



MainPage/DataBase/Lab 1

Университет ИТМО
Факультет ФПИ и КТ

Отчёт по лабораторной работе 1

«Компьютерные сети»

Студент: Чжоу Хунсян
Группа: Р33131
Преподаватель:

Санкт-Петербург 2024

1. Цель и краткая характеристика работы

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

Эта работа имеет целью通过使用免费的 Wireshark 实用程序分析学生计算机上的真实流量来研究协议数据块的结构。

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.

在做作业时，会观察从用户计算机到互联网以及相反方向的传输流量。使用专用的 Wireshark 实用程序可以观察各种网络协议传输的帧、数据包和数据段的结构。执行 URI 时，建议分析命令序列并确定用于组织协议中数据交换的服务数据的用途：ARP、DNS、FTP、HTTP、DHCP。

2. URL(Zhou Hongxiang)

<https://zhousfive.de/>

(217.160.0.33)

3. Выполнение работ

4.1. Анализ трафика утилиты ping

```
C:\Windows\system32\cmd.e
Microsoft Windows [版本 10.0.22621.4169]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\Tolia>ping -l 6000 zhousfive.de

正在 Ping zhousfive.de [217.160.0.33] 具有 6000 字节的数据:
来自 217.160.0.33 的回复: 字节=6000 时间=68ms TTL=51
来自 217.160.0.33 的回复: 字节=6000 时间=67ms TTL=51
来自 217.160.0.33 的回复: 字节=6000 时间=69ms TTL=51
来自 217.160.0.33 的回复: 字节=6000 时间=65ms TTL=51

217.160.0.33 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 65ms, 最长 = 69ms, 平均 = 67ms

C:\Users\Tolia>
```

正在捕获 WLAN

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(V) 无线(W) 工具(T) 帮助(H)

ip.addr == 217.160.0.33

No.	Time	Source	Destination	Protocol	Length	Info
8	2.048458	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=8101) [Reassembled in #12]
9	2.048458	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=8101) [Reassembled in #12]
10	2.048458	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=8101) [Reassembled in #12]
11	2.048458	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4440, ID=8101) [Reassembled in #12]
12	2.048458	192.168.0.14	217.160.0.33	ICMP	122	Echo (ping) request id=0x0001, seq=730/55810, ttl=128 (reply in 21)
13	2.111791	217.160.0.33	192.168.0.14	IPv4	1506	Fragmented IP protocol (proto=ICMP 1, off=0, ID=1b4c) [Reassembled in #21]
14	2.112262	217.160.0.33	192.168.0.14	IPv4	42	Fragmented IP protocol (proto=ICMP 1, off=1472, ID=1b4c) [Reassembled in #21]
15	2.115522	217.160.0.33	192.168.0.14	IPv4	1506	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=1b4c) [Reassembled in #21]
16	2.115522	217.160.0.33	192.168.0.14	IPv4	42	Fragmented IP protocol (proto=ICMP 1, off=2952, ID=1b4c) [Reassembled in #21]
17	2.118783	217.160.0.33	192.168.0.14	IPv4	1506	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=1b4c) [Reassembled in #21]
18	2.118783	217.160.0.33	192.168.0.14	IPv4	42	Fragmented IP protocol (proto=ICMP 1, off=4432, ID=1b4c) [Reassembled in #21]
19	2.121918	217.160.0.33	192.168.0.14	IPv4	1506	Fragmented IP protocol (proto=ICMP 1, off=4440, ID=1b4c) [Reassembled in #21]
20	2.121918	217.160.0.33	192.168.0.14	IPv4	42	Fragmented IP protocol (proto=ICMP 1, off=5912, ID=1b4c) [Reassembled in #21]
21	2.122968	217.160.0.33	192.168.0.14	ICMP	122	Echo (ping) reply id=0x0001, seq=730/55810, ttl=51 (request in 12)
22	3.056630	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=8102) [Reassembled in #26]
23	3.056630	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=8102) [Reassembled in #26]
24	3.056630	192.168.0.14	217.160.0.33	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=8102) [Reassembled in #26]

Frame 8: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF...
Ethernet II, Src: Intel_b2:d5:72 (58:1c:f8:b2:d5:72), Dst: HuaweiTechno_ec:d9:c5 (e0:36:76:ec:d9:c5)
Internet Protocol Version 4, Src: 192.168.0.14, Dst: 217.160.0.33
0100 = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 1500
Identification: 0x8101 (33025)
001. = Flags: 0x1, More fragments
0... = Reserved bit: Not set
0... = Don't fragment: Not set
...1. = More fragments: Set
...0 0000 0000 0000 = Fragment Offset: 0
Time to live: 128
Protocol: ICMP (1)
Header checksum: 0xf9a7 [validation disabled]
[Header checksum status: Unverified]

