МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
“Национальный исследовательский университет ИТМО”

**ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ
И КОМПЬЮТЕРНОЙ ТЕХНИКИ**

Учебно-исследовательская работа №4

«Анализ трафика компьютерных сетей с помощью утилиты Wireshark»

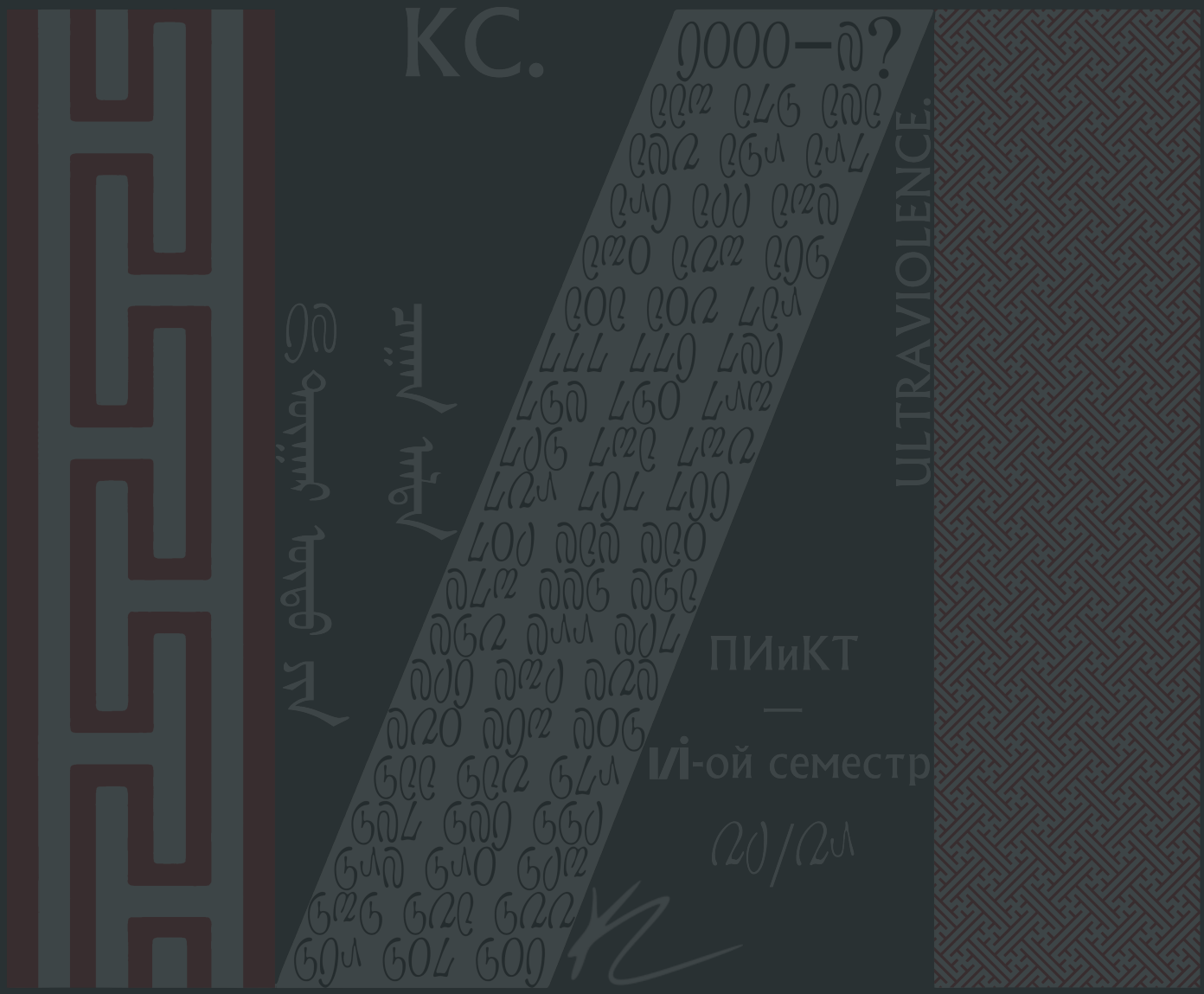
по дисциплине

“Компьютерные сети”

Работу выполнил:
Студент группы P3311
Болорболд Аригуун
Лектор:
Алиев Тауфик Измайлович, д.т.н.
Практик:
Тропченко Андрей Александрович

г. Санкт-Петербург
2025 г.

