трафика	11
6 Анализ трафика утилиты nslookup	13
7 Анализ FTP-трафика	16
8 Анализ DHCP-трафика	19
9 Структуры наблюдаемых пакетов заголовков	20
10 Выводы	29

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

**Вариант:** [www.alexeylapin.ru](http://www.alexeylapin.ru), так как в нем лексически входит фамилия студента.

## 1 Анализ трафика утилиты ping

Команда, запускаемая в терминале, выглядит следующим образом:

```
aleksejlapin@MacBook-Pro-Aleksej-2: scripts % ping -s 3000 -c 5 www.alexeylapin.ru
PING www.alexeylapin.ru (81.177.139.247): 3000 data bytes
3008 bytes from 81.177.139.247: icmp_seq=0 ttl=255 time=90.889 ms
3008 bytes from 81.177.139.247: icmp_seq=1 ttl=255 time=89.150 ms
3008 bytes from 81.177.139.247: icmp_seq=2 ttl=255 time=91.921 ms
3008 bytes from 81.177.139.247: icmp_seq=3 ttl=255 time=91.282 ms
3008 bytes from 81.177.139.247: icmp_seq=4 ttl=255 time=90.268 ms
```

Аргумент **-s** отвечает за размер пакета.

Аргумент **-c** отвечает за количество пакетов.

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

ICMP пакет передается внутри пакета IP, который передается по Ethernet. Минимальный размер кадра Ethernet — 64 байта, максимальный — 1518 байт. Соответственно, при передаче пакет ICMP фрагментируется.

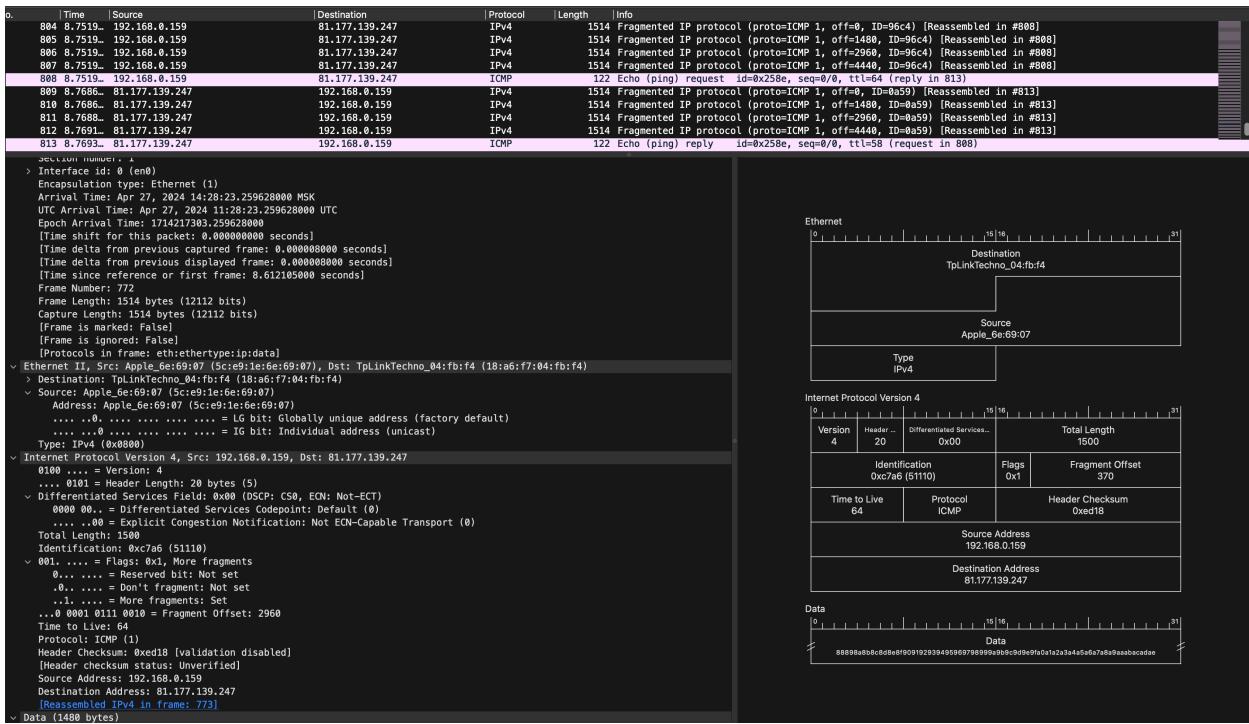


Рис. 1: Результат при передаче 6000 байт

Фрагменты состоят из IPv4 пакетов, которые содержат 1480 байт данных, 20 байт заголовок IP, 14 байт заголовок Ethernet. Всего размер Ethernet кадра равен 1514 байт.

Сам заголовок Ethernet I содержит MAC адреса - 12 байт и 2 байта, отвечающие за тип протокола выше (IPv4).

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

Флаг «More fragments» в пакете IPv4 сигнализирует о том, последуют ли за ним дополнительные фрагменты или это последняя часть фрагментированного пакета.

```

    ▼ 001. .... = Flags: 0x1, More fragments
      0.... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..1. .... = More fragments: Set
      ...0 0010 0010 1011 = Fragment Offset: 4440
  
```

- Чему равно количество фрагментов при передаче ping-пакетов? Так как размер кадра Ethernet ограничен 1518 байтами.

**1.4.109 basic frame:** A MAC frame that carries a Length/Type field with the Length or Type interpretation and has a maximum length of 1518 octets. The basic frame is not intended to allow inclusion of additional tags (i.e., untagged) or encapsulations required by higher layer protocols. (See IEEE Std 802.3, 3.2.7.)

Рис. 2: Из IEEE Standard for Ethernet (IEEE Std 802.3<sup>TM</sup>-2015)

Так в каждый пакет будет умещаться IPv4 максимум 1480 байт данных, потому что пакет IPv4 вкладывается в кадр Ethernet.

Количество фрагментов =  $\text{ceil}(\text{Длина сообщения (байт)} / 1480 \text{ байт (фрагмент)})$   
 $\text{ceil}()$  - округление вверх.

```
~ [5 IPv4 Fragments (6008 bytes): #804(1480), #805(1480), #806(1480), #807(1480), #808(88)]  
  [Frame: 804, payload: 0-1479 (1480 bytes)]  
  [Frame: 805, payload: 1480-2959 (1480 bytes)]  
  [Frame: 806, payload: 2960-4439 (1480 bytes)]  
  [Frame: 807, payload: 4440-5919 (1480 bytes)]  
  [Frame: 808, payload: 5920-6007 (88 bytes)]  
  [Fragment count: 5]  
  [Reassembled IPv4 length: 6008]
```

Рис. 3: Количество фрагментов при передачи сообщения 6000 байт

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

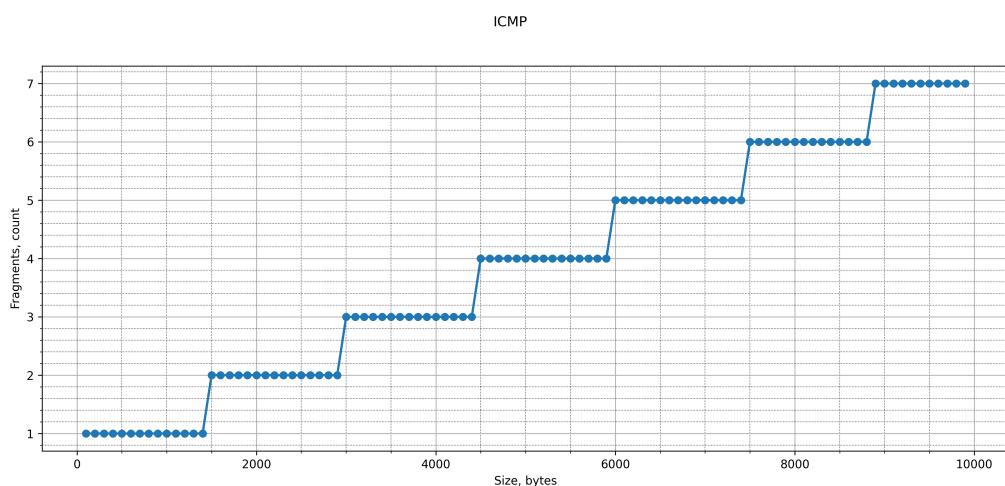


Рис. 4: Количество фрагментов от размера пакета

- Как изменить поле TTL с помощью утилиты ping?

```
-m ttl Set the IP Time To Live for outgoing packets. If not specified, the kernel uses the value of the net.inet.ip.ttl MIB variable.
```

Рис. 5: Изменить ttl ping в Unix

```
-T ttl Set the IP Time To Live for multicasted packets. This flag only applies if the ping destination is a multicast address.
```

Рис. 6: Изменить ttl ping для multicast в Unix

- Что содержится в поле данных ping-пакета?

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	36	37	(*)*+,-./	01234567			
38	39	3a	3b	3c	3d	3e	3f	40	41	42	43	44	45	46	47	89:;<=>?	@ABCDEFG			
48	49	4a	4b	4c	4d	4e	4f	50	51	52	53	54	55	56	57	HIJKLMNO	PQRSTUVWXYZ			
58	59	5a	5b	5c	5d	5e	5f	60	61	62	63	64	65	66	67	XYZ[\]^_`	abcdefg			
68	69	6a	6b	6c	6d	6e	6f	70	71	72	73	74	75	76	77	hijklmno	pqrstuvwxyz			
78	79	7a	7b	7c	7d	7e	7f	80	81	82	83	84	85	86	87	xyz{ }~.	.	.	.	.
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	36	37	(*)*+,-./	01234567			
38	39	3a	3b	3c	3d	3e	3f	40	41	42	43	44	45	46	47	89:;<=>?	@ABCDEFG			
48	49	4a	4b	4c	4d	4e	4f	50	51	52	53	54	55	56	57	HIJKLMNO	PQRSTUVWXYZ			
58	59	5a	5b	5c	5d	5e	5f	60	61	62	63	64	65	66	67	XYZ[\]^_`	abcdefg			
68	69	6a	6b	6c	6d	6e	6f	70	71	72	73	74	75	76	77	hijklmno	pqrstuvwxyz			
78	79	7a	7b	7c	7d	7e	7f	80	81	82	83	84	85	86	87	xyz{ }~.	.	.	.	.
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	.	.	.	.	.

