Лабораторная работа №4

Студент: Султанов Артур Радикович, группа: P3313

Расшифровка HTTPS

Расшифровка HTTPS-трафика Firefox на Linux в Wireshark с помощью SSLKEYLOGFILE

```
$ SSLKEYLOGFILE="$HOME/firefox-keys.log" firefox &
$ ls -lah ~/firefox-keys.log
-rw-r--r-- 1 sultanowskii sultanowskii 9.1K May  2 11:52
/home/sultanowskii/firefox-keys.log
```

Записанный расшифрованный HTTP-трафик не прикладывается к материалам ЛР из соображений безопасности.

Подготовка

Для сбора пакетов был выбран интерфейс `wlp0s20f3` - т.к. ноутбук подключен к Wi-Fi.

Целевым веб-сайтом был выбран https://github.com/sultanowskii - мой Github-профиль.

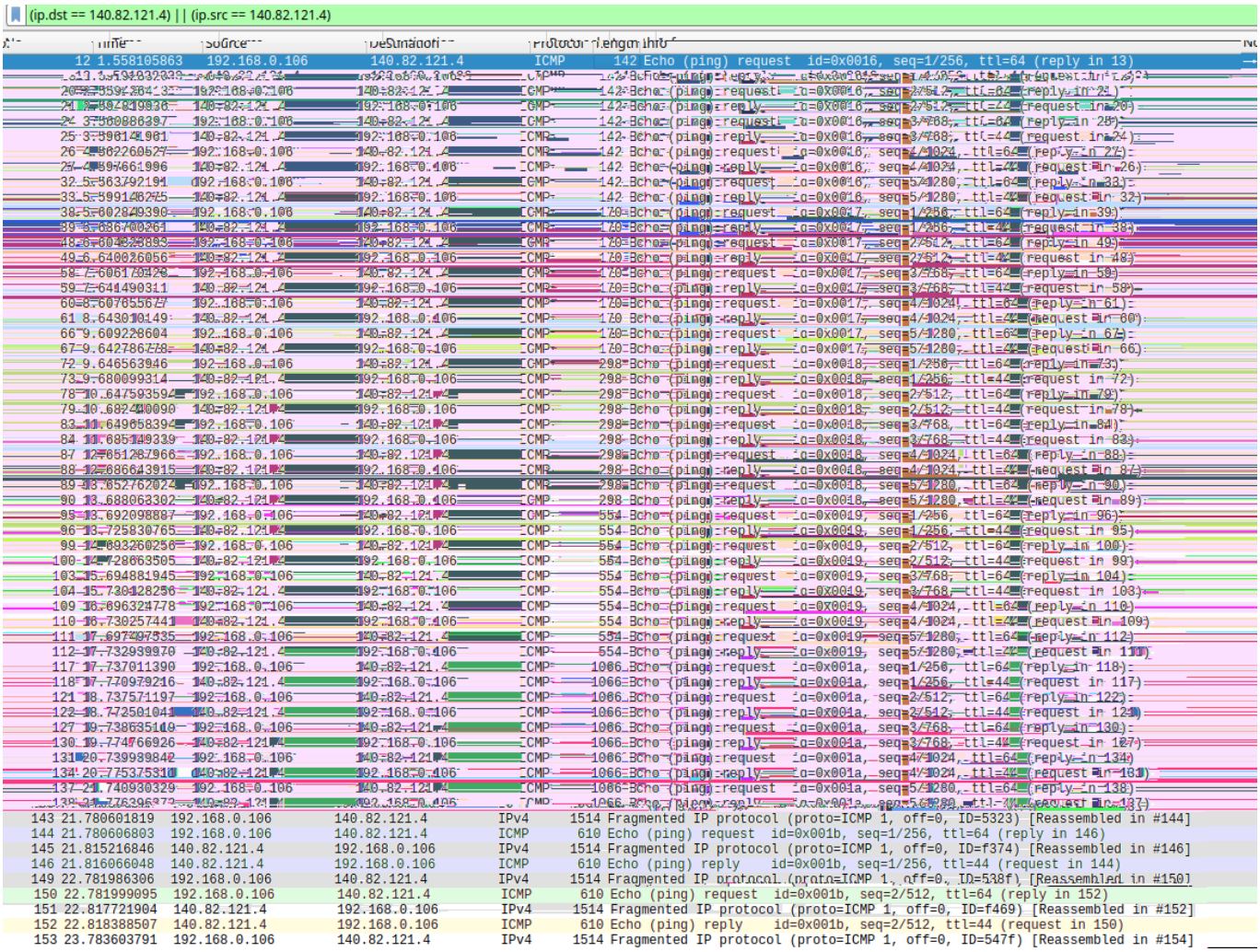
Анализ трафика утилиты `ping`

> `Ping` — утилита для проверки целостности и качества соединений в сетях на основе TCP/IP

Запустим ping. Сделаем это на 100, 128, 256, 512, 1024, 2048, 4096, 10000 байтах.

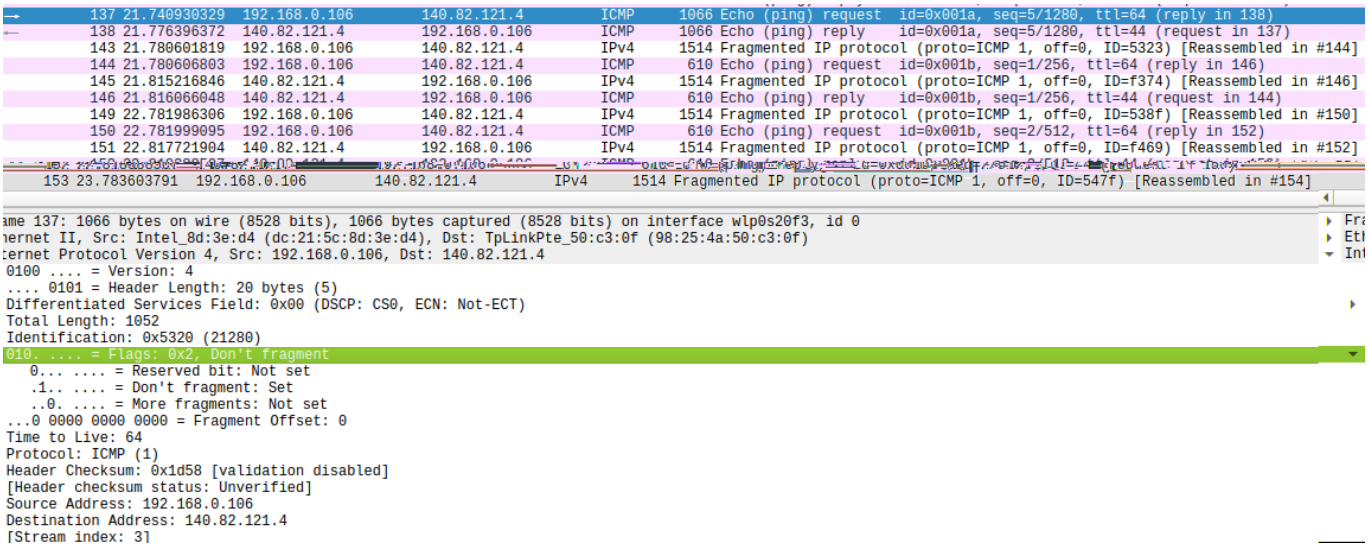
```
ping -c 5 -s 100 github.com
```

И заглянем в получившийся записанный траффик (с фильтром `(ip.dst_host == 140.82.112.4) || (ip.src_host == 140.82.112.4)`):

