

Университет ИТМО
Мегафакультет компьютерных технологий и управления
Факультет программной инженерии и компьютерной техники

Лабораторная работа №5
Анализ трафика компьютерных
сетей утилитой Wireshark

Группа: Р33101
Студент: Патутин Владимир
Крюков Андрей
Преподаватель: Тропченко Андрей Александрович

Санкт-Петербург
2022 г

Цель работы

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

Задание

1. Запустить Wireshark. В появившемся окне выбрать интерфейс для которого необходимо осуществлять анализ проходящих через него пакетов. После выбора адаптера, нужно запустить процесс захвата трафика (кнопка Start).
2. Инициировать процесс передачи трафика по сети
3. Установить значение “Фильтра”, чтобы из всего множества перехватываемых пакетов Wireshark отобразил только те, которые имеют отношение к выполняемому заданию.
4. Дождаться появления данных в списке захваченных пакетов и убедиться, что количество пакетов достаточно для выполнения задания.
5. Сохранить захваченный трафик в файл-трассу (pcap).
6. Описать в отчёте структуру наблюдаемых PDU (т.е. протокольных блоков данных: кадров, пакетов, сегментов) как для запросов, так и ответов.
7. Написать в отчёте ответы на вопросы задания
8. Поместить в отчёт скриншоты окна Wireshark, иллюстрирующие ответы из вышеуказанных п.6 и п.7.

URL - Адрес сайта, в название которого лексически входит фамилия студента:

<http://patutin.pro/>

Ip = 89.253.220.234

Выполнение работы

1. Анализ трафика утилиты ping

Необходимо отследить и проанализировать трафик, создаваемый утилитой ping, запустив её следующим образом из командной строки: “ping -l размер_пакета адрес_сайта_по_варианту”. В качестве “размера_пакета” необходимо поочерёдно использовать различные значения от 100 до 10000, самостоятельно выбрав шаг изменения. По результатам анализа собранной трассы, необходимо ответить на следующие вопросы и выполнить указанные задания.

Ping -s 2000 89.253.220.234

Wireshark:

ip

ip.addr == 89.253.220.234

Структура IP/ICMP пакета:

IP Datagram				
	Bits 0–7	Bits 8–15	Bits 16–23	Bits 24–31
IP Header (20 bytes)	Version/IHL	Type of service	Length	
	Identification		flags and offset	
	Time To Live (TTL)	Protocol	Checksum	
	Source IP address			
	Destination IP address			
ICMP Header (8 bytes)	Type of message	Code	Checksum	
	Header Data			
ICMP Payload (optional)	Payload Data			

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

Да, фрагментация есть для пакетов размером более 1490 байт. Увидеть количество фрагментов можно в поле **more fragments** если оно указано **1**, то это **промежуточный фрагмент** также фрагменты помечаются черным, а финальный с флагом **more fragments = 0** это последний.

ip.addr == 89.253.220.234

Io.	Time	Source	Destination	Protocol	Length Info
62	1.527302804	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b802) [Reassembled in #63]
63	1.527316381	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=1/256, ttl=64 (no response found!)
112	2.540238701	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b91f) [Reassembled in #113]
113	2.540267620	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=2/512, ttl=64 (no response found!)
154	3.550105606	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b9f3) [Reassembled in #155]
155	3.550117397	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=3/768, ttl=64 (no response found!)

Frame 113: 562 bytes on wire (4496 bits), 562 bytes captured (4496 bits) on interface wlp0s20f3, id 0
Ethernet II, Src: IntelCor_e3:75:f1 (94:e2:3c:e3:75:f1), Dst: Routerbo_af:1d:76 (08:55:31:af:1d:76)
Internet Protocol Version 4, Src: 192.168.88.118, Dst: 89.253.220.234

Internet Control Message Protocol

Type: 8 (Echo (ping) request)

Code: 0

Checksum: 0xfaf4 [correct]

[Checksum Status: Good]

Identifier (BE): 7 (0x0007)

Identifier (LE): 1792 (0x0700)

Sequence Number (BE): 2 (0x0002)

Sequence Number (LE): 512 (0x0200)

[No response seen]

Timestamp from icmp data: May 11, 2022 17:14:25.000000000 MSK

[Timestamp from icmp data (relative): 0.193261421 seconds]

Data (1992 bytes)

154	3.550105606	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b9f3) [Reassembled in #155]
155	3.550117397	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=3/768, ttl=64 (no response found!)
7170	97.796608312	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=052a) [Reassembled in #7171]
7171	97.796632238	192.168.88.118	89.253.220.234	ICMP	362 Echo (ping) request id=0x0008, seq=1/256, ttl=64 (no response found!)
7205	98.803557652	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=0639) [Reassembled in #7206]
7206	98.803605485	192.168.88.118	89.253.220.234	ICMP	362 Echo (ping) request id=0x0008, seq=2/512, ttl=64 (no response found!)
7257	99.816792332	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=0651) [Reassembled in #7258]
7258	99.816814643	192.168.88.118	89.253.220.234	ICMP	362 Echo (ping) request id=0x0008, seq=3/768, ttl=64 (no response found!)
7294	100.833440769	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=06c7) [Reassembled in #7295]
7295	100.833461683	192.168.88.118	89.253.220.234	ICMP	362 Echo (ping) request id=0x0008, seq=4/1024, ttl=64 (no response found!)
7462	104.408279649	192.168.88.118	89.253.220.234	ICMP	1342 Echo (ping) request id=0x0009, seq=1/256, ttl=64 (reply in 7467)
7467	104.448762349	89.253.220.234	192.168.88.118	ICMP	1342 Echo (ping) reply id=0x0009, seq=1/256, ttl=57 (request in 7462)
7515	105.410058682	192.168.88.118	89.253.220.234	ICMP	1342 Echo (ping) request id=0x0009, seq=2/512, ttl=64 (reply in 7516)
7516	105.437254995	89.253.220.234	192.168.88.118	ICMP	1342 Echo (ping) reply id=0x0009, seq=2/512, ttl=57 (request in 7515)

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

Поле **More fragments**: 1 - промежуточный, 0 - последний

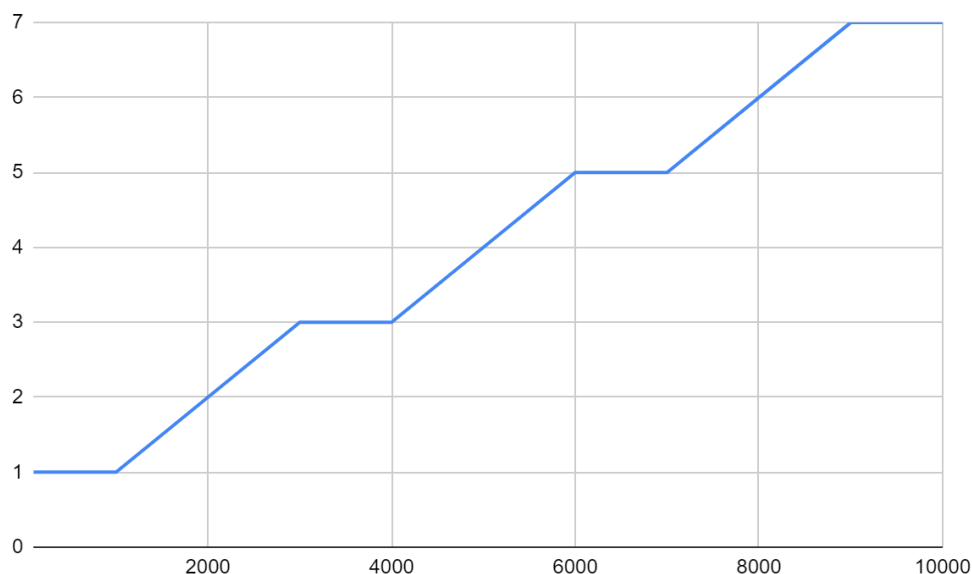
ip.addr == 89.253.220.234					
No.	Time	Source	Destination	Protocol	Length/Info
62	1.527302804	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b802) [Reassembled in #63]
63	1.527316381	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=1/256, ttl=64 (no response found!)
112	2.540238701	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b91f) [Reassembled in #113]
113	2.540267620	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=2/512, ttl=64 (no response found!)
154	3.550105606	192.168.88.118	89.253.220.234	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=b9f3) [Reassembled in #155]
155	3.550117397	192.168.88.118	89.253.220.234	ICMP	562 Echo (ping) request id=0x0007, seq=3/768, ttl=64 (no response found!)