Мусор в виде ASCII символов, если это так можно трактовать. Другими словами, однобайтовые слова, последовательно идущие друг за другом и увеличивающиеся на единицу.

## 2 Анализ трафика утилиты tracert (traceroute)

traceroute отправляет пакет с TTL=1 и смотрит адрес ответившего узла, дальше TTL=2, TTL=3 и так пока не достигнет цели. При TTL=0 пакет уничтожается, а отправителю отсылается сообщение Time Exceeded. Каждый раз отправляется по три пакета и для каждого из них измеряется время прохождения.

По умолчанию traceroute отправляет UDP дейтаграммы, но мы можем изменить на ICMP с помощью аргумента -I.

```

aleksejlapin@MacBook-Pro-Aleksej-2 scripts % traceroute -I alexeylapin.ru
traceroute to alexeylapin.ru (81.177.139.247), 64 hops max, 48 byte packets
 1  192.168.0.1 (192.168.0.1)  5.702 ms  1.601 ms  3.118 ms
 2  94.19.112.1.pool.sknt.ru (94.19.112.1)  2.139 ms  2.035 ms  1.768 ms
 3  k12-core (93.100.0.110)  1.908 ms  4.634 ms  2.261 ms
 4  * * *
 5  185.37.128.0 (185.37.128.0)  4.718 ms  2.731 ms  2.693 ms
 6  spb-r2-cr1.ae54-2153.rascom.as20764.net (80.64.103.130)  2.694 ms  4.274 ms  2.657 ms
 7  * * *
 8  ppp-176.211.83.73.nsk.rt.ru (176.211.83.73)  4.099 ms  4.387 ms  5.263 ms
 9  185.140.148.19 (185.140.148.19)  5.730 ms  5.561 ms  5.430 ms
10  * * *
11  89.191.239.165 (89.191.239.165)  14.359 ms  14.135 ms  14.360 ms
12  msk-bgw1-xe-0-3-0-0.rt-comm.ru (217.106.6.162)  20.328 ms  35.976 ms  13.998 ms
13  msk-bgw1-xe-0-3-0-0.rt-comm.ru (217.106.6.162)  14.056 ms  15.988 ms  13.866 ms
14  srv37-h-st.jino.ru (81.177.139.247)  14.086 ms  16.785 ms  14.090 ms

```

В соответствии с документацией, звездочки могут означать, что маршрутизатор в течение некоторого времени не отвечает, временно перегружен или некоторые маршрутизаторы специально не реагируют на данные сообщения, поскольку им запрещено это делать. Поэтому, даже если выставить большое значение TTL, мы все равно можем не достичь конечного узла в некоторых случаях.

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

В заголовке: 20 байт

В данных: 20 байт

```

.... 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: 48
    Identification: 0xc399 (50073)
    000. .... = Flags: 0x0
        0.... .... = Reserved bit: Not set
        .0... .... = Don't fragment: Not set
        ..0. .... = More fragments: Not set
        ...0 0000 0000 0000 = Fragment Offset: 0
    > Time to Live: 2
        Protocol: ICMP (1)
        Header Checksum: 0x5644 [validation disabled]
        [Header checksum status: Unverified]
        Source Address: 192.168.0.159
        Destination Address: 81.177.139.247
    └ Internet Control Message Protocol
        Type: 8 (Echo (ping) request)
        Code: 0
        Checksum: 0x3466 [correct]
        [Checksum Status: Good]
        Identifier (BE): 50069 (0xc395)
        Identifier (LE): 38339 (0x95c3)
        Sequence Number (BE): 4 (0x0004)
        Sequence Number (LE): 1024 (0x0400)
    > [No response seen]
    > Data (20 bytes)

```

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

Для определения пути, traceroute отправляет пакеты с TTL, начиная от 1 и увеличивая на 1 через каждые 1-3 пакета. Когда TTL в маршрутизаторе становится равным 0, тот посыпает ICMP сообщение с информацией о том, что TTL превышен. Таким образом, по умолчанию сначала отправляется пакет с TTL, равным 1, затем с TTL, равным 2, и так далее, пока не достигнем конечного узла.

165 9.074176	192.168.0.1	192.168.0.159	ICMP	90 Time-to-live exceeded (Time to live exceeded in transit)
166 9.074820	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=2/512, ttl=1 (no response found!)
167 9.076329	192.168.0.1	192.168.0.159	ICMP	90 Time-to-live exceeded (Time to live exceeded in transit)
168 9.076408	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=3/768, ttl=1 (no response found!)
169 9.079467	192.168.0.1	192.168.0.159	ICMP	90 Time-to-live exceeded (Time to live exceeded in transit)
170 9.079540	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=4/1024, ttl=2 (no response found!)
171 9.081687	94.19.112.1	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
172 9.082015	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=5/1280, ttl=2 (no response found!)
173 9.083954	94.19.112.1	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
174 9.084050	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=6/1536, ttl=2 (no response found!)
175 9.085748	94.19.112.1	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
176 9.085825	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=7/1792, ttl=3 (no response found!)
177 9.087653	93.180.0.118	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
178 9.087653	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=8/2048, ttl=3 (no response found!)
179 9.092553	93.180.0.118	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
180 9.092570	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=9/2304, ttl=3 (no response found!)
181 9.095018	93.180.0.118	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
182 9.095113	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=10/2560, ttl=4 (no response found!)
250 14.180394	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=11/2816, ttl=4 (no response found!)
317 19.191748	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=12/3072, ttl=4 (no response found!)
379 24.160681	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=13/3328, ttl=5 (no response found!)
380 24.118030	185.37.128.0	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
383 24.118644	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=14/3584, ttl=5 (no response found!)
384 24.121217	185.37.128.0	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
385 24.121351	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=15/3840, ttl=5 (no response found!)
386 24.123924	185.37.128.0	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
387 24.124063	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=16/4096, ttl=6 (no response found!)
388 24.126565	88.64.183.130	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
389 24.127688	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=17/4352, ttl=6 (no response found!)
390 24.131786	88.64.183.130	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
391 24.131957	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=18/4608, ttl=6 (no response found!)
392 24.134481	88.64.183.130	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
393 24.134618	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=19/4864, ttl=7 (no response found!)
1585 29.136116	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=20/5120, ttl=7 (no response found!)
6185 34.141257	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=21/5360, ttl=7 (no response found!)
6304 39.1433021	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=22/5632, ttl=8 (no response found!)
6305 39.146732	176.211.83.13	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
6306 39.148113	192.168.0.159	81.177.139.247	ICMP	62 Echo (ping) request: id=0xc395, seq=23/5888, ttl=8 (no response found!)
6307 39.152088	176.211.83.13	192.168.0.159	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

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

В отличие от пакетов, генерируемых утилитой ping, пакеты, генерируемые утилитой traceroute, в поле данных содержат нули.

00 30 c3 a8 00 00 07 01 51 35 c0 a8 00 9f 51 b1	0 . . . . . Q5 . . . Q
8b f7 08 00 34 57 c3 95 00 13 00 00 00 00 00 00 00	4W . . . . .
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....

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

Различные значения в поле Type. Error возвращает ошибочные ответы, reply – обычный ответ на запрос от сервера. Оба типа необходимы, чтобы различать причину истечения TTL: если хост успешно достигнут, приходит reply, а если произошла ошибка – error.

Internet Control Message Protocol
Type: 11 (Time-to-live exceeded)
Code: 0 (Time to live exceeded in transit)
Checksum: 0xf4ff [correct]
[Checksum Status: Good]
Unused: 00000000

Рис. 7: ICMP Error

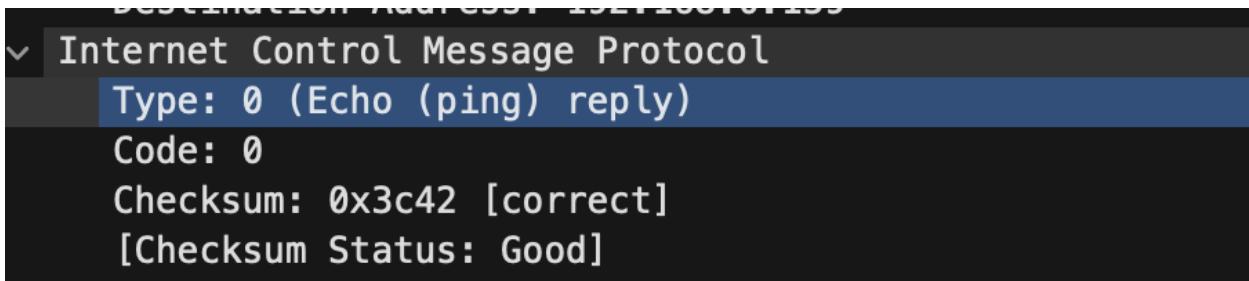


Рис. 8: ICMP Reply

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

Из документации WINDOWS:

Ключ -d предотвращает попытки команды tracert преобразовывать IP-адреса промежуточных маршрутизаторов в имена хостов. Таким образом, в выводе команды не будут отображаться имена хостов, через которые проходит IP-пакет.

Ключ -d в версии Unix, позволяет включить отладку на уровне сокета

### 3 Анализ HTTP-трафика

Так как на выбраном сайте по варианту весь контент динамический, а значит *not modified* там быть не может.