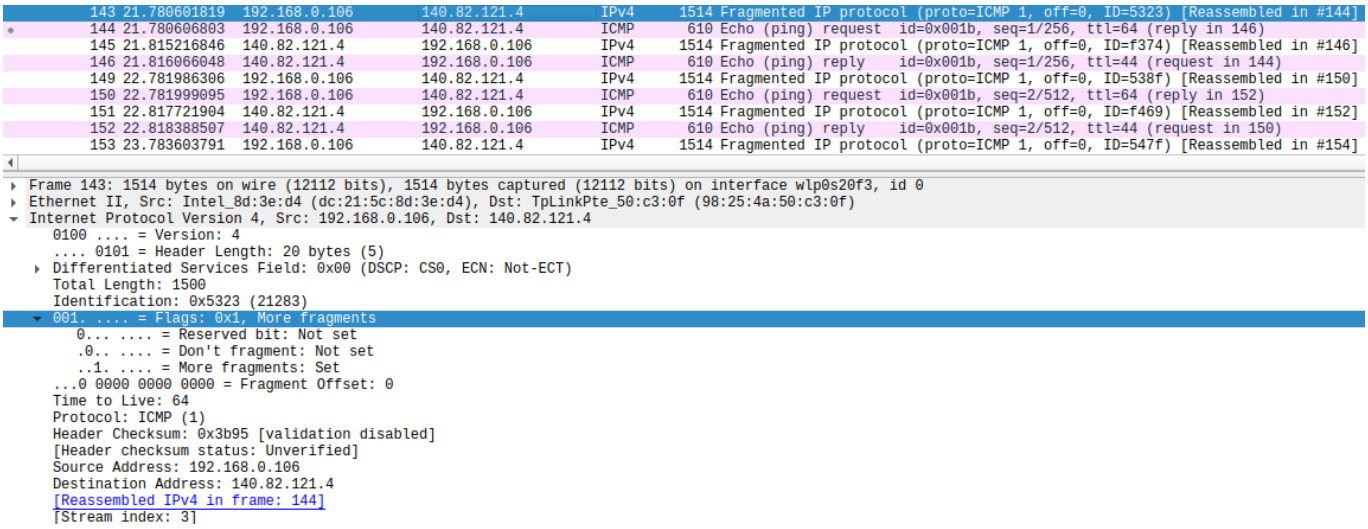


Действительно, видны ICMP-запросы и ответы.

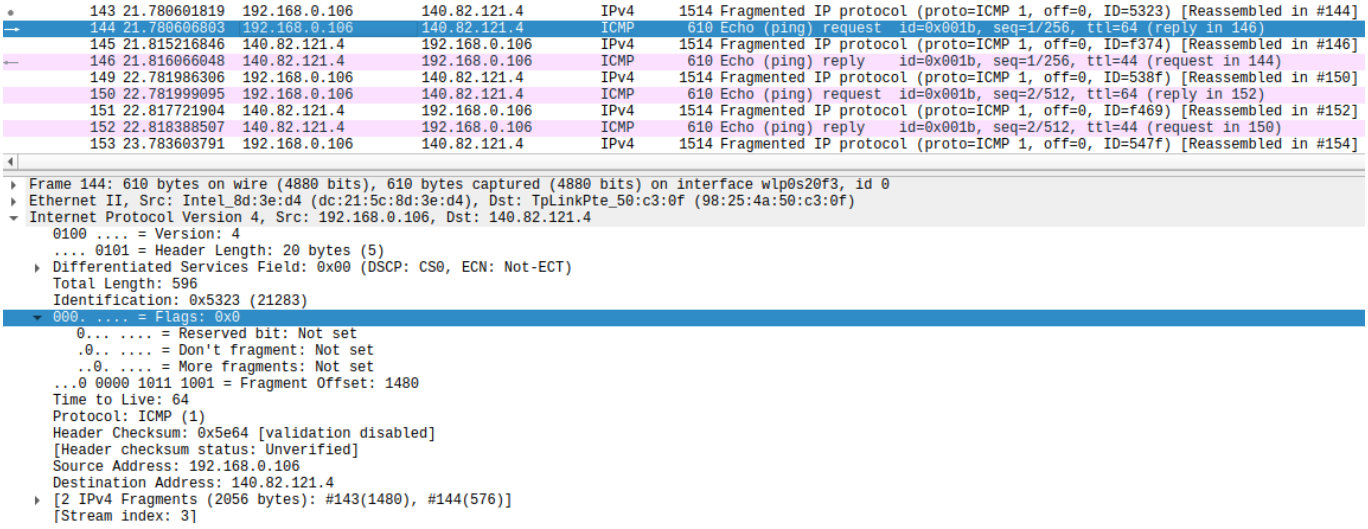
Пакет без фрагментации (DON'T FRAGMENT):