Frame 154: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface wlp0s20f3, id 0	
Ethernet II, Src: IntelCor_e3:75:f1 (94:e2:3c:e3:75:f1), Dst: Routerbo_af:1d:76 (08:55:31:af:1d:76)	
Internet Protocol Version 4, Src: 192.168.88.118, Dst: 89.253.220.234	
0100 = Version: 4 ... 0101 = Header Length: 20 bytes (5) ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT) Total Length: 1500 Identification: 0xb9f3 (47603) ▶ Flags: 0x20, More fragments ... 0 0000 0000 0000 = Fragment Offset: 0 Time to Live: 64 Protocol: ICMP (1) Header Checksum: 0x4b27 [validation disabled] [Header checksum status: Unverified] Source Address: 192.168.88.118 Destination Address: 89.253.220.234 [Reassembled IPv4 in frame: 155]	
Data (1480 bytes)	
Data: 080066cd00070003c2c47b6200000000441903000000000101112131415161718191a1b...	
[Length: 1480]	
0000	08 55 31 af 1d 76 94 e2 3c e3 75 f1 08 00 45 00 .U1..v...<.u...E
0100	05 dc b9 f3 20 00 40 01 4b 27 c0 a8 58 76 59 fd ...@.K'...XY.
0200	dc ea 08 00 66 cd 00 07 00 03 c2 c4 7b 62 00 00 .f.....{b...
0300	00 00 44 19 03 00 00 00 00 00 10 11 12 13 14 15 ..n.....

- Чему равно количество фрагментов при передаче ping-пакетов?

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

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



5. Как изменить поле TTL с помощью утилиты ping?

Использовать ключ `-i <TTL>` (time to live)

6. Что содержится в поле данных ping-пакета?

ASCII-символы, также поле может содержать временную метку или номер пакета.

2. Анализ трафика утилиты `tracert` (`traceroute`)

Необходимо отследить и проанализировать трафик, создаваемый утилитой `tracert` (или `traceroute` в Linux), запустив её следующим образом из командной строки:

`“tracert -d адрес_сайта_по_варианту”`

По результатам анализа собранной трассы, ответьте на следующие вопросы.

```
[yank0vy3rdna@yank0vy3rdna-Notebook ~]$ traceroute 89.253.220.234
traceroute to 89.253.220.234 (89.253.220.234), 30 hops max, 60 byte packets
 1  _gateway (192.168.88.1)  1.688 ms  1.791 ms  1.773 ms
 2  176-53-205-125.customer.pulnet.ru (176.53.205.125)  2.085 ms  2.532 ms  2.516 ms
 3  ae16-306.rt.km.spb.ru.retn.net (87.245.252.178)  106.157 ms  106.147 ms  106.091 ms
 4  ae8-9.rt.dl.msk.ru.retn.net (87.245.233.12)  13.040 ms  12.225 ms  13.011 ms
 5  87.245.193.163 (87.245.193.163)  12.996 ms  * * *
 6  * * *
 7  hw182.rusonyx.ru (89.253.192.35)  12.141 ms  11.911 ms  12.112 ms
 8  slimvps-1048611-17501.host4g.ru (89.253.220.234)  12.340 ms  13.218 ms  12.282 ms
[yank0vy3rdna@yank0vy3rdna-Notebook ~]$
```


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

Заголовок IP - 20 байт

Поле данных - 56 байт

2487	34.046079262	192.168.88.118	89.253.220.234	UDP	74 48386 → 33456 Len=32
2488	34.046093193	192.168.88.118	89.253.220.234	UDP	74 53481 → 33457 Len=32
2489	34.046106466	192.168.88.118	89.253.220.234	UDP	74 33961 → 33458 Len=32
2490	34.046128769	192.168.88.118	89.253.220.234	UDP	74 39291 → 33459 Len=32
2491	34.057914023	89.253.192.35	192.168.88.118	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
2493	34.058129290	89.253.192.35	192.168.88.118	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
2494	34.058129387	89.253.192.35	192.168.88.118	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
2495	34.058160037	192.168.88.118	89.253.220.234	UDP	74 44986 → 33460 Len=32
2496	34.058213675	192.168.88.118	89.253.220.234	UDP	74 35421 → 33461 Len=32
2497	34.058229930	192.168.88.118	89.253.220.234	UDP	74 57851 → 33462 Len=32
2498	34.058370702	89.253.220.234	192.168.88.118	ICMP	102 Destination unreachable (Port unreachable)
2499	34.058370927	89.253.220.234	192.168.88.118	ICMP	102 Destination unreachable (Port unreachable)
2500	34.059293145	89.253.220.234	192.168.88.118	ICMP	102 Destination unreachable (Port unreachable)
2501	34.059293315	89.253.220.234	192.168.88.118	ICMP	102 Destination unreachable (Port unreachable)
2502	34.059293363	89.253.220.234	192.168.88.118	ICMP	102 Destination unreachable (Port unreachable)
2503	34.070036481	89.253.220.234	192.168.88.118	ICMP	102 Destination unreachable (Port unreachable)
2508	34.135843679	87.245.252.178	192.168.88.118	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
2509	34.135875347	87.245.252.178	192.168.88.118	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
2510	34.135879874	87.245.252.178	192.168.88.118	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
2514	34.149078361	87.245.193.162	192.168.88.118	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
2515	34.149103332	87.245.193.162	192.168.88.118	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

Internet Protocol Version 4, Src: 87.245.233.12, Dst: 192.168.88.118

0100 = Version: 4

.... 0101 = Header Length: 20 bytes (5)

▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 56
Identification: 0x0000 (0)
Flags: 0x00
▸ ...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 251
Protocol: ICMP (1)
Header Checksum: 0x65a4 [validation disabled]
[Header checksum status: Unverified]
Source Address: 87.245.233.12
Destination Address: 192.168.88.118

Internet Control Message Protocol

Type: 11 (Time-to-live exceeded)
Code: 0 (Time to live exceeded in transit)

0000 94 e2 3c e3 75 f1 08 55 31 af 1d 76 08 00 65 00 ...< . . . 1

2. Как и почему изменяется поле TTL в следующих друг за другом ICMP-пакетах tracer?

На каждом шаге проверки TTL увеличивается на единицу, чтобы можно было последовательно получить IP-адреса каждого узла в цепочке передачи.

Каждый промежуточный узел отнимает от значения поля TTL единицу, чтобы можно было отследить количество пройденных узлов.

Когда TTL = 0 (или пакет достиг получателя), передача пакета прекращается.

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

Разный тип сообщений: у ping - 8, tracer - 11. Tracer также устанавливает разный TTL на каждом шаге проверки.

```

▶ Frame 18033: 1342 bytes on wire (10736 bits)
▶ Ethernet II, Src: IntelCor_e3:75:f1 (94:00:01:00:00:01), Dst: 89:25:23:20:23:40
▼ Internet Protocol Version 4, Src: 192.168.88.118, Dst: 89.253.220.234
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1328
    Identification: 0x43bc (17340)
    ▶ Flags: 0x40, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0xa20a [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.88.118
    Destination Address: 89.253.220.234
▼ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x3e9c [correct]
    [Checksum Status: Good]
    Identifier (BE): 10 (0x000a)
    Identifier (LE): 2560 (0x0a00)
    Sequence Number (RF): 1 (0x0001)