Содержимое



1 Цель и описание работы

1.1 Цель работы

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

1.2 Краткое описание работы

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

2 Выполнение работы

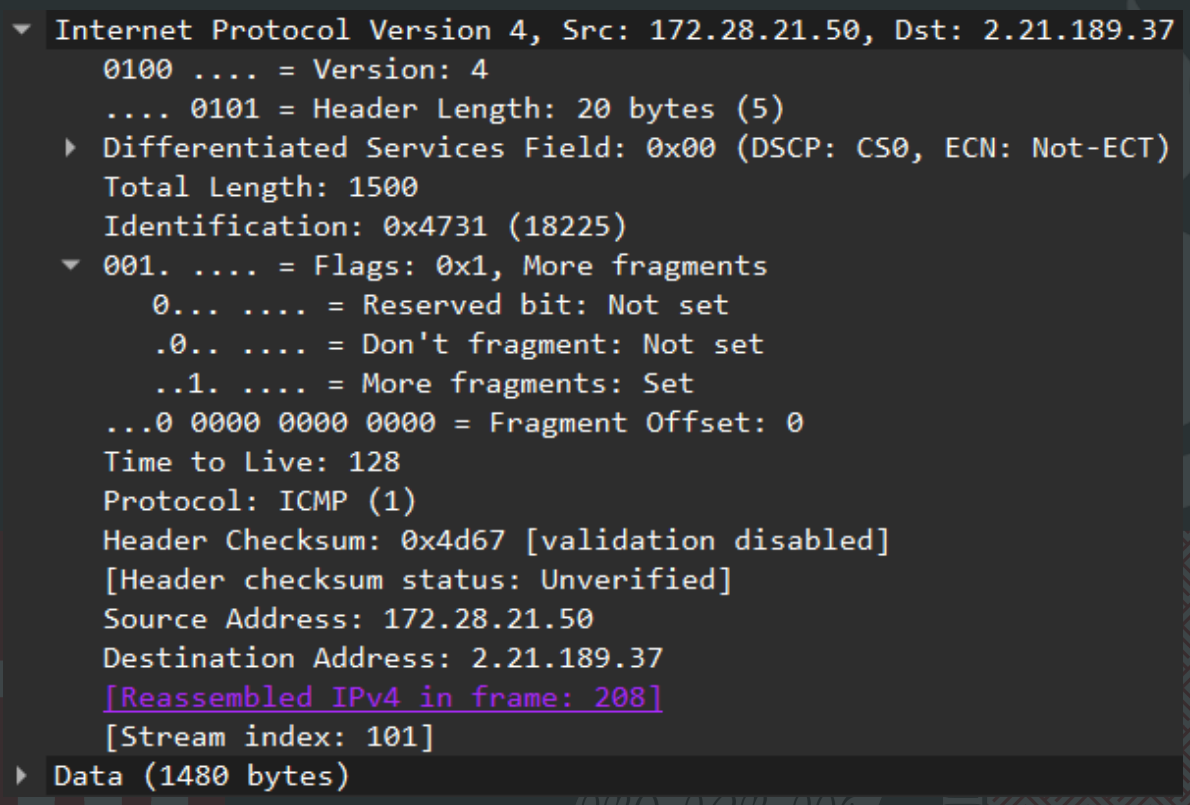
Используемый веб-сайт: https://www.nba.com

2.1 Этап 1. Анализ трафика утилиты ping.

```
PS C:\Users\Admin> ping -l 10000 www.nba.com

Pinging e8017.dscb.akamaiedge.net [2.21.189.37] with 10000 bytes of data:
Reply from 2.21.189.37: bytes=10000 time=17ms TTL=54
Reply from 2.21.189.37: bytes=10000 time=49ms TTL=54
Reply from 2.21.189.37: bytes=10000 time=31ms TTL=54
Reply from 2.21.189.37: bytes=10000 time=37ms TTL=54

Ping statistics for 2.21.189.37:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 17ms, Maximum = 49ms, Average = 33ms
PS C:\Users\Admin>
```