Пакет с фрагментацией (MORE FRAGMENTS):

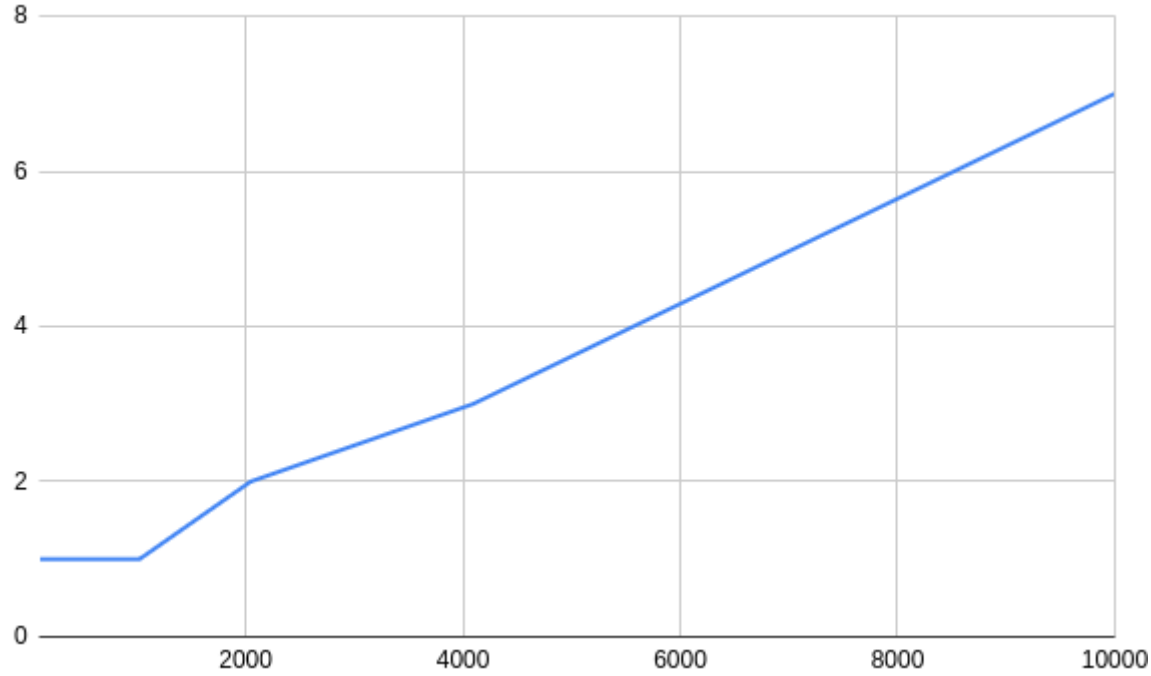


Пакет с фрагментацией, который означает конец потока (не установлены флаги):



Итого, макс. размер ICMP без разбиения (опытным путем) - 1472 байта.