```

```

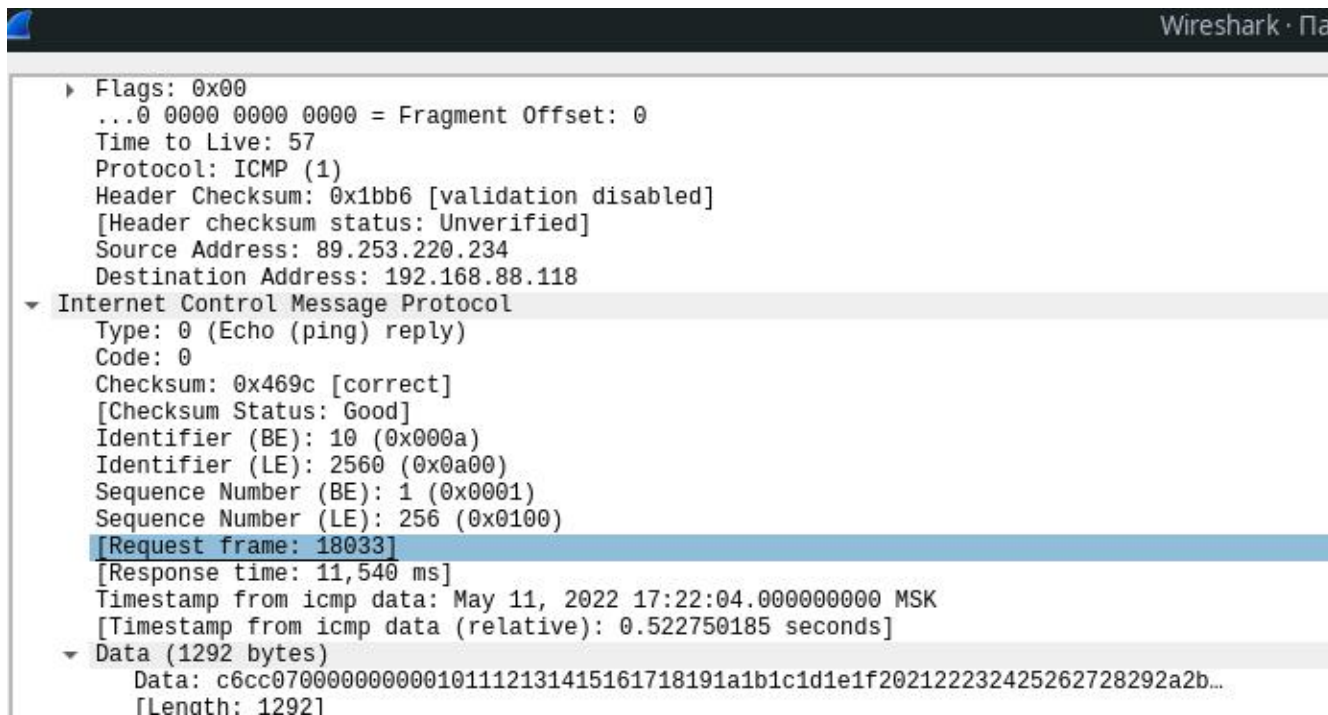
▶ Frame 2509: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
▶ Ethernet II, Src: Routerbo_af:1d:76 (08:55:31:af:1d:76), Dst: IntelCor_e3:75:f1 (94:00:01:00:00:01)
▼ Internet Protocol Version 4, Src: 87.245.252.178, Dst: 192.168.88.118
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x0000 (0)
    ▶ Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 253
    Protocol: ICMP (1)
    Header Checksum: 0x4ffe [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 87.245.252.178
    Destination Address: 192.168.88.118
▼ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0x3a45 [correct]
    [Checksum Status: Good]
    Unused: 00000000
▼ Internet Protocol Version 4, Src: 192.168.88.118, Dst: 89.253.220.234
    0100 .... = Version: 4

```

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

Error посылает промежуточный узел, когда TTL становится равен 0.

Reply посылает конечный узел, когда пакет успешно доходит до него.



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

Произойдет попытка определить имя узла по его IP адресу, т.е. будут произведены дополнительные запросы к DNS-серверу.

3. Анализ HTTP-трафика

Необходимо отследить и проанализировать HTTP-трафик, создаваемый браузером при посещении Интернет-сайта, заданного по варианту. В списке захваченных пакетов необходимо проанализировать следующую пару HTTP-сообщений (запрос-ответ):

- GET-сообщение от клиента (браузера);
- ответ сервера.

Для этого в поле с детальной информацией о пакете нужно развернуть строку «HTTP». Затем необходимо обновить страницу в браузере так, чтобы вместо «HTTP GET» был сгенерирован «HTTP CONDITIONAL GET» (так называемый «условный GET»). Условные запросы GET содержат поля If-Modified-Since, If-Match, If-Range и подобные, которые позволяют при повторном запросе не передавать редко изменяемые данные. В ответ на условный GET тело запрашиваемого ресурса передается только в том случае, если этот ресурс

изменялся после даты «If-Modified-Since». Если ресурс не изменялся, сервер вернет код статуса «304 Not Modified».

По результатам анализа собранной трассы покажите, каким образом протокол HTTP передавал содержимое страницы при первичном посещении страницы и при вторичном запросе-обновлении от браузера (т.е. при различных видах GET-запросов).

Структура TCP-датаграммы:

Структура заголовка				
Бит	0 — 3	4 — 6	7 — 15	16 — 31
0	Порт источника, Source Port			Порт назначения, Destination Port
32	Порядковый номер, Sequence Number (SN)			
64	Номер подтверждения, Acknowledgment Number (ACK SN)			
96	Длина заголовка, (Data offset)	Зарезервировано	Флаги	Размер Окна, Window size
128	Контрольная сумма, Checksum			Указатель важности, Urgent Point
160	Опции (необязательное, но используется практически всегда)			
160/192+	Данные			

Структура HTTP-сообщения:

1. Стартовая строка - <Метод> <URI> HTTP/<Версия>
2. Заголовки - набор параметров и их значений
3. Тело сообщения - опциональные данные

Первый GET-запрос - данные новые - получаем html сайта

No.	Time	Source	Destination	Protocol	Length	Info
322	6.769340319	192.168.88.118	89.253.220.234	TCP	74	35166 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3591613687 TSecr=0 WS=128
323	6.720622317	89.253.220.234	192.168.88.118	TCP	74	80 → 35166 [SYN, ACK] Seq=0 Ack=1 Wlen=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1609585884 TSecr=3591613687 WS=128
324	6.720653158	192.168.88.118	89.253.220.234	TCP	66	35166 → 80 [ACK] Seq=1 Ack=1 Wlen=64256 Len=0 TSval=3591613699 TSecr=1609585884
325	6.720704508	192.168.88.118	89.253.220.234	HTTP	141	GET / HTTP/1.1
327	6.732469508	89.253.220.234	192.168.88.118	TCP	66	80 → 35166 [ACK] Seq=1 Ack=76 Wlen=29056 Len=0 TSval=1609585896 TSecr=3591613699
329	6.733387602	89.253.220.234	192.168.88.118	HTTP	568	HTTP/1.1 301 Moved Permanently (text/html)
330	6.733417032	192.168.88.118	89.253.220.234	TCP	66	35166 → 80 [ACK] Seq=76 Ack=503 Wlen=63872 Len=0 TSval=3591613711 TSecr=1609585897
331	6.733602495	192.168.88.118	89.253.220.234	TCP	66	35166 → 80 [FIN, ACK] Seq=76 Ack=503 Wlen=64128 Len=0 TSval=3591613712 TSecr=1609585897
334	6.744966679	89.253.220.234	192.168.88.118	TCP	66	80 → 35166 [FIN, ACK] Seq=503 Ack=77 Wlen=29056 Len=0 TSval=1609585909 TSecr=3591613712
335	6.744990932	192.168.88.118	89.253.220.234	TCP	66	35166 → 80 [ACK] Seq=77 Ack=504 Wlen=64128 Len=0 TSval=3591613723 TSecr=1609585909

Второй GET-запрос вернул все то же, тк мы не переходили на https. Если перейти, то видно хендшейк и далее обмен шифрованными данными