Возьмем другой сайт: <http://math.gsu.by/wp-content/uploads/courses/networks/r5.1.html>

4554 4.893438	192.168.0.159	37.17.74.99	HTTP	552 GET /wp-content/uploads/courses/networks/r5.1.html HTTP/1.1
4608 4.138014	192.168.0.159	37.17.74.99	HTTP	487 GET /wp-content/uploads/courses/networks/files/styles.css HTTP/1.1
4613 4.146095	37.17.74.99	192.168.0.159	HTTP	1844 HTTP/1.1 200 OK (text/html)
4615 4.146383	192.168.0.159	37.17.74.99	HTTP	530 GET /wp-content/uploads/courses/networks/files/re.gif HTTP/1.1
4619 4.153185	192.168.0.159	37.17.74.99	HTTP	531 GET /wp-content/uploads/courses/networks/files/news.gif HTTP/1.1
4620 4.154272	192.168.0.159	37.17.74.99	HTTP	530 HTTP/1.1 200 OK (text/css)
4621 4.154936	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/files/r1.gif HTTP/1.1
4629 4.169650	37.17.74.99	192.168.0.159	HTTP	1231 HTTP/1.1 200 OK (GIF89a)
4631 4.170010	192.168.0.159	37.17.74.99	HTTP	528 GET /wp-content/uploads/courses/networks/pic/h5d1.jpg HTTP/1.1
4641 4.174134	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d2.jpg HTTP/1.1
4642 4.174180	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5t1.jpg HTTP/1.1
4645 4.174329	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5t2.jpg HTTP/1.1
4647 4.188795	37.17.74.99	192.168.0.159	HTTP	59 HTTP/1.1 200 OK (GIF89a)
4652 4.181393	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5t3.jpg HTTP/1.1
4661 4.183827	37.17.74.99	192.168.0.159	HTTP	59 HTTP/1.1 200 OK (GIF89a)
4663 4.183873	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d3.jpg HTTP/1.1
4689 4.215522	37.17.74.99	192.168.0.159	HTTP	59 HTTP/1.1 200 OK (JPEG JFIF image)
4691 4.215765	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d4.jpg HTTP/1.1
4732 4.223235	37.17.74.99	192.168.0.159	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d5.jpg HTTP/1.1
4733 4.223441	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d6.jpg HTTP/1.1
4752 4.228185	37.17.74.99	192.168.0.159	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d7.jpg HTTP/1.1
4754 4.228444	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d8.jpg HTTP/1.1
4788 4.235969	37.17.74.99	192.168.0.159	HTTP	531 GET /wp-content/uploads/courses/networks/files/back.gif HTTP/1.1
4791 4.235364	192.168.0.159	37.17.74.99	HTTP	59 HTTP/1.1 200 OK (JPEG JFIF image)
4837 4.248968	37.17.74.99	192.168.0.159	HTTP	534 GET /wp-content/uploads/courses/networks/files/forward.gif HTTP/1.1
4841 4.249263	192.168.0.159	37.17.74.99	HTTP	495 HTTP/1.1 200 OK (JPEG JFIF image)
4854 4.252355	37.17.74.99	192.168.0.159	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d7.jpg HTTP/1.1
4858 4.252694	192.168.0.159	37.17.74.99	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d8.jpg HTTP/1.1
4871 4.255264	37.17.74.99	192.168.0.159	HTTP	531 GET /wp-content/uploads/courses/networks/files/back.gif HTTP/1.1
4875 4.255546	192.168.0.159	37.17.74.99	HTTP	59 HTTP/1.1 200 OK (JPEG JFIF image)
4889 4.258627	37.17.74.99	192.168.0.159	HTTP	534 GET /wp-content/uploads/courses/networks/files/forward.gif HTTP/1.1
4919 4.265324	37.17.74.99	192.168.0.159	HTTP	495 HTTP/1.1 200 OK (JPEG JFIF image)
4950 4.279891	37.17.74.99	192.168.0.159	HTTP	529 GET /wp-content/uploads/courses/networks/pic/h5d7.jpg HTTP/1.1
4951 4.280393	37.17.74.99	192.168.0.159	HTTP	1289 HTTP/1.1 200 OK (GIF89a)
4964 4.286580	37.17.74.99	192.168.0.159	HTTP	1293 HTTP/1.1 200 OK (GIF89a)
4975 4.289851	37.17.74.99	192.168.0.159	HTTP	59 HTTP/1.1 200 OK (JPEG JFIF image)
5171 9.318831	192.168.0.159	37.17.74.99	HTTP	134 HTTP/1.1 200 OK (JPEG JFIF image)
5173 9.353655	37.17.74.99	192.168.0.159	HTTP	662 GET /wp-content/uploads/courses/networks/r5.1.html HTTP/1.1
5425 18.542795	192.168.0.159	37.17.74.99	HTTP	249 HTTP/1.1 304 Not Modified
5430 18.575354	37.17.74.99	192.168.0.159	HTTP	662 GET /wp-content/uploads/courses/networks/r5.1.html HTTP/1.1
				249 HTTP/1.1 304 Not Modified

Как можем увидеть, протокол HTTP является прикладным протоколом, и поэтому работает поверх протокола TCP.

ICP payload (496 bytes)
└ Hypertext Transfer Protocol
└ GET /wp-content/uploads/courses/networks/r5.1.html HTTP/1.1\r\n
└ [Expert Info (Chat/Sequence): GET /wp-content/uploads/courses/networks/r5.1.html HTTP/1.1\r\n]
Request Method: GET
Request URI: /wp-content/uploads/courses/networks/r5.1.html
Request Version: HTTP/1.1

Сообщение протокола представляется в виде:

- <Тип запроса (GET/POST) > <путь относительно корня>
- <Протокол/версия>
- Затем с новой строки (переноса) идут значения вида ключ:значение
- После идет тело запроса, может отсутствовать

```

▼ Hypertext Transfer Protocol, has 5 chunks (including last chunk)
  ▼ HTTP/1.1 200 OK\r\n
    > [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
  
```

Ответ выглядит в виде:

- <Протокол/версия> <Статус> <перенос>
- Заголовок вида ключ:значение
- Тело запроса

```

▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 304 Not Modified\r\n
    > [Expert Info (Chat/Sequence): HTTP/1.1 304 Not Modified\r\n]
      Response Version: HTTP/1.1
      Status Code: 304
      [Status Code Description: Not Modified]
      Response Phrase: Not Modified
  
```

При вторичном посещении сайта получаем заголовок «HTTP/1.1 304 Not Modified». Он означает, что не нужно отправлять тело вместе с ним (таким образом экономится пропускная способность). Данные о сайте остались в кэше браузера.

## 4 Анализ DNS-трафика

Очищаем кеш DNS на macOS командой:

`sudo dscacheutil -flushcache`

3258 34.567844	192.168.0.159	192.168.0.1	DNS	74 Standard query 0x38a3 A alexeylapin.ru
3259 34.567893	192.168.0.159	192.168.0.1	DNS	74 Standard query 0x0d2f HTTPS alexeylapin.ru

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

Для того, чтобы получить IP адрес сайта отправляется запрос на DNS сервер, который возвращает IP адрес уже сайта.

- Какие бывают типы DNS-запросов?

- Прямой — запрос на преобразование имени (символьного адреса) хоста в его IP-адрес.

- Обратный - запрос на преобразование адреса хоста в его имя.
- Рекурсивный - DNS-сервер опрашивает серверы (в порядке убывания уровня зон в имени), пока не найдёт ответ или не обнаружит, что домен не существует.
- Итеративный - Данные запросы передаются между DNS-серверами, когда один из них не имеет соответствующих записей. Таким образом, инициатор запроса будет контактировать с сервером, который имеет нужную запись.

Типы записей (Ресурсные записи):

1. A — получение IPv4
2. AAAA - получение IPv6
3. CNAME - получение канонического имени
4. MX - получение информации о почтовых серверах, ответственных за обработку почту для данного домена
5. NS (Name Server) - вернуть список DNS серверов, ответственных за данный домен
6. TXT — любая текстовая информация о домене
7. SPF — список серверов, которым позволено отправлять письма от имени указанного домена
8. SOA — исходная запись зоны, в которой указаны сведения о сервере.
9. PTR - обратная DNS-запись или запись указателя связывает IP-адрес хоста с его каноническим именем.

Подробнее: Что такое ресурсные записи DNS - Reg.ru

- **В какой ситуации нужно выполнять независимые DNS-запросы для получения содержащихся на сайте изображений?**

- Если изображения хранятся на отдельном сервере или поддомене
- Используется CDN (Content Delivery Network), где ресурсы могут храниться на разных серверах

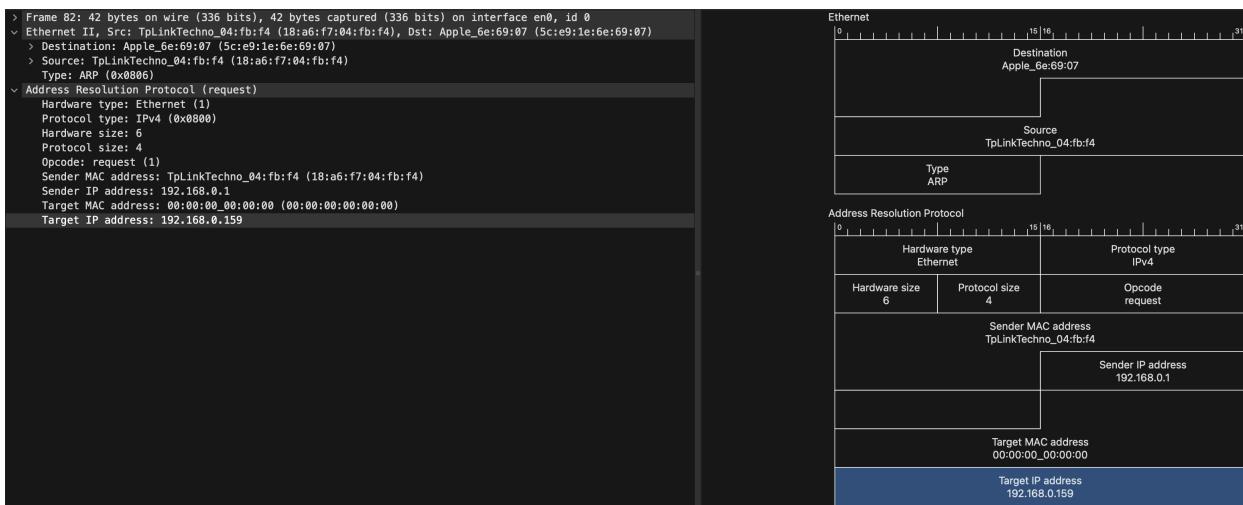
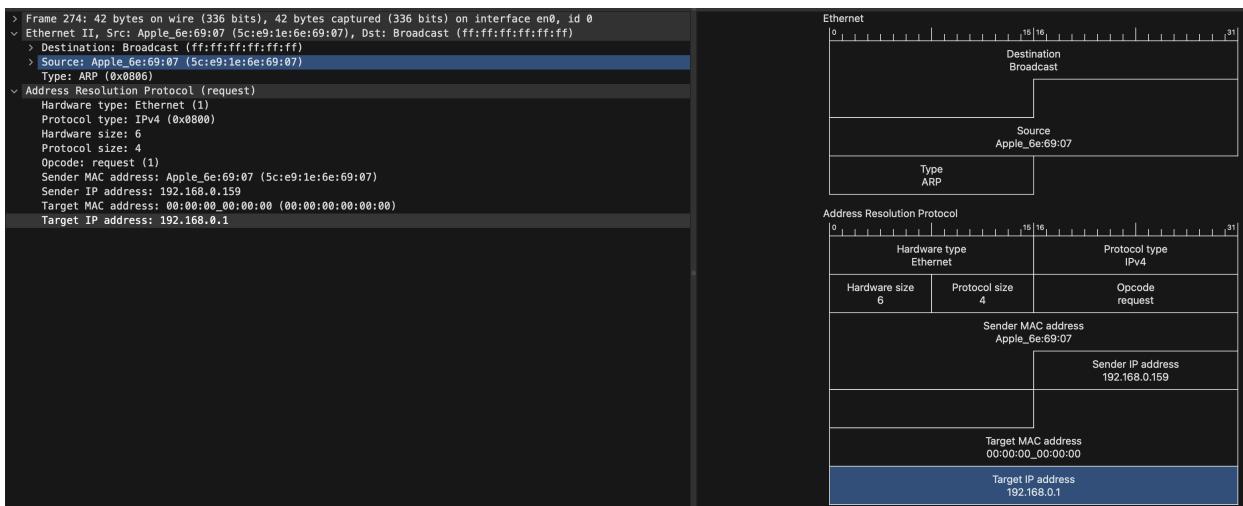
## 5 Анализ ARP-трафика