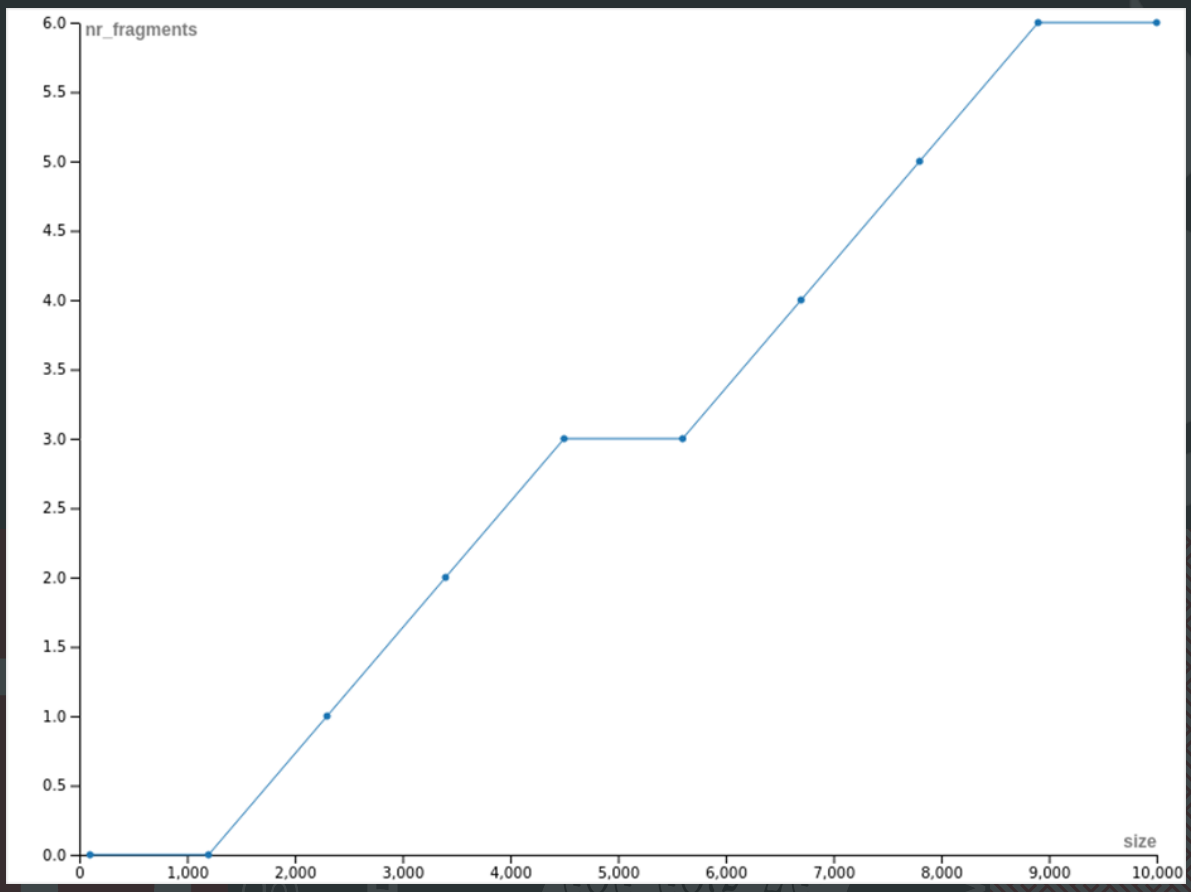
Анализ полученных пакетов

207	0.338283	172.28.21.50	2.21.189.37	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=4731) [Reassembled in #208]
208	0.338283	172.28.21.50	2.21.189.37	ICMP	562 Echo (ping) request id=0x0001, seq=48/12288, ttl=128 (reply in 228)
227	0.369099	2.21.189.37	172.28.21.50	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=7026) [Reassembled in #228]
228	0.369099	2.21.189.37	172.28.21.50	ICMP	562 Echo (ping) reply id=0x0001, seq=48/12288, ttl=54 (request in 208)
648	1.347321	172.28.21.50	2.21.189.37	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=4732) [Reassembled in #649]
649	1.347321	172.28.21.50	2.21.189.37	ICMP	562 Echo (ping) request id=0x0001, seq=50/12800, ttl=128 (reply in 657)
656	1.361732	2.21.189.37	172.28.21.50	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=7083) [Reassembled in #657]
657	1.361732	2.21.189.37	172.28.21.50	ICMP	562 Echo (ping) reply id=0x0001, seq=50/12800, ttl=54 (request in 649)
1313	2.356081	172.28.21.50	2.21.189.37	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=4733) [Reassembled in #1314]
1314	2.356081	172.28.21.50	2.21.189.37	ICMP	562 Echo (ping) request id=0x0001, seq=51/13056, ttl=128 (reply in 1326)
1325	2.373774	2.21.189.37	172.28.21.50	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=70c1) [Reassembled in #1326]
1326	2.373774	2.21.189.37	172.28.21.50	ICMP	562 Echo (ping) reply id=0x0001, seq=51/13056, ttl=54 (request in 1314)
2086	3.366515	172.28.21.50	2.21.189.37	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=4734) [Reassembled in #2087]
2087	3.366515	172.28.21.50	2.21.189.37	ICMP	562 Echo (ping) request id=0x0001, seq=52/13312, ttl=128 (reply in 2103)
2102	3.381558	2.21.189.37	172.28.21.50	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=72ed) [Reassembled in #2103]
2103	3.381558	2.21.189.37	172.28.21.50	ICMP	562 Echo (ping) reply id=0x0001, seq=52/13312, ttl=54 (request in 2087)