2406	35.569752560	89.253.220.234	192.168.88.118	TCP	66 80 → 35168 [ACK] Seq=1 Ack=76 Win=29056 Len=0 TSval=1609614733 TSecr=3591642536
2407	35.570452735	89.253.220.234	192.168.88.118	HTTP	568 HTTP/1.1 301 Moved Permanently (text/html)
2408	35.570481602	192.168.88.118	89.253.220.234	TCP	66 35168 → 80 [ACK] Seq=76 Ack=503 Win=63872 Len=0 TSval=3591642549 TSecr=1609614734
2409	35.570752407	192.168.88.118	89.253.220.234	TCP	66 35168 → 80 [FIN, ACK] Seq=76 Ack=503 Win=64128 Len=0 TSval=3591642549 TSecr=1609614734
2410	35.581868498	89.253.220.234	192.168.88.118	TCP	66 80 → 35168 [FIN, ACK] Seq=503 Ack=77 Win=29056 Len=0 TSval=1609614746 TSecr=3591642549
2411	35.581923027	192.168.88.118	89.253.220.234	TCP	66 35168 → 80 [ACK] Seq=77 Ack=504 Win=64128 Len=0 TSval=3591642560 TSecr=1609614746
3126	47.969732254	192.168.88.118	89.253.220.234	TCP	74 35170 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3591654948 TSecr=0 WS=128
3133	47.981152702	89.253.220.234	192.168.88.118	TCP	74 80 → 35170 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1609627145 TSecr=3591654948 WS=128
3134	47.981189194	192.168.88.118	89.253.220.234	TCP	66 35170 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3591654959 TSecr=1609627145
3135	47.981325395	192.168.88.118	89.253.220.234	HTTP	704 GET / HTTP/1.1
3143	48.017732216	89.253.220.234	192.168.88.118	TCP	66 80 → 35170 [ACK] Seq=1 Ack=639 Win=30336 Len=0 TSval=1609627157 TSecr=3591654959
3144	48.017732285	89.253.220.234	192.168.88.118	HTTP	568 HTTP/1.1 301 Moved Permanently (text/html)
3145	48.017757526	192.168.88.118	89.253.220.234	TCP	66 35170 → 80 [ACK] Seq=639 Ack=593 Win=63872 Len=0 TSval=3591654996 TSecr=1609627157
3148	48.026144072	192.168.88.118	89.253.220.234	TCP	74 51676 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3591654998 TSecr=0 WS=128
3152	48.034746787	89.253.220.234	192.168.88.118	TCP	74 443 → 51676 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1609627195 TSecr=3591654998 WS=128
3153	48.034780822	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3591655013 TSecr=1609627195
3154	48.034922958	192.168.88.118	89.253.220.234	TLSv1.2	583 Client Hello
3158	48.046757839	89.253.220.234	192.168.88.118	TCP	66 443 → 51676 [ACK] Seq=1 Ack=518 Win=30000 Len=0 TSval=1609627210 TSecr=3591655013
3160	48.050316206	89.253.220.234	192.168.88.118	TLSv1.2	1514 Server Hello
3161	48.050340513	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=518 Ack=1449 Win=63104 Len=0 TSval=3591655028 TSecr=1609627212
3162	48.051732083	89.253.220.234	192.168.88.118	TLSv1.2	2762 Certificate, Server Key Exchange, Server Hello Done
3163	48.051734548	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=518 Ack=4145 Win=81056 Len=0 TSval=3591655030 TSecr=1609627212
3169	48.056448167	192.168.88.118	89.253.220.234	TLSv1.2	192 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
3171	48.056903009	192.168.88.118	89.253.220.234	TLSv1.2	952 Application Data
3174	48.060805753	89.253.220.234	192.168.88.118	TLSv1.2	324 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
3175	48.060829470	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=1530 Ack=4403 Win=63872 Len=0 TSval=3591655046 TSecr=1609627232
3184	48.108924109	89.253.220.234	192.168.88.118	TCP	66 443 → 51676 [ACK] Seq=4403 Ack=1530 Win=31872 Len=0 TSval=1609627273 TSecr=3591655035
3193	48.187382396	89.253.220.234	192.168.88.118	TCP	1514 443 → 51676 [ACK] Seq=4403 Ack=1530 Win=31872 Len=1448 TSval=1609627346 TSecr=3591655035 [TCP segment of a reassembled PDU]
3194	48.187423369	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=1530 Ack=5851 Win=63104 Len=0 TSval=3591655165 TSecr=1609627346
3195	48.187749492	89.253.220.234	192.168.88.118	TCP	1514 443 → 51676 [ACK] Seq=5851 Ack=1530 Win=31872 Len=1448 TSval=1609627346 TSecr=3591655035 [TCP segment of a reassembled PDU]
3196	48.187758447	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=1530 Ack=7299 Win=63104 Len=0 TSval=3591655166 TSecr=1609627346
3197	48.189289206	89.253.220.234	192.168.88.118	TCP	2962 443 → 51676 [ACK] Seq=7299 Ack=1530 Win=31872 Len=2896 TSval=1609627346 TSecr=3591655035 [TCP segment of a reassembled PDU]
3198	48.189289378	89.253.220.234	192.168.88.118	TLSv1.2	1540 Application Data
3199	48.189305628	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=1530 Ack=10195 Win=61056 Len=0 TSval=3591655167 TSecr=1609627346
3200	48.189324081	192.168.88.118	89.253.220.234	TCP	66 51676 → 443 [ACK] Seq=1530 Ack=11669 Win=59648 Len=0 TSval=3591655167 TSecr=1609627347
3201	48.210601572	192.168.88.118	89.253.220.234	TCP	74 51678 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3591655189 TSecr=0 WS=128

Через chrome devtools можно увидеть https запросы


DevTools is now available in Russian! Always match Chrome's language Switch DevTools to Russian Don't show again

Elements Console Sources **Network** Performance Memory Application Security >>

Preserve log Disable cache No throttling

Filter ☐ Invert ☐ Hide data URLs ☒ All Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other

☐ Has blocked cookies ☐ Blocked Requests ☐ 3rd-party requests



Name	Status	Type	Initiator	Size	Time	Waterfall
patutin.pro	200	document	Other	7.2 kB	91 ms	
logo-2.png	200	png	(index)	(memory...)	0 ms	
c8fce992e36b0d23df843808f20df...	200	jpeg	(index)	(memory...)	0 ms	
63c498a38ce306b36edd42cbd208...	200	jpeg	(index)	(memory...)	0 ms	
4edbefddfd578f2fe0b53f78371ccf8...	200	jpeg	(index)	(memory...)	0 ms	
2848569532db75b901092e94249f...	200	jpeg	(index)	(memory...)	0 ms	
cd020e97a2bbb94915703a73846c...	200	jpeg	(index)	(memory...)	0 ms	
79dbf6e52d5401c7ac584440ce16a...	200	jpeg	(index)	(memory...)	0 ms	
color.css?1652279438	200	stylesheet	(index)	1.1 kB	14 ms	
9cf59f92e20e1cc03c0f623ecf8720...	200	jpeg	(index)	(memory...)	0 ms	
3119975c889ed75ba4a2b5625cc2f...	200	jpeg	(index)	(memory...)	0 ms	
1dbbec8923478f05fd20134dfc1ea...	200	jpeg	(index)	(memory...)	0 ms	
3473e0bb56e7e5823a21bb396f15...	200	jpeg	(index)	(memory...)	0 ms	
4a66c61d96d2c2e012767b83e8e4...	200	jpeg	(index)	(memory...)	0 ms	
334ab8da1228155fc66957f793d4b...	200	jpeg	(index)	(memory...)	0 ms	
ddd920129f8f361a45975917bea5...	200	jpeg	(index)	(memory...)	0 ms	
tf.png	200	png	(index)	(memory...)	0 ms	
html5.js	200	script	(index)	(memory...)	0 ms	
kernel_main.js?1605172109280146	200	script	(index)	(memory...)	0 ms	
template_9a9e57f3356f5faad789df...	200	script	(index)	(memory...)	0 ms	
jquery.2site.carousel.js	200	script	(index)	(memory...)	0 ms	
PTS55F_W.woff	200	font	font.css	(memory...)	0 ms	
PTS75F_W.woff	200	font	font.css	(memory...)	0 ms	
icons.png	200	png	template_c373d8...	(memory...)	0 ms	
search-btn.png	200	png	template_c373d8...	(memory...)	0 ms	
icons.png	200	png	template_c373d8...	(memory...)	0 ms	
icons.png	200	png	template_c373d8...	(memory...)	0 ms	
icons.png	200	png	template_c373d8...	(memory...)	0 ms	
switcher.png	200	png	template_c373d8...	(memory...)	0 ms	
switcher.png	200	png	template_c373d8...	(memory...)	0 ms	
icons.png	200	png	template_c373d8...	(memory...)	0 ms	
social.png	200	png	template_c373d8...	(memory...)	0 ms	
social.png	200	png	template_c373d8...	(memory...)	0 ms	
social.png	200	png	template_c373d8...	(memory...)	0 ms	
social.png	200	png	template_c373d8...	(memory...)	0 ms	
6Qcu2IYSZi	200	script	(index);342	(memory...)	0 ms	
ba.js	200	script	(index);346	(memory...)	0 ms	
switcher.png	200	png	template_c373d8...	(memory...)	0 ms	
switcher.png	200	png	template_c373d8...	(memory...)	0 ms	
6Qcu2IYSZi	200	xhr	6Qcu2IYSZi:1	(disk cac...)	1 ms	
bx_stat	200	xhr	ba.js:1	534 B	63 ms	
6Qcu2IYSZi?rnd=0.326807966981...	200	xhr	6Qcu2IYSZi:1	136 B	15 ms	
prompt.js	200	script	playback.js:1	7.3 kB	27 ms	
bundle_ru_RU.js?rand=1651824215	200	script	6Qcu2IYSZi:1	(memory...)	0 ms	
favicon.ico	200	x-icon	Other	1.7 kB	13 ms	
data:image/svg+xml,...	200	svg+xml	Other	(memory...)	0 ms	
data:image/svg+xml,...	200	svg+xml	Other	(memory...)	0 ms	
data:image/svg+xml,...	200	svg+xml	Other	(memory...)	0 ms	