Очищаем таблицу MAC адресов:

```
aleksejlapin@MacBook-Pro-Aleksej-2 scripts % sudo arp -a -d
Password:
169.254.169.254 (169.254.169.254) deleted
192.168.0.1 (192.168.0.1) deleted
192.168.0.153 (192.168.0.153) deleted
192.168.0.156 (192.168.0.156) deleted
192.168.0.198 (192.168.0.198) deleted
224.0.0.251 (224.0.0.251) deleted
239.255.255.250 (239.255.255.250) deleted
```

Как можно заметить ниже ARP запросы посылаются как от маршрутизатора (192.168.0.1), так и от компьютера (192.168.0.159). Broadcast означает, что данный запрос предназначен всем узлам в данной локальной сети.

82 3.038903	TpLinkTechno_04:fb:f4	Apple_6e:69:07	ARP	42 Who has 192.168.0.159? Tell 192.168.0.1
83 3.039035	Apple_6e:69:07	TpLinkTechno_04:fb:f4	ARP	42 192.168.0.159 is at 5:c:e9:1e:6e:69:07
274 16.510091	Apple_6e:69:07	Broadcast	ARP	42 Who has 192.168.0.17 Tell 192.168.0.159
275 16.517191	TpLinkTechno_04:fb:f4	Apple_6e:69:07	ARP	42 192.168.0.1 is at 18:a6:f7:04:fb:f4
301 18.217563	Apple_6e:69:07	Broadcast	ARP	42 Who has 192.168.0.153 Tell 192.168.0.159
302 18.247814	4:a:66:a3:93:6d:cd	Apple_6e:69:07	ARP	42 192.168.0.153 is at 4:a:66:a3:93:6d:cd
305 19.657408	Apple_6e:69:07	Broadcast	ARP	42 Who has 192.168.0.17 Tell 192.168.0.159
306 19.661640	TpLinkTechno_04:fb:f4	Apple_6e:69:07	ARP	42 192.168.0.1 is at 18:a6:f7:04:fb:f4
313 20.263949	Apple_6e:69:07	Broadcast	ARP	42 Who has 192.168.0.17 Tell 192.168.0.159
314 20.265620	TpLinkTechno_04:fb:f4	Apple_6e:69:07	ARP	42 192.168.0.1 is at 18:a6:f7:04:fb:f4
458 30.371693	TpLinkTechno_04:fb:f4	Apple_6e:69:07	ARP	42 Who has 192.168.0.159? Tell 192.168.0.1
459 30.371798	Apple_6e:69:07	TpLinkTechno_04:fb:f4	ARP	42 192.168.0.159 is at 5:c:e9:1e:6e:69:07
521 33.854739	Apple_6e:69:07	Broadcast	ARP	42 Who has 169.254.169.254? Tell 192.168.0.159
697 34.855976	Apple_6e:69:07	Broadcast	ARP	42 Who has 169.254.169.254? Tell 192.168.0.159
705 35.857695	Apple_6e:69:07	Broadcast	ARP	42 Who has 169.254.169.254? Tell 192.168.0.159



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

- TpLinkTechno\_04:fb:f4 (18:a6:f7:04:fb:f4) – маршрутизатор
- 00:00:00:00:00:00 – broadcast-адрес
- Apple\_6e:69:07 (5:c:e9:1e:6e:69:07) – компьютера

```
Protocol size: 4
Opcode: request (1)
Sender MAC address: TpLinkTechno_04:fb:f4 (18:a6:f7:04:fb:f4)
Sender IP address: 192.168.0.1
Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
Target IP address: 192.168.0.159
```

```
Opcode: reply (2)
Sender MAC address: Apple_6e:69:07 (5c:e9:1e:6e:69:07)
Sender IP address: 192.168.0.159
Target MAC address: TpLinkTechno_04:fb:f4 (18:a6:f7:04:fb:f4)
Target IP address: 192.168.0.1
```

При запросе присутствует MAC адрес отправителя и Target это broadcast-адресс

При ответе присутствуют MAC адрес отправителя и Target того устройства, которое делало запрос.

```
▼ Ethernet II, Src: Apple_6e:69:07 (5c:e9:1e:6e:69:07), Dst: TpLinkTechno_04:fb:f4 (18:a6:f7:04:fb:f4)
  ▼ Destination: TpLinkTechno_04:fb:f4 (18:a6:f7:04:fb:f4)
    Address: TpLinkTechno_04:fb:f4 (18:a6:f7:04:fb:f4)
    .... 0. .... .... .... .... = LG bit: Globally unique address (factory default)
    .... 0. .... .... .... .... = IG bit: Individual address (unicast)
  ▼ Source: Apple_6e:69:07 (5c:e9:1e:6e:69:07)
    Address: Apple_6e:69:07 (5c:e9:1e:6e:69:07)
    .... 0. .... .... .... .... = LG bit: Globally unique address (factory default)
    .... 0. .... .... .... .... = IG bit: Individual address (unicast)
```

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

В HTTP-пакетах содержатся те же MAC-адреса, что и в ARP-запросе. Они служат для идентификации отправителя и получателя HTTP-пакета.

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

Чтобы узел-получатель мог добавить информацию об узле-отправителе в свою таблицу ARP. С помощью этой таблицы получатель сможет сопоставить IP с MAC-адресом при отправке следующих пакетов, без необходимости снова посыпать ARP-запрос.

## 6 Анализ трафика утилиты nslookup

NSLOOKUP — это утилита, которая позволяет через командную строку узнать содержимое DNS. Утилита поможет:

- узнать IP-адрес,
- узнать A, NS, SOA, MX-записи для домена.

С помощью А-записи домен прикрепляется к IP-адресу. Таким образом, А-запись позволяет найти IP.

```
aleksejlapin@MacBook-Pro-Aleksej-2 scripts % nslookup alexeylapin.ru
Server:      192.168.0.1
Address:     192.168.0.1#53

Non-authoritative answer:
Name:   alexeylapin.ru
Address: 81.177.139.247
```

Утилита NSLOOKUP позволяет определить, какие NS-серверы использует сайт.

```
aleksejlapin@MacBook-Pro-Aleksej-2 scripts % nslookup -type=NS alexeylapin.ru
Server:      192.168.0.1
Address:     192.168.0.1#53

Non-authoritative answer:
alexeylapin.ru nameserver = ns2.jino.ru.
alexeylapin.ru nameserver = ns4.jino.ru.
alexeylapin.ru nameserver = ns1.jino.ru.
alexeylapin.ru nameserver = ns3.jino.ru.

Authoritative answers can be found from:
ns1.jino.ru      internet address = 217.107.34.200
ns2.jino.ru      internet address = 195.161.62.86
ns3.jino.ru      internet address = 217.107.219.170
ns4.jino.ru      internet address = 81.177.139.205
ns2.jino.ru      has AAAA address 2001:1bb0:e000:1e::917
ns4.jino.ru      has AAAA address 2001:1bb0:e000:1e::1cd
```

В ответе на любую команду утилиты показывает, с какого сервера была получена информация. Ответ приходит от серверов двух типов:

- authoritative
- non-authoritative

Authoritative answer (авторитетный ответ) – это ответ, который получен от основного (официального) сервера. Non-authoritative answer (неавторитетный ответ) – это ответ от промежуточного сервера.

39 1.823779	192.168.0.159	192.168.0.1	DNS	74 Standard query 0x1617 A alexeylapin.ru
40 1.826384	192.168.0.1	192.168.0.159	DNS	98 Standard query response 0x1617 A alexeylapin.ru A 81.177.139.247
57 3.809372	192.168.0.159	192.168.0.1	DNS	74 Standard query 0x9f9f NS alexeylapin.ru
62 3.913361	192.168.0.1	192.168.0.159	DNS	273 Standard query response 0x9f9f NS alexeylapin.ru NS ns1.jino.ru NS ns3.jino.ru NS ns2.jino.ru

- Чем различается трасса трафика в п.2 и п.4, указанных выше?
  - При запуске в п.2 утилиты ищет IP-адрес хоста (запись типа A (IPv4) или AAAA (IPv6)).
  - При запуске в п.4 утилиты ищет Name Server для запрашиваемого хоста.

NS-запись (Authoritative name server) указывает на DNS-серверы, которые отвечают за хранение остальных ресурсных записей домена. Количество NS записей должно строго соответствовать количеству всех обслуживающих его серверов. Критически важна для работы службы DNS.