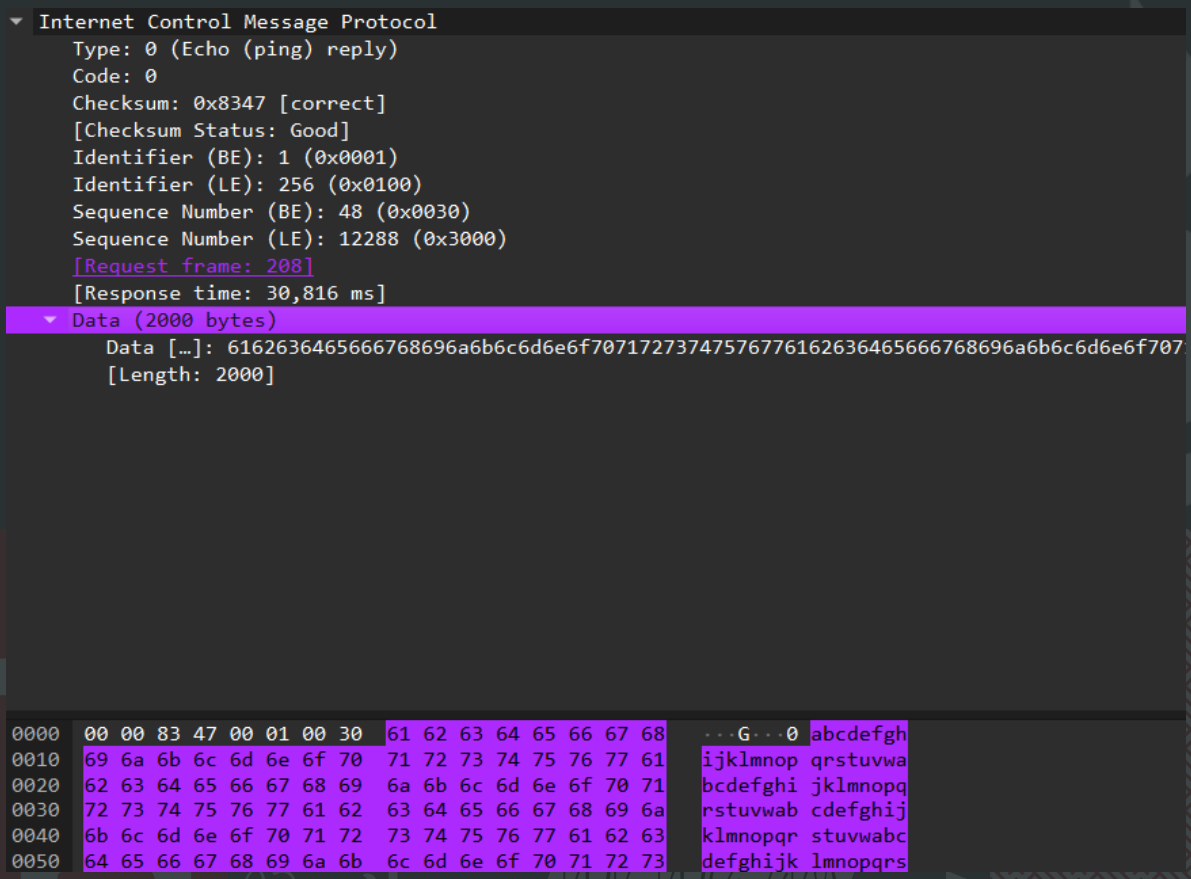


1. Да, имеет место фрагментация пакета. Указано в поле MF IPv4 пакета.
2. Для промежуточных пакетов поле MF=1, для последнего MF=0
3. Ping передает данные по 32 байта, так что фрагментации для них нет, т.е. 0 фрагментов