Видно, что вебсервер не возвращает 304 ответ, таким образом запрещая кэшировать исходные коды сайта

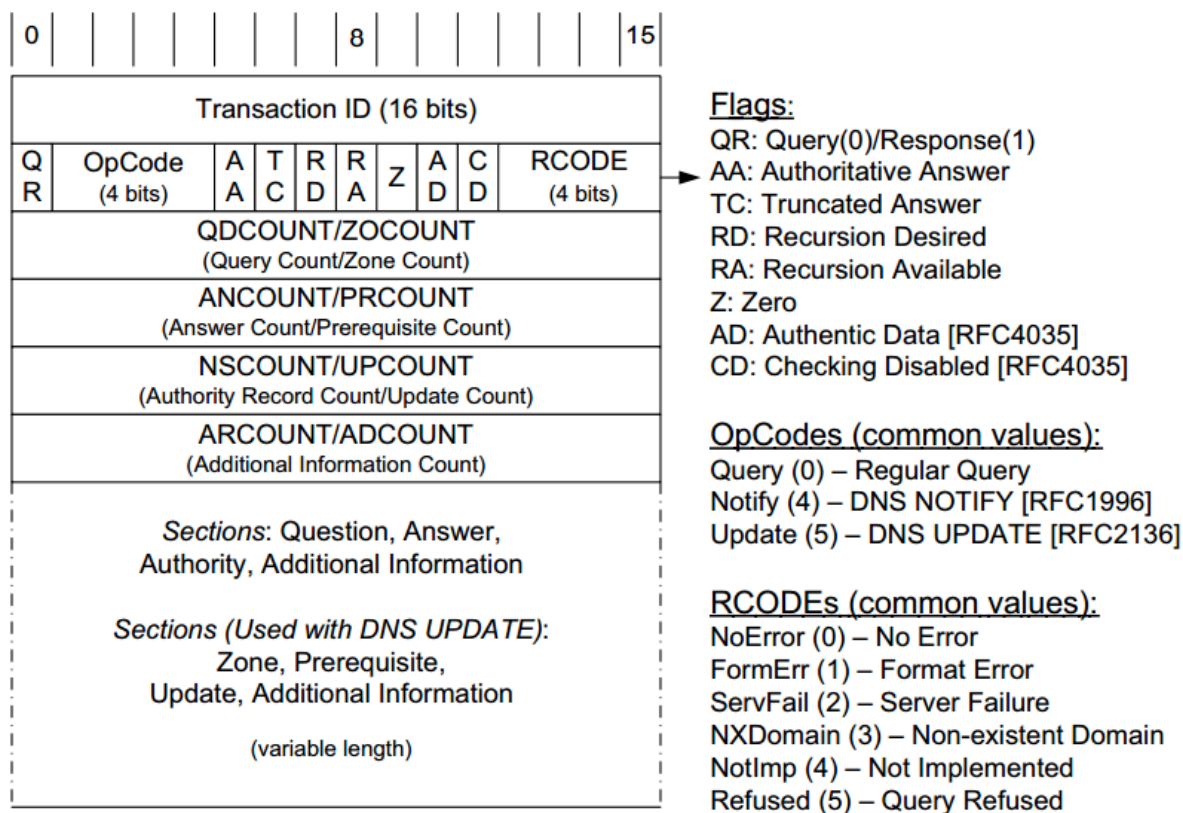
4. Анализ DNS-трафика

Необходимо отследить и проанализировать трафик протокола DNS, сгенерированный в результате выполнения следующих действий:

- настроить Wireshark-фильтр: “ip.addr == ваш_IP_адрес”;
- очистить кэш DNS с помощью команды `ipconfig /flushdns` в командной строке;
- очистить кэш браузера;
- зайти на Интернет-сайт, заданный по варианту.

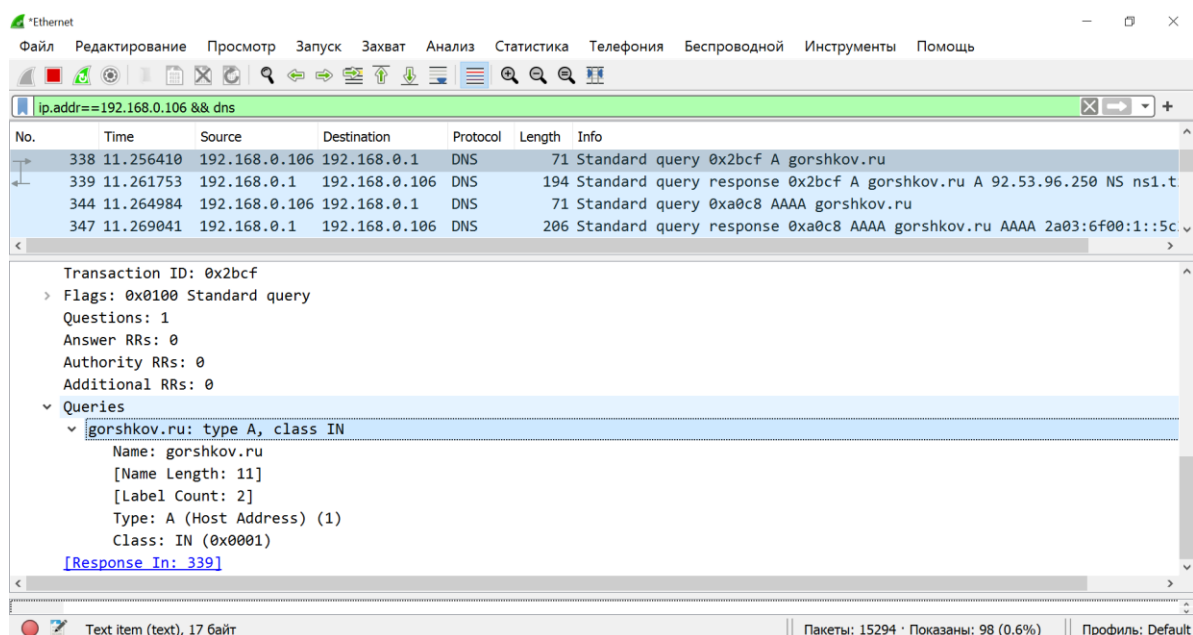
По результатам анализа собранной трассы, ответьте на следующие вопросы.

Структура DNS-пакета:



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

Запрос отправляется не на сайт, а на специальный DNS-сервер.



flushds

2. Какие бывают типы DNS-запросов?