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

```
▼ Queries
  > alexeylapin.ru: type A, class IN
▼ Answers
  ▼ alexeylapin.ru: type A, class IN, addr 81.177.139.247
    Name: alexeylapin.ru
    Type: A (1) (Host Address)
    Class: IN (0x0001)
    Time to live: 2457 (40 minutes, 57 seconds)
    Data length: 4
    Address: 81.177.139.247
```

Рис. 9: Answer для типа A

```
▼ Queries
  > alexeylapin.ru: type NS, class IN
▼ Answers
  > alexeylapin.ru: type NS, class IN, ns ns1.jino.ru
  > alexeylapin.ru: type NS, class IN, ns ns3.jino.ru
  > alexeylapin.ru: type NS, class IN, ns ns2.jino.ru
  > alexeylapin.ru: type NS, class IN, ns ns4.jino.ru
```

Рис. 10: Answer для типа NS

```
▼ Answers
  ▼ alexeylapin.ru: type NS, class IN, ns ns1.jino.ru
    Name: alexeylapin.ru
    Type: NS (2) (authoritative Name Server)
    Class: IN (0x0001)
    Time to live: 2455 (40 minutes, 55 seconds)
    Data length: 13
    Name Server: ns1.jino.ru
```

Рис. 11: Answer для типа NS

- NAME - имя хоста.
- TYPE - тип ресурсной записи. Определяет формат и назначение данной ресурсной записи.
- CLASS — класс ресурсной записи. Обычно IN для Internet (Код 0x0001)

- TTL - (Time To Live) - допустимое время хранения данной ресурсной записи в кэше неответственного DNS-сервера.
- Data length - длина поля данных.
- Name Server - dns сервер с которого получена информация
- Address - запрашиваемый IP адресс

Данные запрашиваемого типа DNS-записи: для A - IPv4-адрес, для NS - список authoritative Name Server.

- Каковы имена серверов, возвращающих авторитетивный (authoritative) отклик?

```

    < Queries
      > alexeylapin.ru: type NS, class IN
    < Answers
      > alexeylapin.ru: type NS, class IN, ns ns1.jino.ru
      > alexeylapin.ru: type NS, class IN, ns ns3.jino.ru
      > alexeylapin.ru: type NS, class IN, ns ns2.jino.ru
      > alexeylapin.ru: type NS, class IN, ns ns4.jino.ru

```

Рис. 12: Имена авторитетивных серверов

DNS-серверы, которые отвечают за хранение остальных ресурсных записей домена. То есть они являются ответственными за зону, в которой описана информация, необходимая DNS-клиенту.

## 7 Анализ FTP-трафика

```

aleksejlapin@MacBook-Pro-Aleksej-2 scripts % wget ftp://gitlab.se.ifmo.ru/
--2024-04-28 14:49:52--  ftp://gitlab.se.ifmo.ru/
                      => '.listing'
Resolving gitlab.se.ifmo.ru (gitlab.se.ifmo.ru)... 77.234.196.4
Connecting to gitlab.se.ifmo.ru (gitlab.se.ifmo.ru)|77.234.196.4|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.  ==> PWD ... done.
==> TYPE I ... done.  ==> CWD not needed.
==> PASV ... done.  ==> LIST ... done.

.listing          [ <=> ]  242 --.-KB/s  in 0s

2024-04-28 14:49:54 (10.5 MB/s) - '.listing' saved [242]

Removed '.listing'.
Wrote HTML-ized index to 'index.html' [388].

```

Рис. 13: Получим, что-нибудь gitlab

27	2.781885	77.234.196.4	192.168.0.159	FTP	121 Response: 220 ProFTPD 1.3.5a Server (FTP Server) [77.234.196.4]
29	2.782132	192.168.0.159	77.234.196.4	FTP	82 Request: USER anonymous
30	2.783968	77.234.196.4	192.168.0.159	FTP	141 Response: 331 Anonymous login ok, send your complete email address as your password
32	2.785835	192.168.0.159	77.234.196.4	FTP	79 Response: 110 55 -wget@
33	2.794117	77.234.196.4	192.168.0.159	FTP	116 Response: 230 Anonymous access granted, restrictions apply
35	2.794388	192.168.0.159	77.234.196.4	FTP	72 Request: SYST
36	2.814515	77.234.196.4	192.168.0.159	FTP	88 Response: 215 UNIX Type: L8
38	2.814685	192.168.0.159	77.234.196.4	FTP	71 Request: PWD
39	2.834777	77.234.196.4	192.168.0.159	FTP	100 Response: 257 "/" is the current directory
41	2.835219	192.168.0.159	77.234.196.4	FTP	74 Request: TYPE I
42	2.854357	77.234.196.4	192.168.0.159	FTP	88 Response: 200 Type set to I
44	2.854937	192.168.0.159	77.234.196.4	FTP	72 Request: PASV
45	2.862269	77.234.196.4	192.168.0.159	FTP	117 Response: 227 Entering Passive Mode (77.234.196.4,200,223).
51	2.884965	192.168.0.159	77.234.196.4	FTP	75 Request: LIST -a
52	2.890142	77.234.196.4	192.168.0.159	FTP	121 Response: 150 Opening BINARY mode data connection for file list
53	2.890143	77.234.196.4	192.168.0.159	FTP-DATA	308 FTP Data: 242 bytes (PASV) (LIST -a)
58	2.914532	77.234.196.4	192.168.0.159	FTP	92 Response: 226 Transfer complete

- Сколько байт данных содержится в пакете FTP-DATA?

```
TCP payload (242 bytes)
FTP Data (242 bytes data)
[Setup frame: 45]
[Setup method: PASV]
[Command: LIST -a]
[Command frame: 51]
[Current working directory: /]
Line-based text data (4 lines)
drwxr-xr-x 4 root      wheel      4 Dec  5 2016 .
drwxr-xr-x 4 root      wheel      4 Dec  5 2016 ..
d-wxrwxt-- 2 uploader   ftp        2 Mar 23 2020 incoming
drwxr-xr-x 4 root      wheel      4 May 16 2018 pub

FTP Data: Protocol
```

242 байта

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

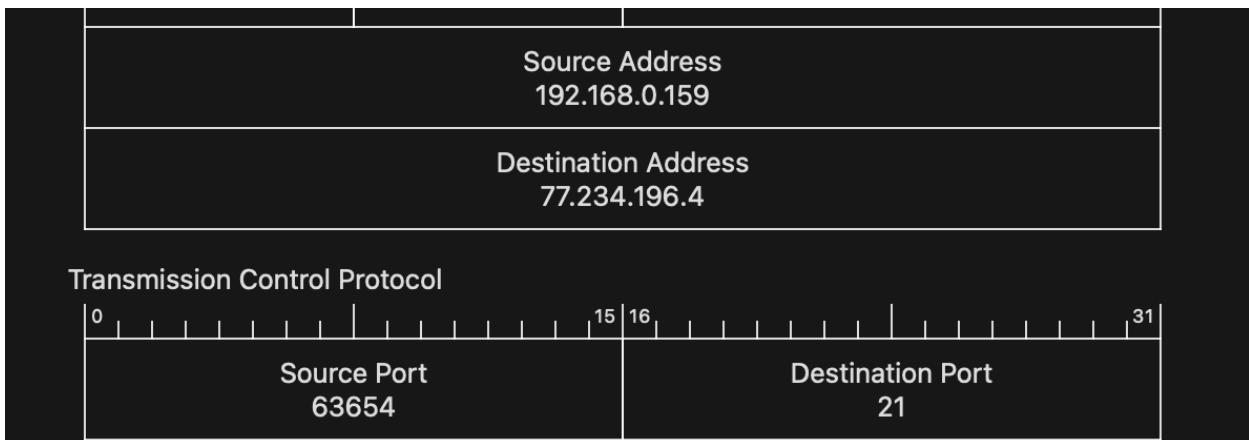


Рис. 14: FTP

Для протокола FTP стандартный порт для управления - 21.

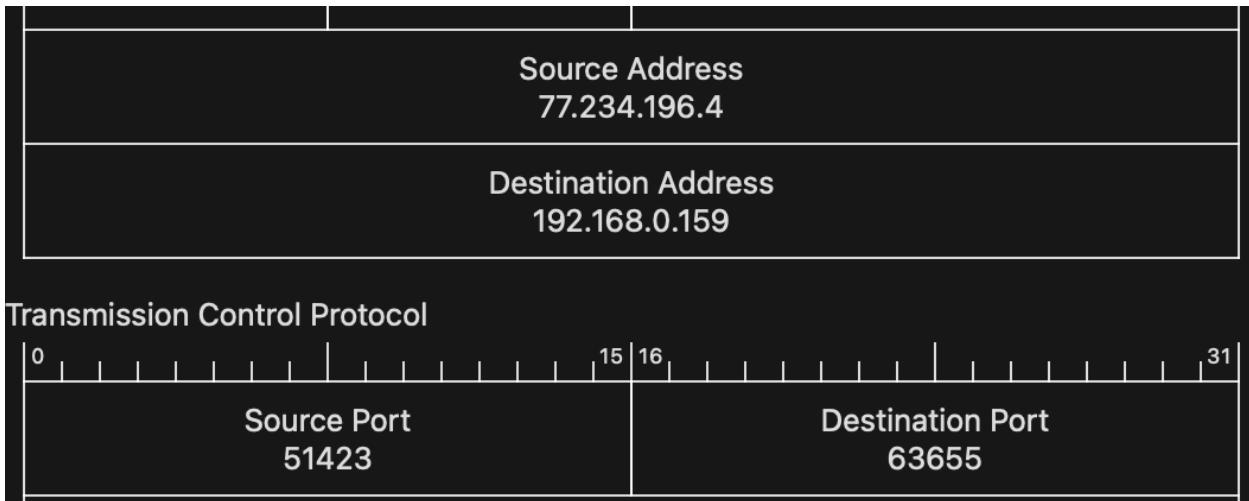


Рис. 15: FTP

Для FTP-DATA может быть выбран любой порт, либо по умолчанию используется 20.

- **Чем отличаются пакеты FTP от FTP-DATA?**

FTP используется для управления каналом: аутентификации, навигации и общего управления передачей файлов.

FTP-DATA - отвечает за передачу фактических данных файла между клиентом и сервером.

Это можно увидеть по описанию сообщений в Wireshark.