Зависимость кол-ва фрагментов от размера сообщения:



Изменить TTL можно параметром `-t`:

> `-t <ttl> define time to live`

Пример:

```
$ ping -c 1 -t 2 -s 1470 github.com
PING github.com (140.82.121.3) 1470(1498) bytes of data.
From 10.40.86.1 icmp_seq=1 Time to live exceeded

--- github.com ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

В данных может лежать что угодно (главное, чтобы в ответ пришло ровно то, что отправлялось), но есть и определенные реализации, которые задают формат.

Анализ трафика утилиты tracert (traceroute)

> `Traceroute` — служебная компьютерная программа, предназначенная для определения маршрутов следования данных в сетях TCP/IP.

Может использовать ICMP, TCP, UDP, GRE по своему усмотрению. Можно настроить.

Принцип работы traceroute - каждый раз, когда роутер пересылает сообщение дальше, он уменьшает TTL (в IP) на один. Соответственно, traceroute последовательно отправляет запросы со все растущим TTL. На первый запрос (`TTL=1`) вернет ответ 1 роутер с `TTL Exceeded`. Потом отправится второй пакет, с `TTL=2`, на который с `TTL Exceeded` ответит второй роутер и т.д. Делает он так по умолчанию 3 раза (3 попытки для каждого TTL) - потому и 3 колонки времени.

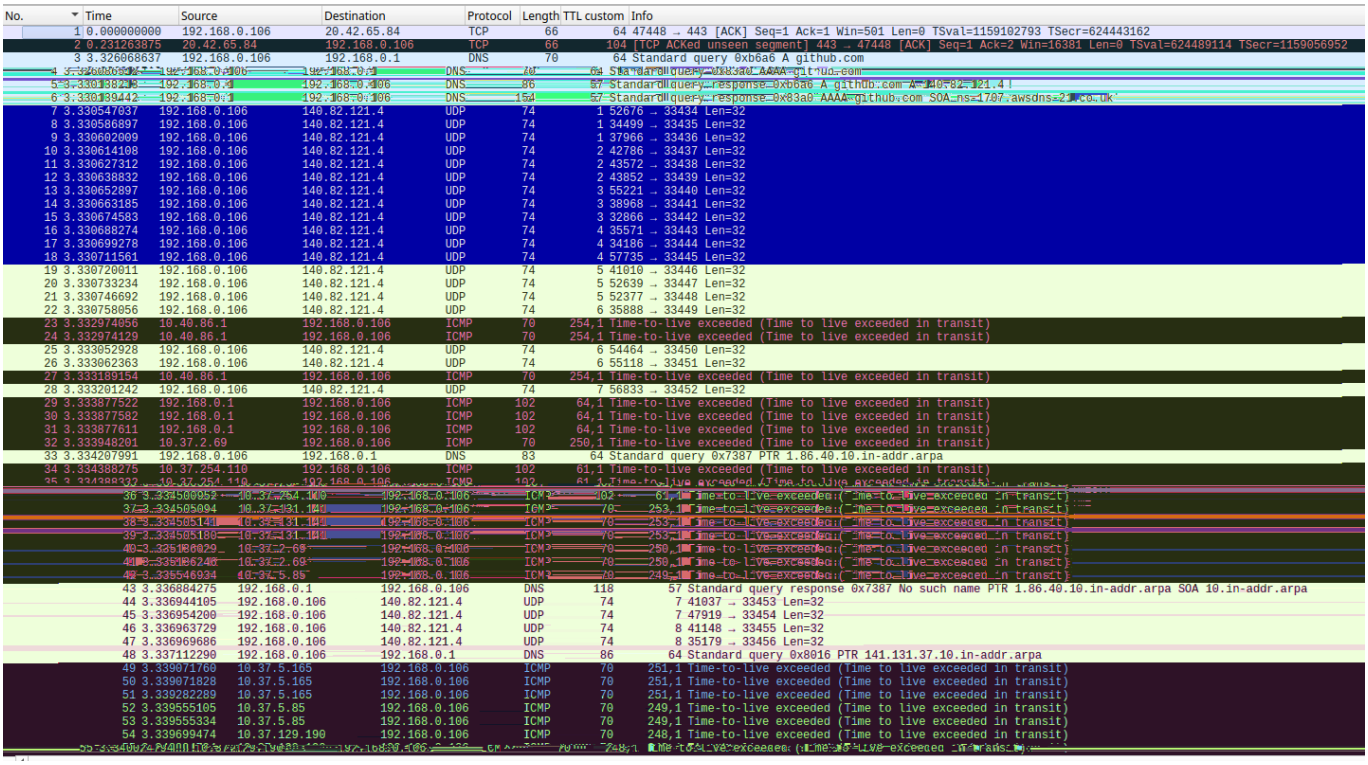
`* * *` в выводе означают, что на этом `TTL` он превысил время ожидания ответа - либо никто не ответил, либо роутер умеет распознавать такие "простуки".

Флаг `-n`:

> `-n Do not resolve IP addresses to their domain names`