Рекурсивный - DNS-серверу посылается доменное имя, для которого он должен вернуть IP-адрес. DNS-сервер может обратиться к другим серверам для выполнения запроса

Итеративный - при получении такого запроса DNS-сервер не опрашивает другие сервера, а возвращает либо IP-адрес, либо имя другого DNS-сервера, который может знать ответ

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

Если адрес, на котором хранится изображение отличается от адреса сайта.

5. Анализ ARP-трафика

Необходимо отследить и проанализировать трафик протокола ARP, сгенерированный в результате выполнения следующих действий:

- очистить ARP-таблицу командой “netsh interface ip delete arpcache”;
- очистить кэш браузера;
- зайти на Интернет-сайт, заданный по варианту.

По результатам анализа собранной трассы, ответьте на следующие

вопросы.

Структура ARP-пакета:

Структура ARP-пакета

+	0 - 7	8 - 15	16 - 31
0	Hardware type (HTYPE)		Protocol type (PTYPE)
32	Hardware length (HLEN)	Protocol length (PLEN)	Operation (OPER)
64	Sender hardware address (SHA)		
?	Sender protocol address (SPA)		
?	Target hardware address (THA)		
?	Target protocol address (TPA)		

HTYPE – тип сети, назначено каждому стандартному типу LAN. Например, для Ethernet = 1.

PTYPE – тип протокола. Например, для протокола IPv4 = 0x0800.

HLEN – длина физического адреса в байтах. Для MAC-адреса = 6.

PLEN – длина логического адреса в байтах. Для IPv4 = 4, IPv6 = 16.

OPER – операция, тип пакета. Запрос ARP = 1, ответ ARP = 2.

SHA – физический адрес передатчика.

SPA – логический адрес передатчика.

THA – физический адрес приемника.

TPA – логический адрес приемника. переменная длина, задается HLEN, PLEN

1. Какие MAC-адреса присутствуют в захваченных пакетах ARP-протокола?

Что означают эти адреса? Какие устройства они идентифицируют?

B0:4e:26:d0:12:f0 - адрес отправителя (маршрутизатор) (ip = 192.168.0.1)

B4:b6:86:dc:75:55 - адрес устройства получателя (компьютер, с которого производится запрос на сайт) (ip = 192.168.0.106)

00:00:00:00:00:00 - broadcast-адрес

*Ethernet

Файл Редактирование Просмотр Запуск Захват Анализ Статистика Телефония Беспроводной Инструменты Помощь

arp

No.	Time	Source	Destination	Protocol	Length	Info
490	7.047973	Espressi_ed:6a:43	Broadcast	ARP	60	ARP Announcement for 192.168.0.103
783	7.157471	Tp-LinkT_d0:12:f0	HewlettP_dc:75:55	ARP	60	Who has 192.168.0.106? Tell 192.168.0.1
784	7.157489	HewlettP_dc:75:55	Tp-LinkT_d0:12:f0	ARP	42	192.168.0.106 is at b4:b6:86:dc:75:55
2153	17.036939	Espressi_ed:6a:43	Broadcast	ARP	60	ARP Announcement for 192.168.0.103
2292	27.038997	Espressi_ed:6a:43	Broadcast	ARP	60	ARP Announcement for 192.168.0.103
2468	36.533845	Tp-LinkT_d0:12:f0	HewlettP_dc:75:55	ARP	60	Who has 192.168.0.106? Tell 192.168.0.1
2469	36.533859	HewlettP_dc:75:55	Tp-LinkT_d0:12:f0	ARP	42	192.168.0.106 is at b4:b6:86:dc:75:55

Address Resolution Protocol (request)

Hardware type: Ethernet (1)
 Protocol type: IPv4 (0x0800)
 Hardware size: 6
 Protocol size: 4
 Opcode: request (1)
 Sender MAC address: Tp-LinkT_d0:12:f0 (b0:4e:26:d0:12:f0)
 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: 192.168.0.106

Sender MAC address (arp.src.hw_mac), 6 байт

Пакеты: 79127 · Показаны: 374 (0.5%) | Профиль: Default

*Ethernet

Файл Редактирование Просмотр Запуск Захват Анализ Статистика Телефония Беспроводной Инструменты Помощь

arp

No.	Time	Source	Destination	Protocol	Length	Info
490	7.047973	Espressi_ed:6a:43	Broadcast	ARP	60	ARP Announcement for 192.168.0.103
783	7.157471	Tp-LinkT_d0:12:f0	HewlettP_dc:75:55	ARP	60	Who has 192.168.0.106? Tell 192.168.0.1
784	7.157489	HewlettP_dc:75:55	Tp-LinkT_d0:12:f0	ARP	42	192.168.0.106 is at b4:b6:86:dc:75:55
2153	17.036939	Espressi_ed:6a:43	Broadcast	ARP	60	ARP Announcement for 192.168.0.103
2292	27.038997	Espressi_ed:6a:43	Broadcast	ARP	60	ARP Announcement for 192.168.0.103
2468	36.533845	Tp-LinkT_d0:12:f0	HewlettP_dc:75:55	ARP	60	Who has 192.168.0.106? Tell 192.168.0.1
2469	36.533859	HewlettP_dc:75:55	Tp-LinkT_d0:12:f0	ARP	42	192.168.0.106 is at b4:b6:86:dc:75:55

Address Resolution Protocol (reply)

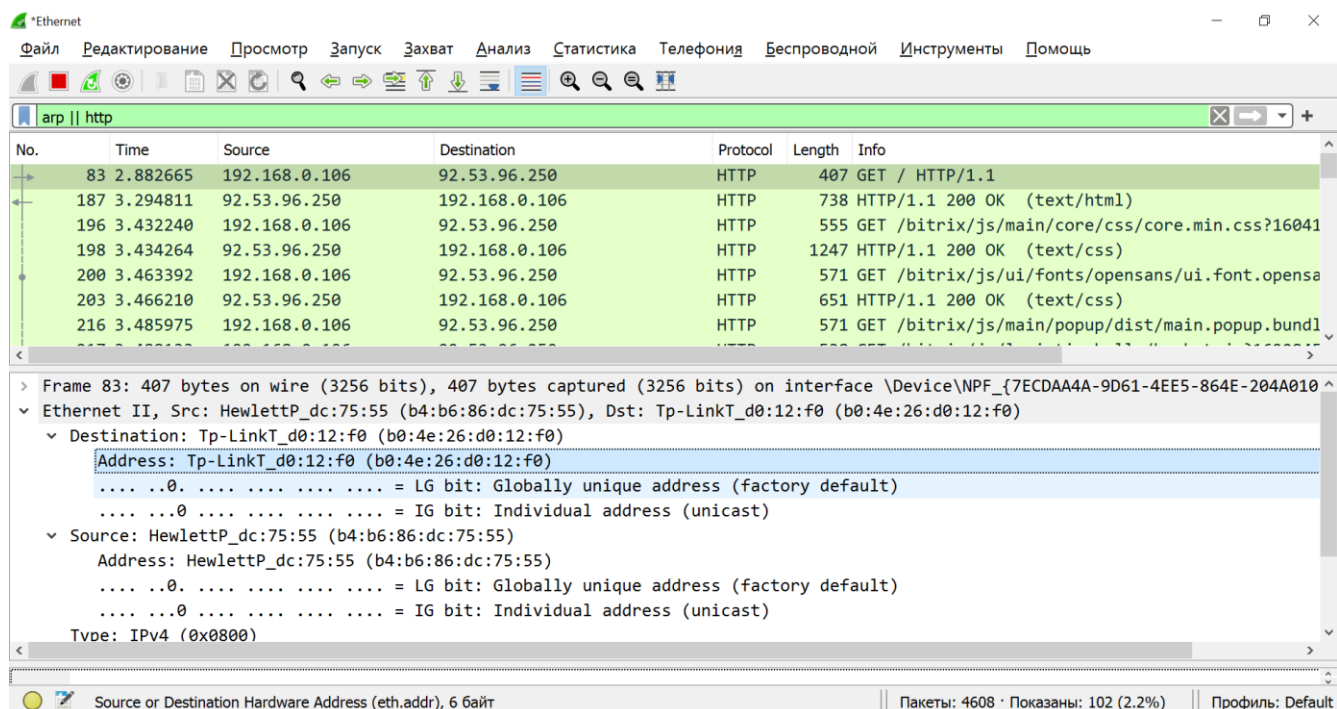
Hardware type: Ethernet (1)
 Protocol type: IPv4 (0x0800)
 Hardware size: 6
 Protocol size: 4
 Opcode: reply (2)
 Sender MAC address: HewlettP_dc:75:55 (b4:b6:86:dc:75:55)
 Sender IP address: 192.168.0.106
 Target MAC address: Tp-LinkT_d0:12:f0 (b0:4e:26:d0:12:f0)
 Target IP address: 192.168.0.1

Sender MAC address (arp.src.hw_mac), 6 байт

Пакеты: 79689 · Показаны: 379 (0.5%) | Профиль: Default

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

Присутствует MAC-адрес устройства, с которого производится http запрос и MAC-адрес маршрутизатора



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

Чтобы узел-получатель мог добавить информацию об узле-отправителе в свою ARP-таблицу.