## 8 Анализ DHCP-трафика

```
C:\Users\aleksejlapin>ipconfig /release  
Windows IP Configuration  
  
Ethernet adapter Ethernet:  
  
Connection-specific DNS Suffix . . . :  
IPv6 Address . . . . . : fdb2:2c26:f4e4:0:41bf:e07c:7646:da06  
Temporary IPv6 Address . . . . . : fdb2:2c26:f4e4:0:b0a7:d42b:7fe3:9012  
Link-local IPv6 Address . . . . . : fe80::b9f1:58d4:2319:9fdc%7  
Default Gateway . . . . . :  
  
C:\Users\aleksejlapin>ipconfig /renew  
Windows IP Configuration  
  
Ethernet adapter Ethernet:  
  
Connection-specific DNS Suffix . . . : localdomain  
IPv6 Address . . . . . : fdb2:2c26:f4e4:0:41bf:e07c:7646:da06  
Temporary IPv6 Address . . . . . : fdb2:2c26:f4e4:0:b0a7:d42b:7fe3:9012  
Link-local IPv6 Address . . . . . : fe80::b9f1:58d4:2319:9fdc%7  
IPv4 Address . . . . . : 10.211.55.3  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : 10.211.55.1
```

No.	Time	Source	Destination	Protocol	Length	Info
6	2.771418	10.211.55.3	10.211.55.1	DHCP	342	DHCP Release - Transaction ID 0x8e4a6f99
22	7.259687	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x99dd8368
23	7.260070	10.211.55.1	10.211.55.3	DHCP	329	DHCP Offer - Transaction ID 0x99dd8368
24	7.260639	0.0.0.0	255.255.255.255	DHCP	348	DHCP Request - Transaction ID 0x99dd8368
25	7.264551	10.211.55.1	10.211.55.3	DHCP	329	DHCP ACK - Transaction ID 0x99dd8368

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

Оба запроса выполняются клиентом, DHCP Discover ищет DHCP-сервер в своей канальной среде, а DHCP Request принимает предлагаемый адрес и уведомляет DHCP-сервер об этом.

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

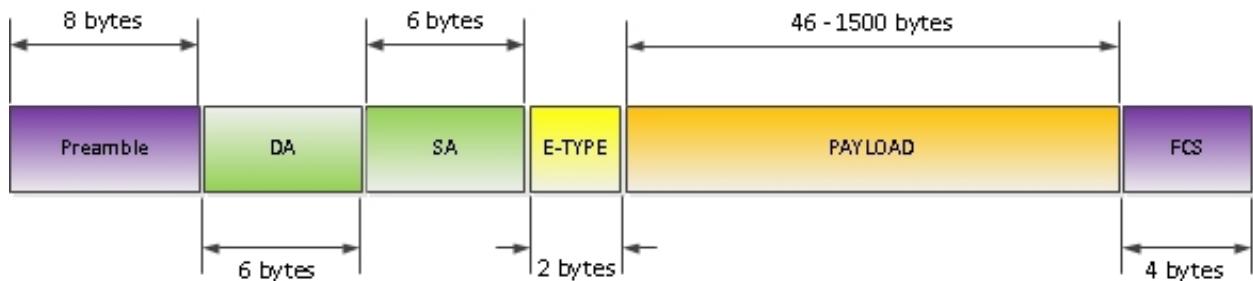
1. DHCP-discover - инициируется клиентом. MAC адрес отправителя — компьютер, IP адрес - 0.0.0.0 (не задан); MAC адрес получателя — broadcast, IP адрес — 255.255.255.255 (limited broadcast — все устройства в локальной сети).
2. DHCP-offer - инициируется сервером. MAC адрес и IP адрес отправителя - сервера; IP адрес получателя — предложенный, MAC адрес получателя — тот кто запросил
3. DHCP-request - инициируется клиентом. MAC адрес отправителя — компьютер, IP адрес - 0.0.0.0 (не задан); MAC адрес получателя — broadcast, IP адрес — 255.255.255.255 (limited broadcast)

4. DHCP-ack - инициируется сервером. MAC адрес и IP адрес отправителя — сервера; MAC адрес получателя — компьютер, IP адрес — запрошенный в DHCP-request.

- Каков IP-адрес DHCP-сервера? 10.211.55.1
- Что произойдёт, если очистить использованный фильтр «bootp»? Поскольку это был единственный фильтр, то будут просто отображаться все запросы-ответы.

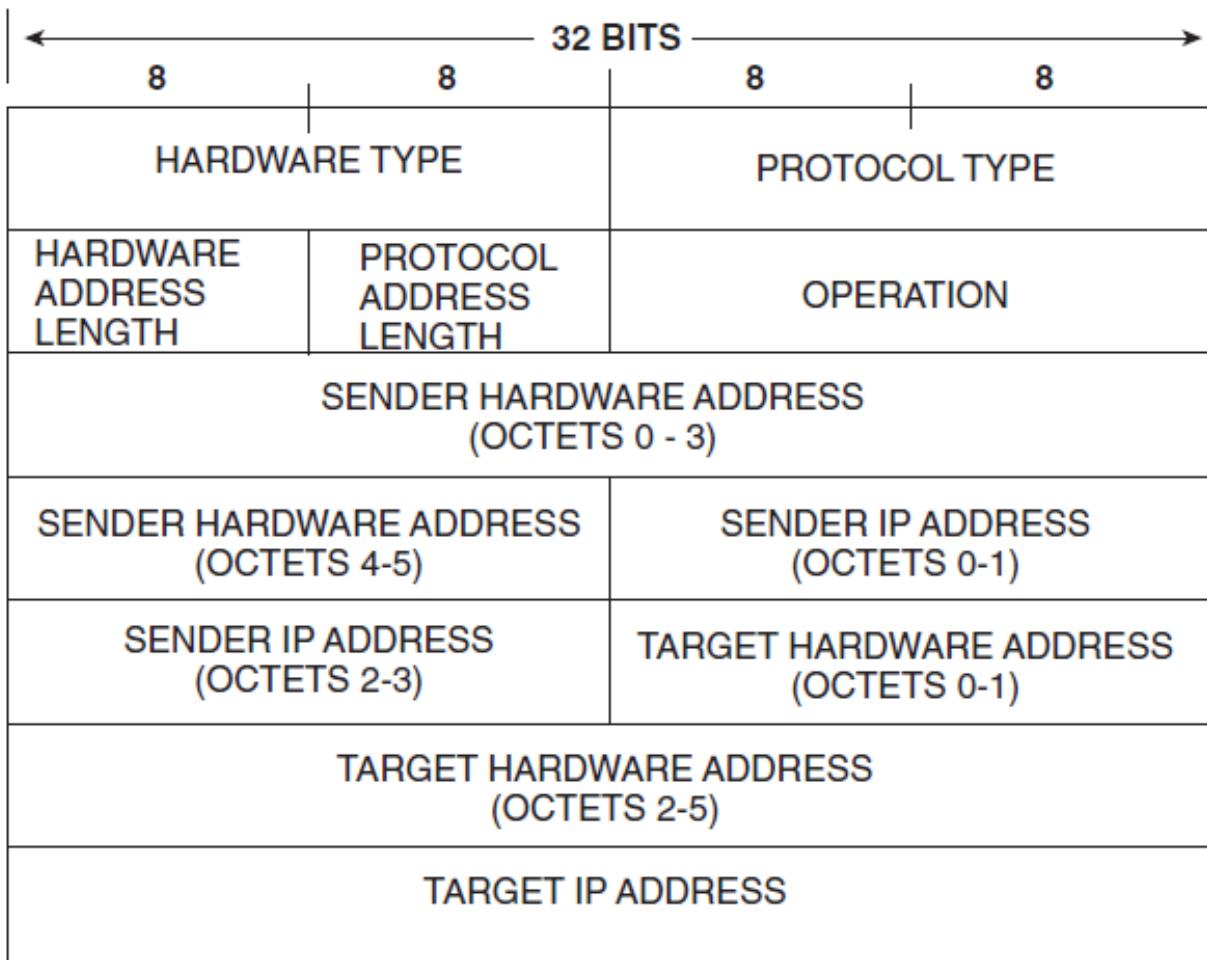
## 9 Структуры наблюдаемых пакетов заголовков

### Ethernet II



- Preamble – последовательность бит, по сути, не являющаяся частью ETH заголовка определяющая начало Ethernet фрейма.
- DA (Destination Address) – MAC адрес назначения, может быть юникастом, мультикастом, бродкастом.
- SA (Source Address) – MAC адрес отправителя. Всегда юникаст.
- E-TYPE (EtherType) – Идентифицирует L3 протокол (к примеру 0x0800 – Ipv4, 0x86DD – IPv6, 0x8100- указывает что фрейм тегирован заголовком 802.1q, и т.д. Список всех EtherType – [standards.ieee.org/develop/regauth/ethertype/eth.txt](http://standards.ieee.org/develop/regauth/ethertype/eth.txt) )
- Payload – L3 пакет размером от 46 до 1500 байт
- FCS (Frame Check Sequences) – 4 байтное значение CRC используемое для выявления ошибок передачи. Вычисляется отправляющей стороной, и помещается в поле FCS. Принимающая сторона вычисляет данное значение самостоятельно и сравнивает с полученным.

### ARP



1. Hardware Type - 16-битное поле, определяющее “тип канальной среды”. Наиболее часто используемые типы представлены в таблице ниже

Номер	Тип среды
1	Ethernet
15	Frame Relay
17	HDLC
18	Fiber Channel
20	Serial Link

2. Protocol Type - 16-битное поле, определяющее протокол сетевого уровня, который отправитель связывает с идентификатором канала передачи данных. Для протокола IP версии 4 значение данного поля равно 0x0800
3. Hardware Address Length - 8-битное поле, определяющее длину идентификатора канальной среды в байтах. MAC-адреса имеют длину 48 бит или 6 байт.