```
$ sudo traceroute -n github.com
traceroute to github.com (140.82.121.3), 30 hops max, 60 byte packets
 1  192.168.0.1  2.938 ms  2.912 ms  2.905 ms
 2  10.40.86.1  2.536 ms  2.529 ms  2.522 ms
 3  10.37.131.141  3.109 ms  3.102 ms  3.096 ms
 4  10.37.254.110  3.090 ms  3.084 ms  4.797 ms
 5  10.37.5.165  4.849 ms  4.843 ms  4.837 ms
 6  10.37.2.69  3.829 ms  3.144 ms  3.128 ms
 7  10.37.5.85  3.277 ms  2.921 ms  2.880 ms
 8  10.37.129.190  3.547 ms  3.470 ms  3.460 ms
 9  194.68.128.180  13.232 ms  13.210 ms  13.203 ms
10  94.103.180.71  34.187 ms  33.320 ms  33.444 ms
11  94.103.180.3  34.251 ms  34.244 ms  33.989 ms
12  94.103.180.2  33.981 ms  34.509 ms  34.500 ms
13  * * *
14  * * *
15  * * *
16  * * *
17  94.103.180.24  33.284 ms  33.272 ms  33.758 ms
18  45.153.82.37  33.823 ms  33.833 ms 45.153.82.39  33.804 ms
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

> A hop that outputs `* * *` means that the router at that hop doesn't respond to the type of packet you were using for the traceroute.



Как видно, у последующих запросов растет TTL - для того, чтобы "достать" до все более дальнего и дальнего роутера.

Основное отличие traceroute и ping с точки зрения ICMP - что у traceroute при каждой итерации увеличивается TTL.

Если убрать `-n`, то дополнительно будет идти DNS-трафик.

Анализ HTTP-трафика

Записанный трафик:

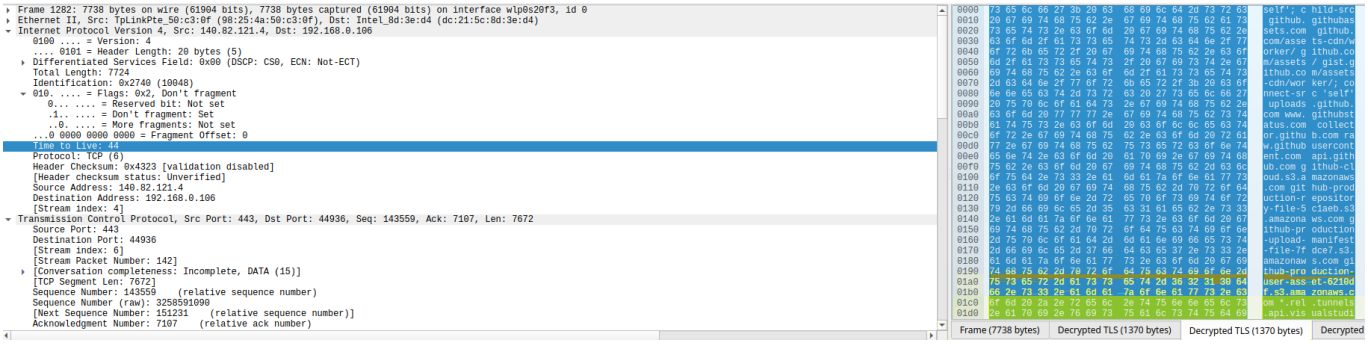
Первый запрос на страницу (записано 1279 сообщений):

```
    87 1.551919523   192.168.0.106   140.82.121.4    HTTP2    1483    64 HEADERS[3]: GET /sultanowskii, WINDOW_UPDATE[3]
    91 1.585667180   140.82.121.4    192.168.0.106   HTTP2     130    44 SETTINGS[0], SETTINGS[0]
    92 1.585737960   192.168.0.106   140.82.121.4    HTTP2      97    64 SETTINGS[0]
    94 1.946348605   140.82.121.4    192.168.0.106   HTTP2    5762    44 HEADERS[3]: 200 OK[TLS segment of a reassembled PDU]
   104 1.980538376   140.82.121.4    192.168.0.106   TLSv1.3  2914    44 DATA[3]
   113 1.981786278   140.82.121.4    192.168.0.106   TLSv1.3  2914    44 DATA[3]
   132 2.015666443   140.82.121.4    192.168.0.106   TLSv1.3  2914    44 DATA[3]
   133 2.015666515   140.82.121.4    192.168.0.106   HTTP2    2914    44 DATA[3][TLS segment of a reassembled PDU]
   142 2.016455477   140.82.121.4    192.168.0.106   TLSv1.3   841    44 DATA[3], DATA[3] (text/html)
```

Перезагрузка страницы происходит на пакете 1280. После перезагрузки было записано 1080 пакетов.

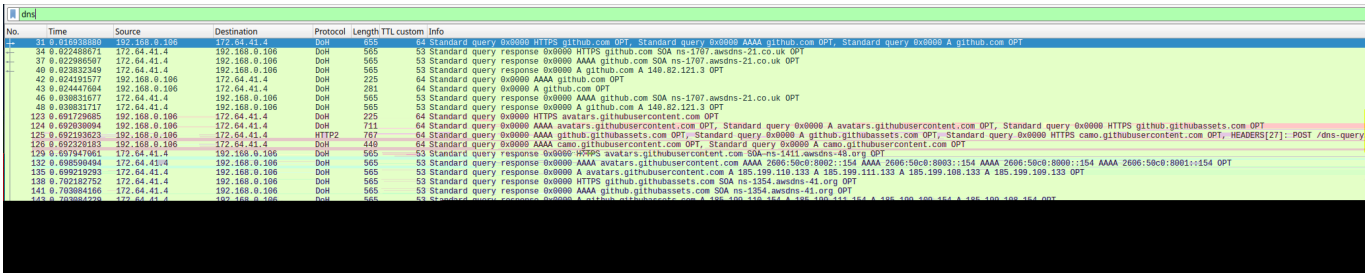
```
  1280 9.380076197   192.168.0.106   140.82.121.4    HTTP2    1471    64 HEADERS[11]: GET /sultanowskii, WINDOW_UPDATE[11]
  1282 9.933838779   140.82.121.4    192.168.0.106   HTTP2    7738    44 HEADERS[11]: 200 OK, DATA[11]
  1288 9.933920269   140.82.121.4    192.168.0.106   TLSv1.3  4338    44 [TLS segment of a reassembled PDU]
  1293 9.933945921   140.82.121.4    192.168.0.106   TLSv1.3  1260    44 DATA[11]
  1301 9.934127961   140.82.121.4    192.168.0.106   TLSv1.3  3907    44 DATA[11]
  1303 9.934128010   140.82.121.4    192.168.0.106   HTTP2    1490    44 DATA[11]
  1310 9.934636207   140.82.121.4    192.168.0.106   TLSv1.3  4108    44 DATA[11], DATA[11] (text/html)
```

Если внимательно приглядеться к ответам в обоих случаях (и с другими файлами, css, js, ...), то можно заметить, что веб-сайт не прибегает к 304 и всегда отдает страницу как "новую" - 200. Это не очень эффективно и порождает немалое количество запросов на ресурсы, которые уже были получены клиентом ранее.