6. Анализ трафика утилиты nslookup

Необходимо отследить и проанализировать трафик протокола DNS, сгенерированный в результате выполнения следующих действий:

1. Настроить Wireshark-фильтр: “ip.addr == ваш_IP_адрес”.
2. Запустить в командной строке команду “nslookup адрес_сайта_по_варианту”.
3. Дождаться отправки трёх DNS-запросов и трёх DNS-ответов (в работе нужно использовать только последние из них, т.к. первые два набора запросов/ответов специфичны для nslookup и не генерируются другими сетевыми приложениями).
4. Повторить предыдущие два шага, используя команду: “nslookup -type=NS имя_сайта_по_варианту”.

nslookup 92.53.96.250

```
[yank0vy3rdna@yank0vy3rdna-Notebook ~]$ nslookup 89.253.220.234
234.220.253.89.in-addr.arpa      name = slimvps-1048611-17501.host4g.ru.
dns
Authoritative answers can be found from:
.       nameserver = j.root-servers.net.
.       nameserver = k.root-servers.net.
.       nameserver = l.root-servers.net.
.       nameserver = m.root-servers.net.
.       nameserver = b.root-servers.net.
.       nameserver = c.root-servers.net.
.       nameserver = d.root-servers.net.
.       nameserver = e.root-servers.net.
.       nameserver = f.root-servers.net.
.       nameserver = g.root-servers.net.
.       nameserver = h.root-servers.net.
.       nameserver = a.root-servers.net.
.       nameserver = i.root-servers.net.
a.root-servers.net      internet address = 198.41.0.4
```

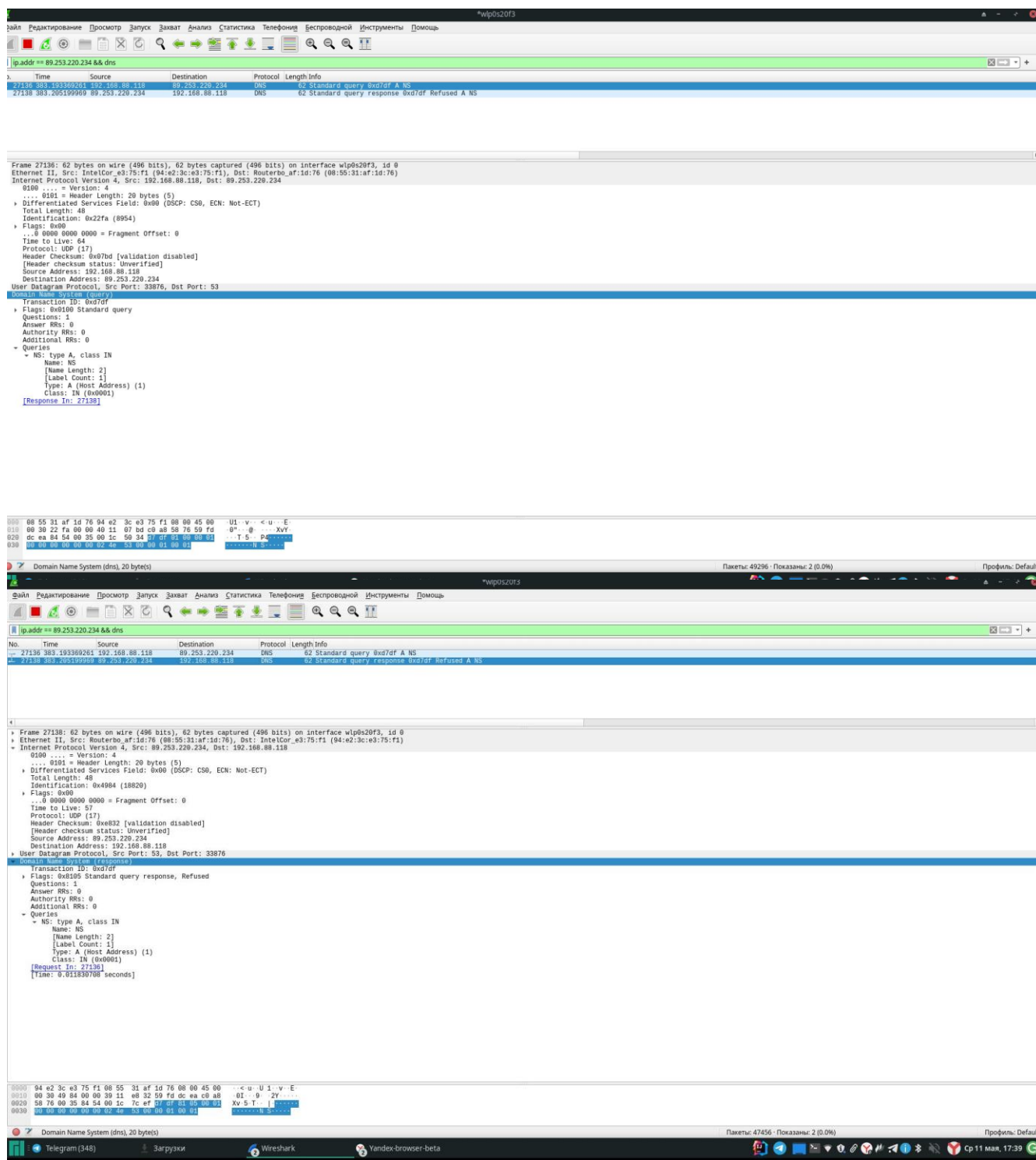
nslookup -type=NS 92.53.96.250

По результатам анализа собранной трассы, ответьте на следующие вопросы.

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

При запуске в п.2 утилита ищет IP-адрес хоста (запись типа A (IPv4) или AAAA (IPv6)).

При запуске в п.4 утилита ищет Name Server для запрашиваемого хоста.



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

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

```

Time to Live: 57
Protocol: UDP (17)
Header Checksum: 0xe832 [validation disabled]
[Header checksum status: Unverified]
Source Address: 89.253.220.234
Destination Address: 192.168.88.118
▶ User Datagram Protocol, Src Port: 53, Dst Port: 33876
▼ Domain Name System (response)
  Transaction ID: 0xd7df
  ▶ Flags: 0x8105 Standard query response, Refused
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 0
  ▼ Queries
    ▼ NS: type A, class IN
      Name: NS
      [Name Length: 2]
      [Label Count: 1]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      [Request In: 27136]
      [Time: 0.011830708 seconds]

```

0000	94 e2 3c e3 75 f1 08 55 31 af 1d 76 08 00 45 00	..<u..U 1..v..E..
0010	00 30 49 84 00 00 39 11 e8 32 59 fd dc ea c0 a8	..0I...9...2Y.....
0020	58 76 00 35 84 54 00 1c 7c ef d7 df 81 05 00 01	Xv.5.T..
0030	00 00 00 00 00 00 02 4e 53 00 00 01 00 01N S.....