4. Protocol Address Length - 8-битное поле, определяющее длину адреса сетевого уровня в байтах. IP-адреса имеет длину 32 бита или 4 байта.
5. Operation - 16-битное поле, которое определяет какой тип пакета ARP используется:
- ARP Request - 1
  - ARP Reply - 2
  - Reverse ARP Request - 3
  - Reverse ARP Reply - 4
  - Inverse ARP Request - 8
  - Inverse ARP Reply - 9
6. Последние 20 байт приходятся на адресацию канальной среды и сетевого уровня источника и назначения запроса (MAC-адрес 6 байт \* 2 + IP-адрес 4 байт \* 2 = 20)

## IPv4

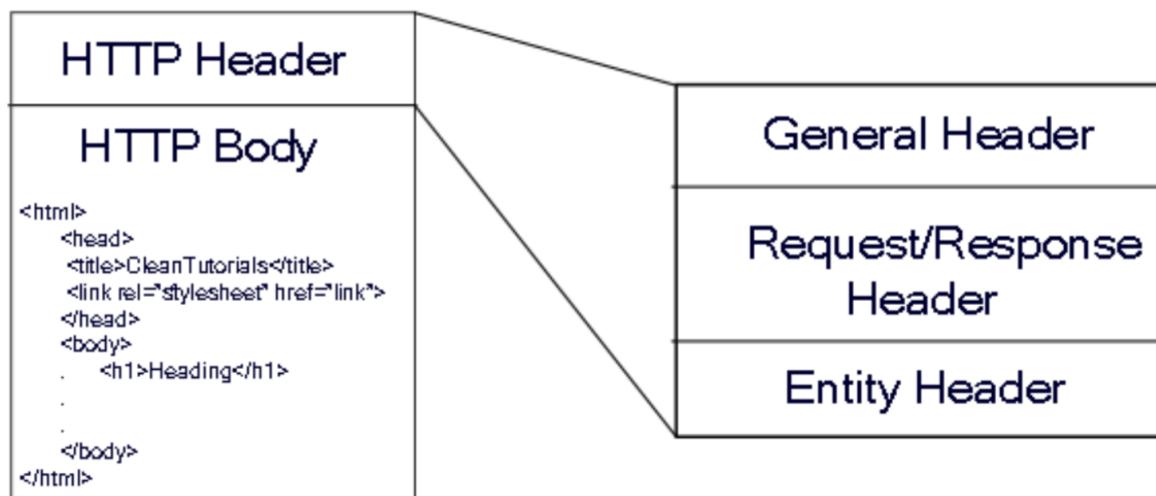
Биты	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31				
Номер версии	Длина заголовка		Тип сервиса (DS-байт)				Общая длина (байт)																													
Идентификатор пакета																										Флаги		Смещение фрагмента (кратно 8 байтам)								
-		DF		MF																																
Время жизни (TTL – Time To Live)		Протокол (6 - TCP, 17 - UDP, 1 - ICMP)		Контрольная сумма заголовка (в дополнительном коде)																																
IP-адрес источника																										IP-адрес назначения										
																										Параметры		Наполнение								

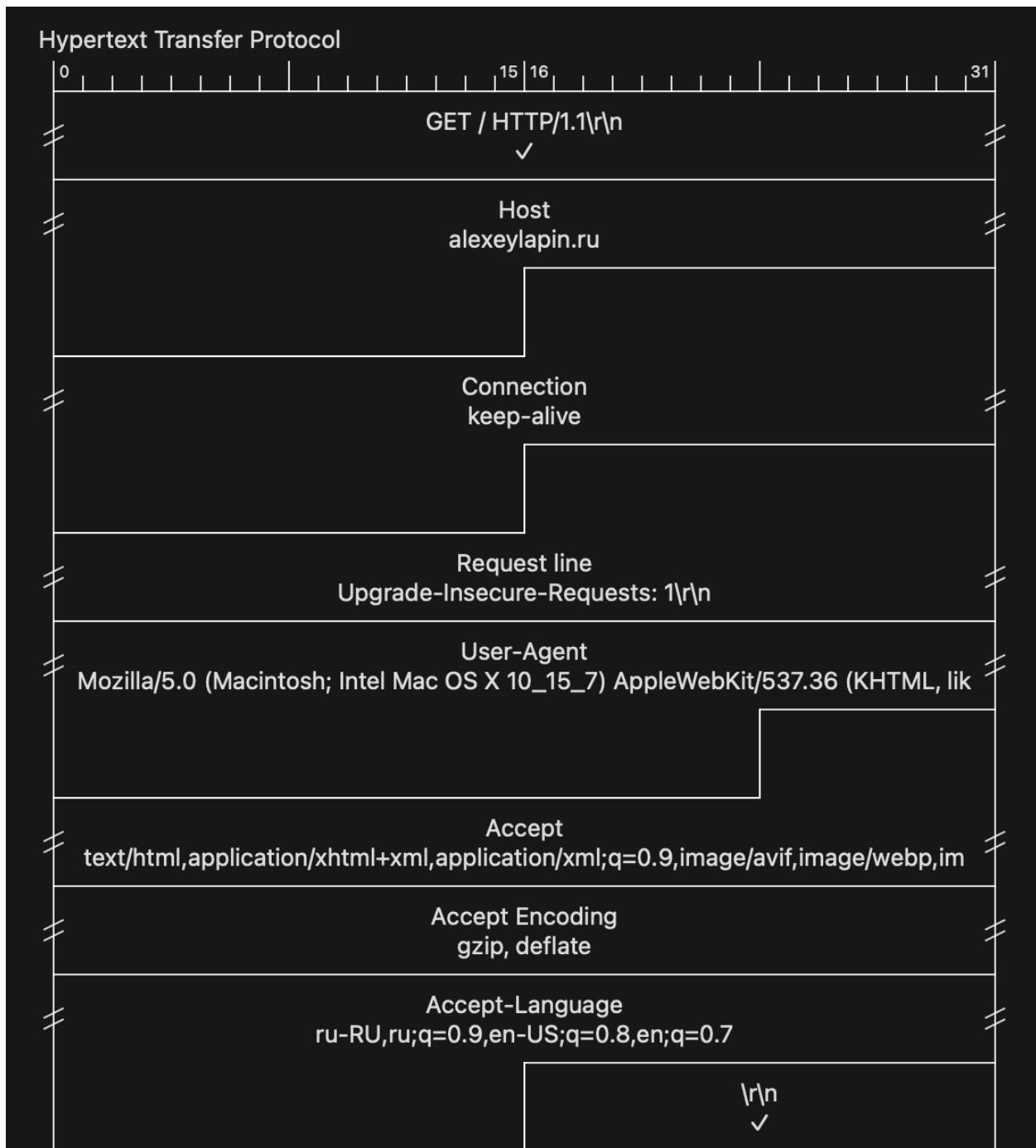
## ICMP

Биты	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Тип (Type)		Код (Kod)		Контрольная сумма (Check Sum)																												
Служебная информация (зависит от типа и кода)																																