График

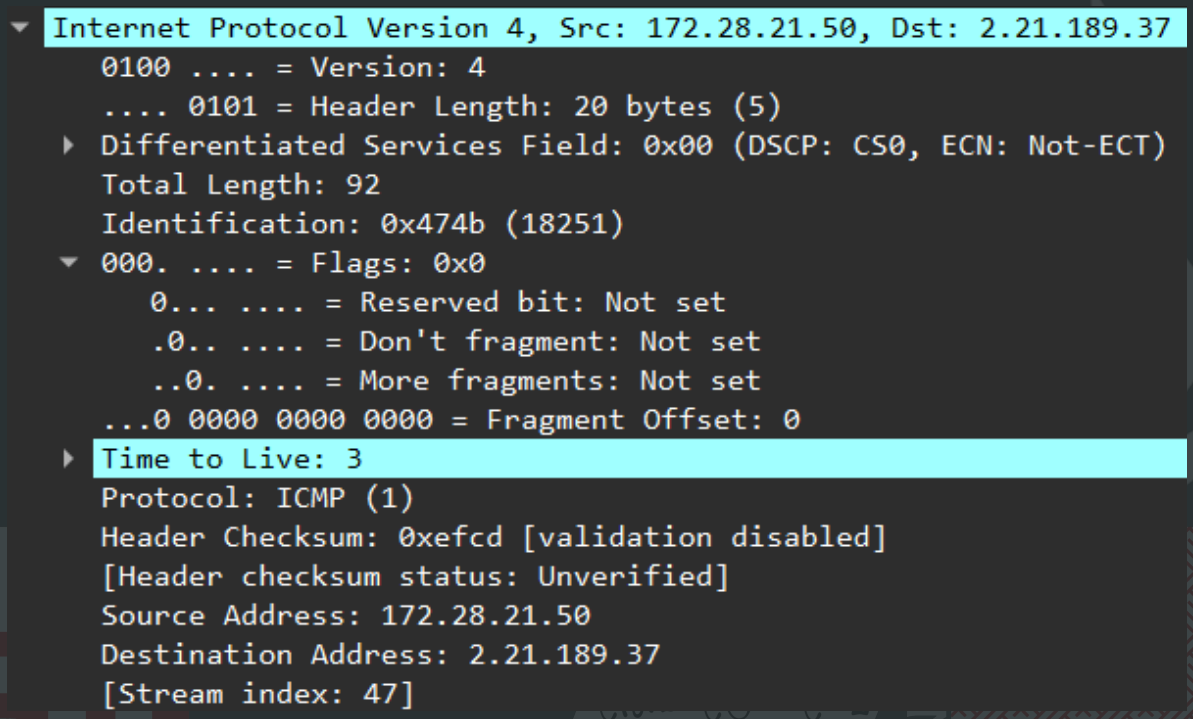


1. Использовать флаг -i <ttl>
2. Символы английского алфавита, иногда может быть отметка времени и Sequence Number ICMP-пакета.