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

```

[yank0vy3rdna@yank0vy3rdna-Notebook ~]$ dig patutin.pro
27136 383.193369261 19
27138 383.205199969 89
<<>> DiG 9.18.0 <<>> patutin.pro
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY status: NOERROR, id: 17731
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 13, ADDITIONAL: 1

;; QUESTION SECTION:
patutin.pro.
Type: A (Host Address) (1)
Class: IN (0x0001)
13102 In:IN136 A 89.253.220.234
[time: 0.011830708 seconds]

;; ANSWER SECTION:
987 IN NS c.root-servers.net.
987 IN NS d.root-servers.net.
987 IN NS e.root-servers.net.
987 IN NS f.root-servers.net.
987 IN NS g.root-servers.net.
987 IN NS h.root-servers.net.
987 IN NS a.root-servers.net.
987 IN NS i.root-servers.net.
987 IN NS j.root-servers.net.
987 IN NS k.root-servers.net.
987 IN NS l.root-servers.net.
987 IN NS m.root-servers.net.
987 IN NS b.root-servers.net.

;; ADDITIONAL SECTION:
a.root-servers.net. 162562 IN A 198.41.0.4

Internet Protocol Version 4
0100 .... = Version: 4
.... 0101 = Header
Differentiated Services Code Point: 0
Total Length: 48
Identification: 0x4
Flags: 0x00
Source Address: 89.253.220.234
Destination Address: 192.168.88.1
Protocol: UDP (17)
Header Checksum: 0x0000
[time: 0.011830708 seconds]
Query time: 3 msec
SERVER: 192.168.88.1#53(192.168.88.1) (UDP)
WHEN: Wed May 11 17:38:36 MSK 2022
MSG SIZE rcvd: 272

```

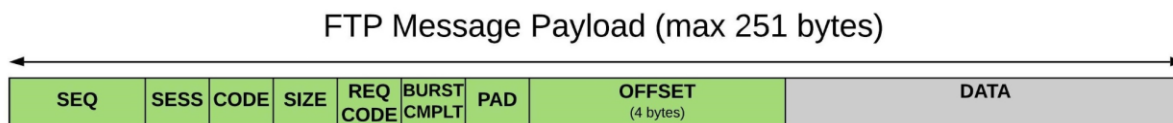

7. Анализ FTP-трафика

Необходимо отследить и проанализировать трафик протокола FTP, сгенерированный в результате выполнения следующих действий:

- настроить Wireshark-фильтр «ftp || ftp-data»;
- скачать в браузере небольшой файл с соответствующего варианту FTP-сервера в Интернете.

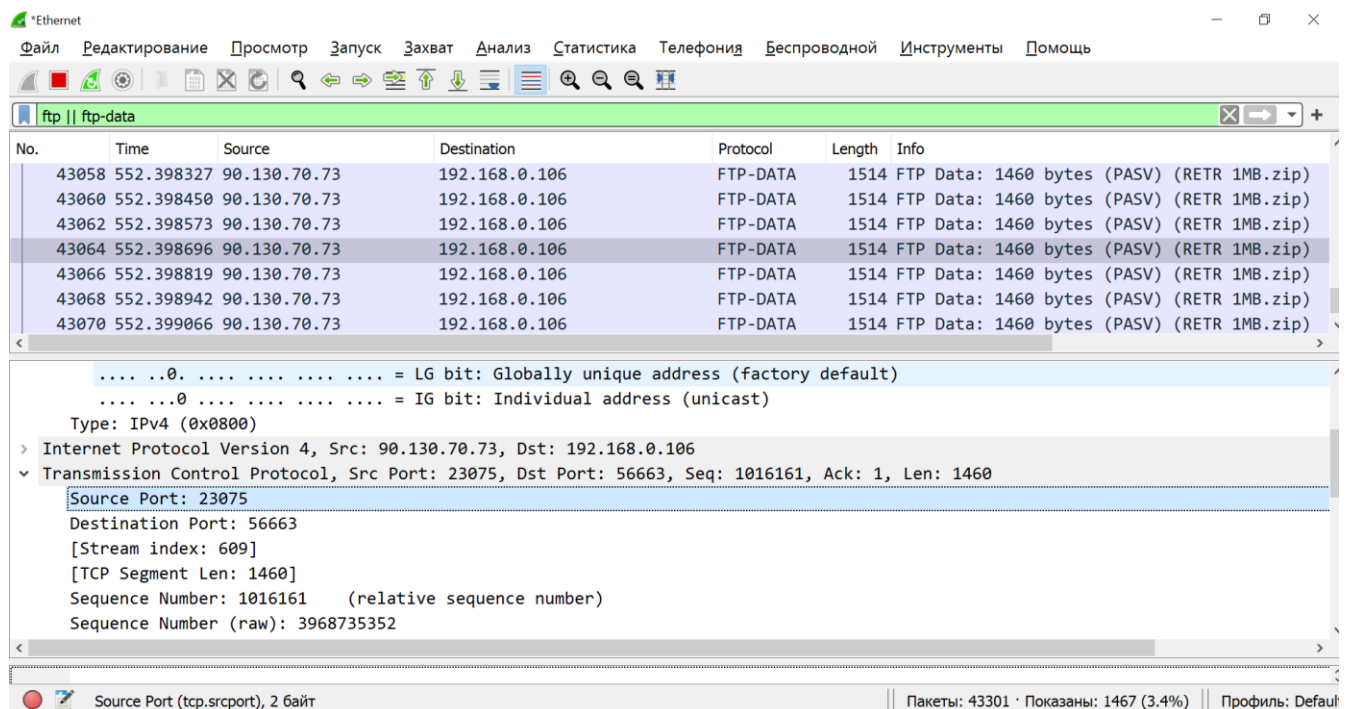
По результатам анализа собранной трассы, ответьте на следующие вопросы.

Структура FTP-сообщения:



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

1460 байт



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

Для потока управления на сервере используется порт 21. Для передачи данных используется порт 20, если передача идет в активном режиме, либо с любого порта клиента к любому порту сервера в пассивном режиме.

The top screenshot shows a Wireshark capture of FTP traffic. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	Info
34123	435.593046	90.130.70.73	192.168.0.106	FTP-DATA	1447	FTP Data: 1393 bytes (PASV) (LIST -a)
34127	435.605384	90.130.70.73	192.168.0.106	FTP	78	Response: 226 Directory send OK.
34358	465.691465	192.168.0.106	90.130.70.73	FTP	62	Request: TYPE I
34359	465.703802	90.130.70.73	192.168.0.106	FTP	85	Response: 200 Switching to Binary mode.
34745	495.745585	192.168.0.106	90.130.70.73	FTP	62	Request: TYPE I

The packet details pane for the selected packet (No. 34123) shows:

- Type: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 90.130.70.73, Dst: 192.168.0.106
- Transmission Control Protocol, Src Port: 21, Dst Port: 56589, Seq: 330, Ack: 85, Len: 24
 - Source Port: 21
 - Destination Port: 56589
 - [Stream index: 540]
 - [TCP Segment Len: 24]
 - Sequence Number: 330 (relative sequence number)
 - Sequence Number (raw): 129473212
 - [Next Sequence Number: 354 (relative sequence number)]
 - Acknowledgment Number: 85 (relative ack number)

The bottom screenshot shows another Wireshark capture of FTP traffic. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	Info
43058	552.398327	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)
43060	552.398450	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)
43062	552.398573	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)
43064	552.398696	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)
43066	552.398819	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)
43068	552.398942	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)
43070	552.399066	90.130.70.73	192.168.0.106	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (RETR 1MB.zip)

The packet details pane for the selected packet (No. 43058) shows:

- Type: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 90.130.70.73, Dst: 192.168.0.106
- Transmission Control Protocol, Src Port: 23075, Dst Port: 56663, Seq: 1016161, Ack: 1, Len: 1460
 - Source Port: 23075
 - Destination Port: 56663
 - [Stream index: 609]
 - [TCP Segment Len: 1460]
 - Sequence Number: 1016161 (relative sequence number)
 - Sequence Number (raw): 3968735352

3. Чем отличаются пакеты FTP от FTP-DATA?

В FTP-пакетах идет передача команд и ответов сервера на них. Эти пакеты небольшого размера и не содержат данных. FTP-DATA-пакеты содержат передаваемые данные, они значительно большего размера.

Вывод

Мы познакомились с приложением Wireshark, созданное для анализа трафика, научились писать фильтры для запросов, разобрались в том из чего состоят пакеты, изучили их структуру, и проанализировали ее, ответив на вопросы.