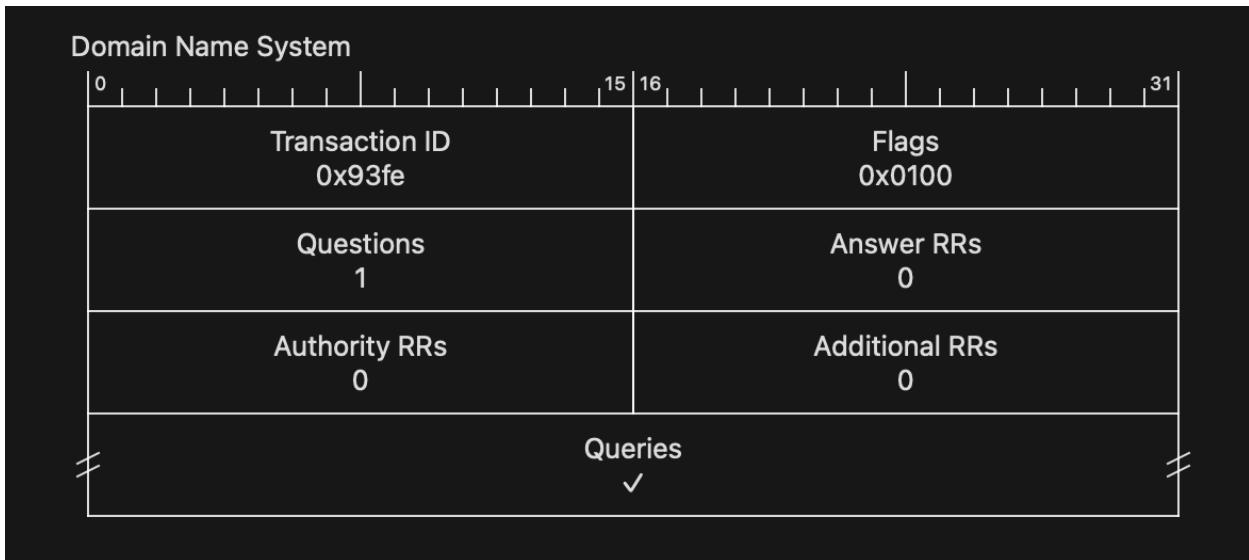
## HTTP

## HTTP Request/response

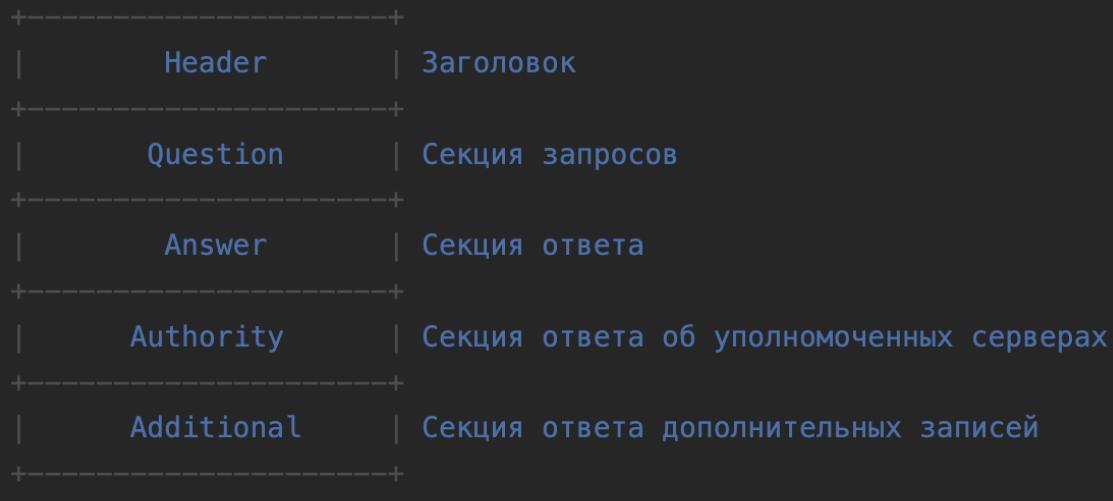




DNS

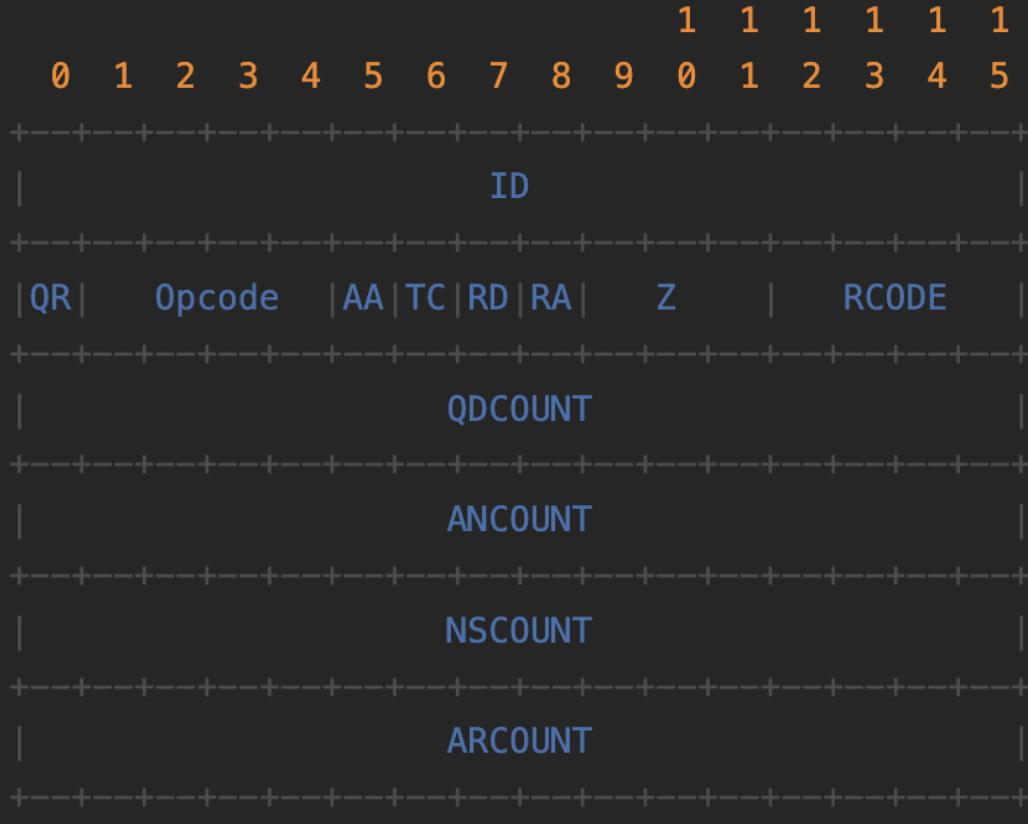


## Структура пакета DNS



1. Header — Заголовок DNS пакета, состоящий из 12 октет.
2. Question section — в этой секции DNS-клиент передает запросы DNS-серверу сообщая о том, для какого имени необходимо разрешить (зарезолвить) запись DNS, а также какого типа (NS, A, TXT и т.д.). Сервер при ответе, копирует эту информацию и отдает клиенту обратно в этой же секции.
3. Answer section — сервер сообщает клиенту ответ или несколько ответов на запрос, в котором сообщает вышеуказанные данные.
4. Authoritative Section — содержит сведения о том, с помощью каких авторитетных серверов было получена информация включенная в секцию DNS-ответа.
5. Additional Record Section — дополнительные записи, которые относятся к запросу, но не являются строго ответами на вопрос.

## Структура заголовка DNS

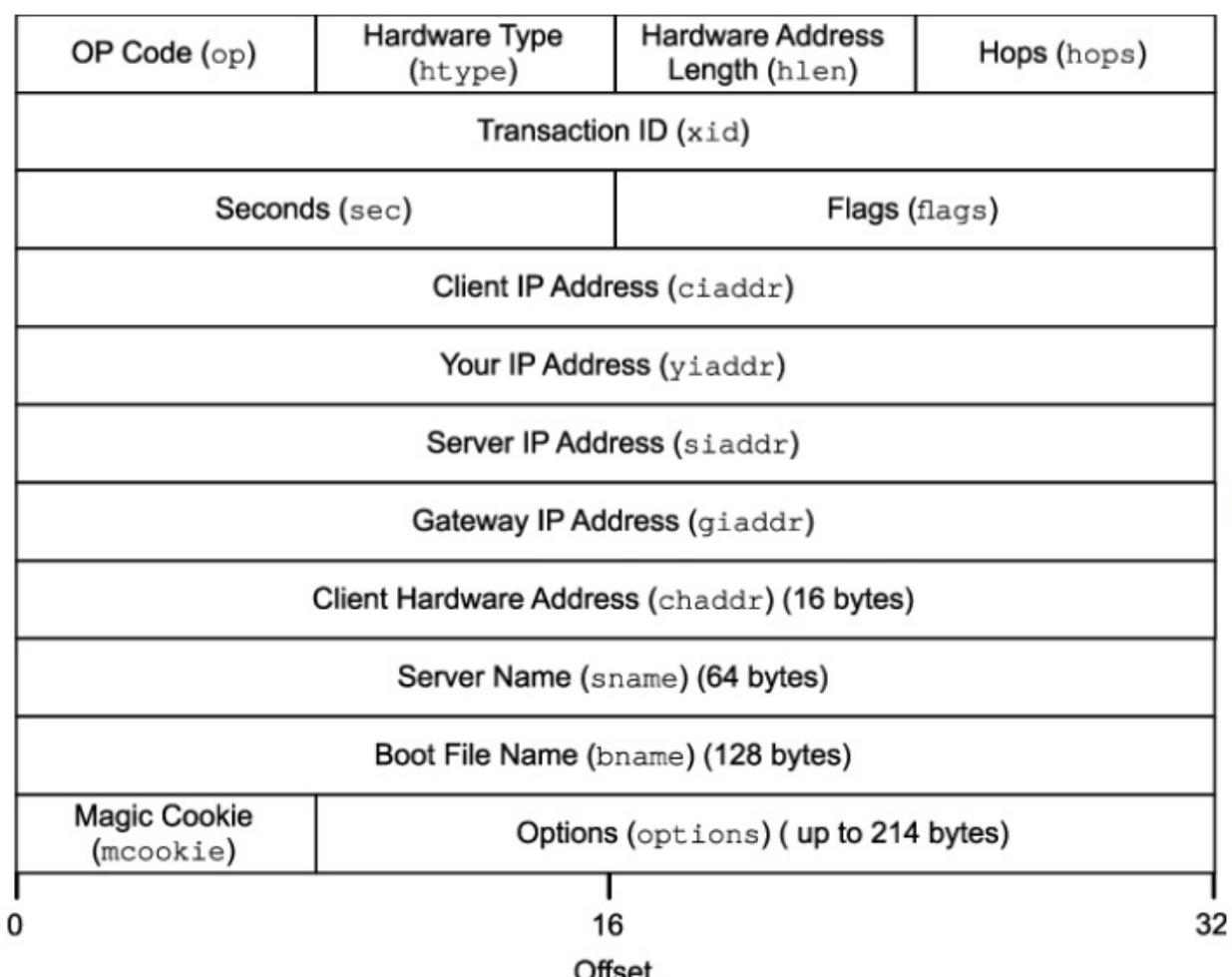


1. ID (16 бит) — данное поле используется как уникальный идентификатор транзакции. Указывает на то, что пакет принадлежит одной и той же сессии “запросов-ответов” и занимает 16 бит.
2. QR (1 бит) — данный бит служит для индентификации того, является ли пакет запросом (QR = 0) или ответом (QR = 1).
3. Opcode (4 бита) — с помощью данного кода клиент может указать тип запроса, где обычное значение:
  - 0 — стандартный запрос,
  - 1 — инверсный запрос,
  - 2 — запрос статуса сервера.
  - 3-15 – зарезервированы на будущее.
4. AA (1 бит) — данное поле имеет смысл только в DNS-ответах от сервера и сообщает о том, является ли ответ авторитетным либо нет.
5. TC (1 бит) — данный флаг устанавливается в пакете ответе в том случае если сервер не смог поместить всю необходимую информацию в пакет из-за существующих ограничений.

6. RD (1 бит) — этот однобитовый флаг устанавливается в запросе и копируется в ответ. Если он флаг устанавливается в запросе — это значит, что клиент просит сервер не сообщать ему промежуточных ответов, а вернуть только IP-адрес.
7. RA (1 бит) — отправляется только в ответах, и сообщает о том, что сервер поддерживает рекурсию
8. Z (3 бита) — являются зарезервированными и всегда равны нулю.
9. RCODE (4 бита) — это поле служит для уведомления клиентов о том, успешно ли выполнен запрос или с ошибкой.
  - 0 — значит запрос прошел без ошибок;
  - 1 — ошибка связана с тем, что сервер не смог понять форму запроса;
  - 2 — эта ошибка с некорректной работой сервера имен;
  - 3 — имя, которое разрешает клиент не существует в данном домене;
  - 4 — сервер не может выполнить запрос данного типа;
  - 5 — этот код означает, что сервер не может удовлетворить запроса клиента в силу административных ограничений безопасности.
10. QDCOUNT(16 бит) — количество записей в секции запросов
11. ANCOUNT(16 бит) — количество записей в секции ответы
12. NSCOUNT(16 бит) — количество записей в Authority Section
13. ARCOUNT(16 бит) — количество записей в Additional Record Section

Подробнее: Структура DNS пакета - [habr.com](https://habr.com)

### **Структура DNS пакета**



- Operation code - тип DHCP-сообщения. Если значение  $0x01$  – запрос к серверу, иначе — оно является ответом DHCP-сервера.
  - Hardware Type - тип адреса на канальном уровне. DHCP может работать поверх различных протоколов на канальном уровне, поэтому нужно указывать на каком именно.
  - Hardware Length - длина аппаратного адреса в байтах.
  - Hops - количество промежуточных маршрутизаторов (так называемых агентов ретрансляции DHCP), через которые прошло сообщение.
  - Transaction ID - Уникальный идентификатор транзакции в 4 байта, генерируемый клиентом в начале процесса получения адреса.
  - Seconds Elapsed - Время в секундах с момента начала процесса получения адреса. Может не использоваться (в этом случае оно устанавливается в 0)
  - Flags - Поле для флагов — специальных параметров протокола DHCP
  - Client IP Address - ip-адрес клиента. Заполняется только в том случае, если клиент уже имеет собственный ip-адрес (это возможно, если клиент выполняет процедуру обновления адреса по истечении срока аренды).
  - Your IP Address - Новый ip-адрес клиента, предложенный сервером.

- Server IP Address - IP-адрес сервера.
- Client Hardware Address - MAC клиента.
- Server Hostname – доменное имя сервера (если присутствует).
- Boot File - служит указателем для бездисковых рабочих станций о имени файла инициализации на сервере.
- Options – информация для динамической конфигурации хоста.

## 10 Выводы

В ходе выполнения лабораторной работы мною были получены навыки работы с анализатором трафика Wireshark, где были захвачены и изучены пакеты разных протоколов, их расположение по уровням TCP/IP модели, назначение и структура.

Убедился в том, каким образом представляются пакеты, как они обрабатываются при проходе от соседних уровней или одного и того же уровня.