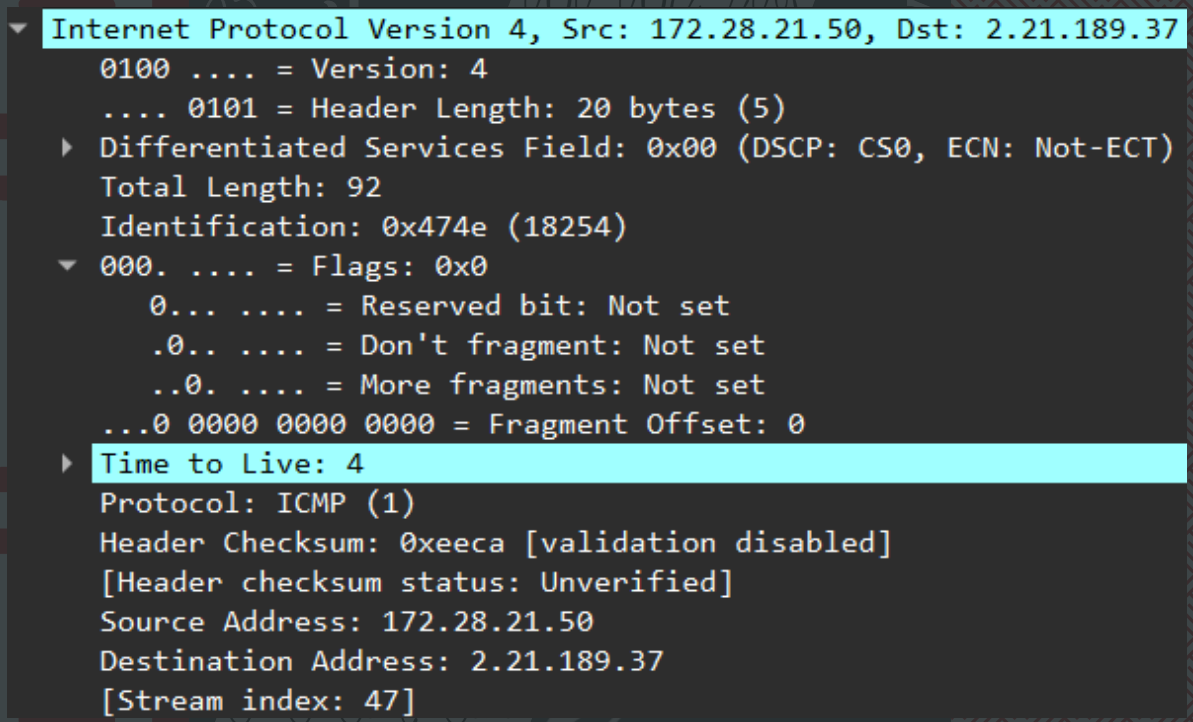


2.2 Этап 2. Анализ трафика утилиты tracert (traceroute).

```
PS C:\Users\Admin> tracert www.nba.com

Tracing route to e8017.dscb.akamaiedge.net [2.21.189.37]
over a maximum of 30 hops:

  1     2 ms     1 ms     2 ms  172.28.16.1
  2    14 ms    15 ms     2 ms  77.234.199.66
  3     *        *        *     Request timed out.
  4    12 ms     3 ms     4 ms  spb-sel-cr2.ae61-2159.rascom.as20764.net [80.64.103.132]
  5     *        *        *     Request timed out.
  6    32 ms    70 ms    23 ms  netnod-ix-ge-a-sth-1500.akamai.com [194.68.123.130]
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10    13 ms    20 ms    13 ms  a2-21-189-37.deploy.static.akamaitechnologies.com [2.21.189.37]

Trace complete.
```



1. Заголовок: 20 байт, данные: 92.
2. Увеличивается на 1 каждые 3 пакета, чтобы выявить расстояние в хопах до хоста

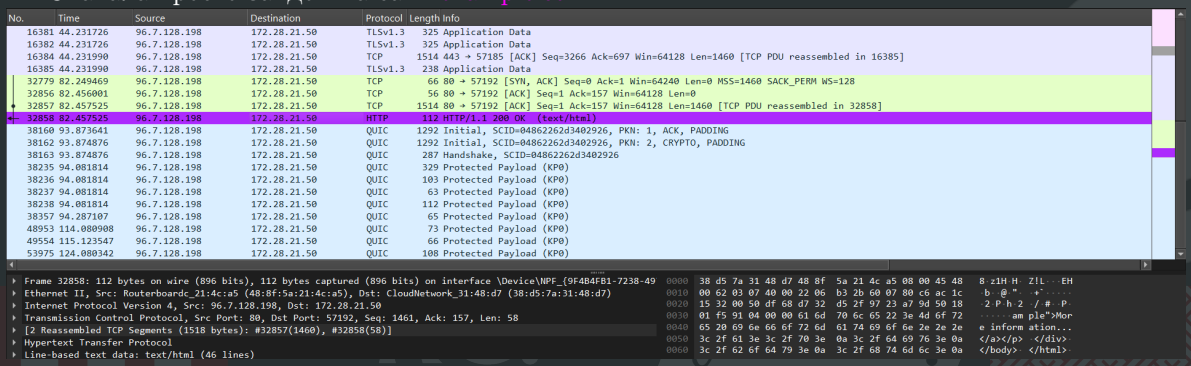


1. В ping-пакетах есть отметка времени, дефолтное значение TTL сильно выше, не меняется со следующим пакетом (в силу специфики утилиты).
2. Различные значения в поле Type.
3. Будут также слаться DNS запросы, чтобы разрешить IP-адреса в доменные имена.