Анализ DNS-трафика

172.64.41.4 - Cloudflare, CDN. Прежде, чем "дать доступ" к Github, Cloudflare анализирует трафик и только потом "пропускает", давая IP-адрес и сетевой доступ.




```
▾ HyperText Transfer Protocol 2
  ▸ Stream: DATA, Stream ID: 7, Length 468
  ▾ Domain Name System (response)
      Transaction ID: 0x0000
    ▸ Flags: 0x8180 Standard query response, No error
      Questions: 1
      Answer RRs: 1
      Authority RRs: 0
      Additional RRs: 1
    ▸ Queries
    ▾ Answers
      ▾ github.com: type A, class IN, addr 140.82.121.3
          Name: github.com
          Type: A (1) (Host Address)
          Class: IN (0x0001)
          Time to live: 41 (41 seconds)
          Data length: 4
          Address: 140.82.121.3
```

Виды DNS-записей:

- `A`: IPv4-адрес
- `AAAA`: IPv6 адрес
- `CNAME`: форвардинг с поддомена/домена на поддомен (`mail.yandex.ru`)
- `MX`: направляет письмо на email-сервер
- `TXT`: произвольные текстовые заметки
- `SOA`: информация о поддомене
- `SRV`: указание порта для специфичных сервисов
- `NS`: указание NS-сервера, ответственного за определенное имя

Анализ ARP-трафика

```
$ sudo arp -d 192.168.0.1
$
```

Как видно, при попытке "сходить в интернет", отсылается (от компьютера `192.168.0.106`) ARP-запрос (2) на поиск роутера `192.168.0.1`. После - роутер отвечает (`192.168.0.1`).

No.	Time	Source	Destination	Protocol	Length	TTL custom	Info
1	0.000000000	TpLinkPte_50:c3:0f	Broadcast	ARP	42		Who has 192.168.0.124? Tell 192.168.0.1
2	3.653372171	Intel_8d:3e:d4	Broadcast	ARP	42		Who has 192.168.0.1? Tell 192.168.0.106
3	3.656225468	TpLinkPte_50:c3:0f	Intel_8d:3e:d4	ARP	42		192.168.0.1 is at 98:25:4a:50:c3:0f
4	11.980697747	TpLinkPte_50:c3:0f	Broadcast	ARP	42		Who has 192.168.0.124? Tell 192.168.0.1
5	13.004739789	TpLinkPte_50:c3:0f	Broadcast	ARP	42		Who has 192.168.0.124? Tell 192.168.0.1
6	14.028565212	TpLinkPte_50:c3:0f	Broadcast	ARP	42		Who has 192.168.0.124? Tell 192.168.0.1

ARP-запрос:

```
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Intel_8d:3e:d4 (dc:21:5c:8d:3e:d4)
    Sender IP address: 192.168.0.106
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.0.1
```

Он содержит адрес отправителя, чтобы за меньшее число операций у других участников сети появилась информация об этом устройстве. К тому же, если бы не было этого поля, то коммутатор просто не смог бы обучаться.

Вывод

В данной лабораторной работе была произведена запись сетевого трафика с помощью утилиты Wireshark. Я записал трафик утилит ping, tracert, а также запросов в веб-браузере. Были на практике проанализированы протоколы ICMP, UDP, IP, DNS, HTTP, ARP.