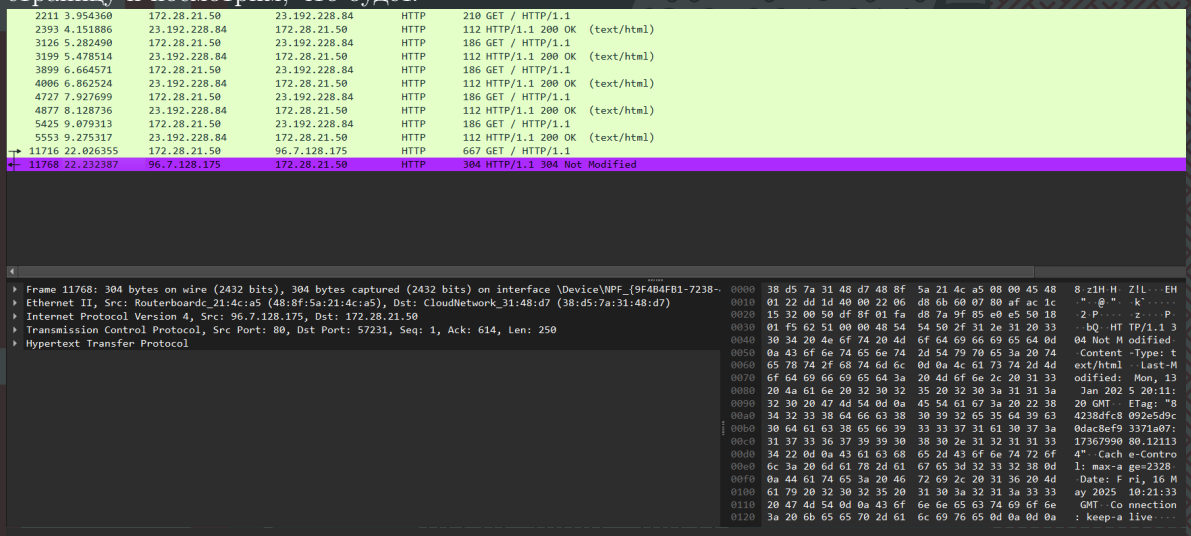
2.3 Этап 3. Анализ HTTP-трафика.

Запустим анализ в Wireshark и перейдём на сайт www.nba.com. К сожалению, сайт, который подходит нам по варианту, мало того, что не обладает возможностью принимать условные GET-запросы, так ещё и запрещает обращения по HTTP вместо HTTPS. Сколько раз не обновляй мы не можем получить ответ 304. Поэтому воспользуемся сайтом, который точно обладает такой возможностью, а именно сайтом example.com

Сначала просто зайдем на сайт example.com.



Затетим, что все отработало как надо, мы получаем ответ 200. Теперь попробуем обновить страницу и посмотрим, что будет.




```
Content-Type: text/html\r\n
Last-Modified: Mon, 13 Jan 2025 20:11:20 GMT\r\n
```

Заметим, что мы получаем совсем другую ситуацию. Здесь у нас получилось отправить условный GET-запрос. И мы получаем ответ 304 от сервера, которого можно понять по появившимся полям Last-Modified и If-Modified-Since.

2.4 Этап 4. Анализ ARP-трафика.

Для начала очистим ARP-таблицу с помощью команды:

```
netsh interface ip delete arpcache
```

Получим вот такую ARP-таблицу:

```
PS C:\WINDOWS\system32> netsh interface ip delete arpcache
Ok.

PS C:\WINDOWS\system32>
```

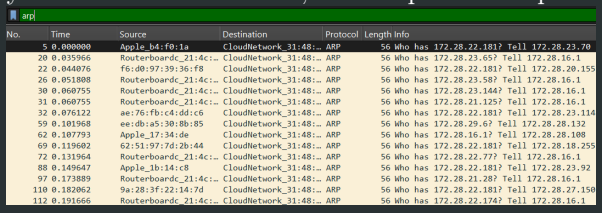
После удаления кэша браузера отправимся на сайт www.nba.com и увидим новую запись в ARP-

таблице.

```
PS C:\WINDOWS\system32> arp -a

Interface: 172.28.21.50 --- 0xc
  Internet Address      Physical Address
  172.28.16.1           48-8f-5a-21-4c-a5
  172.28.16.11          14-13-33-36-a6-55
  172.28.16.12          08-9d-f4-f4-ce-41
  172.28.19.59          bc-03-58-7e-32-54
  172.28.20.126         f0-a6-54-27-29-e9
  172.28.21.138         48-51-c5-6d-a9-b6
  172.28.21.252         f0-20-ff-dc-d0-3e
  172.28.22.49          e4-aa-ea-64-35-65
  172.28.22.60          f4-6a-dd-43-44-c7
  172.28.22.62          68-34-21-c8-98-f3
  172.28.23.43          6c-2f-80-f4-5b-64
  172.28.24.81          28-6b-35-a0-dc-8c
  172.28.24.175         6c-94-66-1c-92-ec
  172.28.26.239         6c-2f-80-20-fe-ae
  224.0.0.2             01-00-5e-00-00-02
  224.0.0.22            01-00-5e-00-00-16
  224.0.0.253           01-00-5e-00-00-fd
  239.192.152.143       01-00-5e-40-98-8f
  239.255.102.18        01-00-5e-7f-66-12
  239.255.188.58        01-00-5e-7f-bc-3a
  255.255.255.255       ff-ff-ff-ff-ff-ff
PS C:\WINDOWS\system32>
```

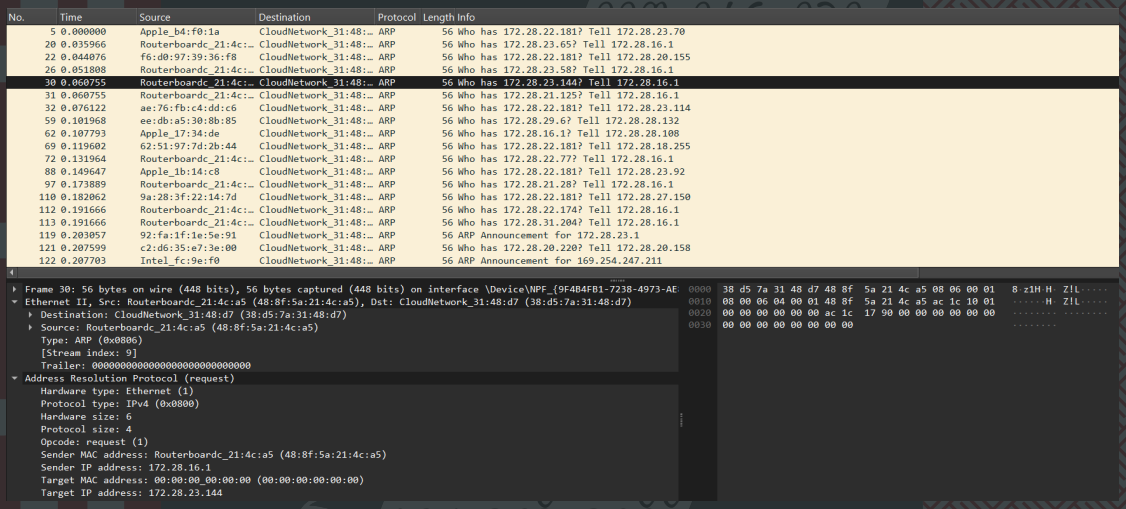
Заметим, что это вообще не похоже на IP-адрес сайта, на который мы перешли. А всё, потому что MAC-адреса используется только в локальной сети. Мы не сможем увидеть ARP-запрос, который узнаёт MAC-адрес нашего сайта, так как его и вовсе нет. Но мы видим IP-адрес 192.168.1.1 —

скорее всего это IP-адрес нашего маршрутизатора, который как раз таки и взялся в дальнейшем уже за поиск того сайта, на который мы перешли.



1. В ARP-пакетах мы увидим 2 типа MAC-адресов:

   - MAC-адрес отправителя запроса — адрес нашего компьютера. Он используется в поле Sender MAC-address.
   - MAC-адрес искомого устройства:
     - В ARP-запросе (who-has) поле Target MAC Address будет заполнено нулями, потому что он ещё известен.
     - В ARP-ответе (is-at) это будет MAC-адрес шлюза/маршрутизатора, провайдера или другого узла локальной сети, связанного с IP, на который отправляется запрос.



2. HTTP работает поверх TCP/IP в Ethernet. В Ethernet-заголовке каждого HTTP-пакета указывается:

   - MAC-адрес источника — это MAC-адрес нашего компьютера
   - MAC-адрес назначения — это обычно MAC-адрес ближайшего маршрутизатора/шлюза, через который трафик пойдет в интернет.

   MAC-адреса веб-сайта, на который мы заходим, мы не увидим, потому что MAC-адреса используются только внутри локальной сети.

3. ARP-запросы содержит IP-адреса источника, чтобы:

   - Получатель запроса (тот, чей IP-адрес запрашивается) мог записать в свою ARP-таблицу соответствие, и тем самым сократить количество ARP-запросов в будущем. Получатель понимал, кто запрашивает — это нужно для формирования ARP-запроса/ответа.

   IP-адрес источника нужен для обратной связи и корректного построения локальной маршрутизации.

3 Вывод

Во время выполнения лабораторной работы мы познакомились с работой различных протоколов передачи данных, проанализировали переданные пакеты с помощью программы Wireshark и протестировали соединения через разные утилиты.