0000 e0 36 76 ec d9 c5 58 1c f8 b2 d5 72 08 00 45 00 6v...X...n...E...
0010 05 dc 81 01 20 00 80 01 f9 a7 c0 a8 00 0e d9 a0
0020 00 21 08 00 72 98 00 01 02 da 61 62 63 64 65 66abcde
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklmn opqrstuv
0040 77 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f wabcdeghijklmno
0050 70 71 72 73 74 75 76 77 61 62 63 64 65 66 67 68 pqrstuvw abcdefgh
0060 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 77 61 ijklmnop qrstuvw
0070 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 bcdefghi jklmnopq
0080 72 73 74 75 76 77 61 62 63 64 65 66 67 68 69 6a rstuvwab cdefghij
0090 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 77 61 62 63 klmnopqr stuvwabc
00a0 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 defghijk lmnopqrs
00b0 74 75 76 77 61 62 63 64 65 66 67 68 69 6a 6b 6c tuvabcde fghijkl
00c0 6d 6e 6f 70 71 72 73 74 75 76 77 61 62 63 64 65 mnopqrst uvwabcde
00d0 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 fghijklm nopqrstu
00e0 76 77 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e wabcdeghijklm
00f0 6f 70 71 72 73 74 75 76 67 68 69 6a 6b 6c 6d 6e opqrstuv wabcdegh
0100 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 77 hijklmno pqrstuvw

分箱: 26202 · Displayed: 56 (0.2%) 配置: Default

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

указывает?

Да, фрагментация исходного пакета имеет место. На втором скриншоте из Wireshark видно, что передаваемый ICMP-пакет разделён на несколько фрагментов. В Wireshark это отображено как "Fragmented IP protocol". Поля, указывающие на фрагментацию, включают:

- Поле "Flags" в заголовке IP-пакета: Бит "More Fragments" (MF) установлен (значение равно 1), что указывает, что пакет был фрагментирован.
- Поле "Fragment Offset": Указывает смещение текущего фрагмента относительно начала исходного пакета.

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

Информация, которая указывает, является ли фрагмент последним или промежуточным, находится в поле "Flags" в заголовке IP-пакета:

- Если бит "More Fragments" (MF) установлен (равен 1), это означает, что это промежуточный фрагмент, и за ним последуют другие фрагменты.
- Если бит MF сброшен (равен 0), это означает, что это последний фрагмент пакета.

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

На скриншоте видно, что при передаче `ping -l 6000` пакет был разделён на 5 фрагментов. Wireshark показывает несколько IP-пакетов с одинаковым значением идентификатора (ID) и разным значением смещения фрагмента ("Fragment Offset").

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

Размер пакета	количество фрагментов
1000	1
2000	2
3000	3
4000	3

Размер пакета	количество фрагментов
5000	4
6000	5
7000	5
8000	6
9000	7
10000	7

5. Как изменить поле TTL с помощью утилиты ping?

Чтобы изменить значение TTL при использовании утилиты `ping`, используйте следующие команды:

- В Windows: `ping -i [значение TTL] [адрес]`
- В Linux/Unix: `ping -t [значение TTL] [адрес]`

6. **Что содержится в поле данных ping-пакета?**

Поле данных ping-пакета содержит дополнительные данные, отправленные вместе с ICMP-запросом. По умолчанию это могут быть псевдослучайные байты или повторяющийся шаблон данных. На втором скриншоте видно содержимое поля данных в шестнадцатеричном и текстовом представлении.

4.2. Анализ трафика утилиты tracert (traceroute)

```
C:\Windows\system32\cmd.e: X + v
    最短 = 72ms, 最长 = 76ms, 平均 = 74ms

C:\Users\Tolia>ping -l 10000 zhousfive.de

正在 Ping zhousfive.de [217.160.0.33] 具有 10000 字节的数据:
来自 217.160.0.33 的回复: 字节=10000 时间=78ms TTL=51
来自 217.160.0.33 的回复: 字节=10000 时间=78ms TTL=51
来自 217.160.0.33 的回复: 字节=10000 时间=82ms TTL=51
来自 217.160.0.33 的回复: 字节=10000 时间=74ms TTL=51

217.160.0.33 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 74ms, 最长 = 82ms, 平均 = 78ms

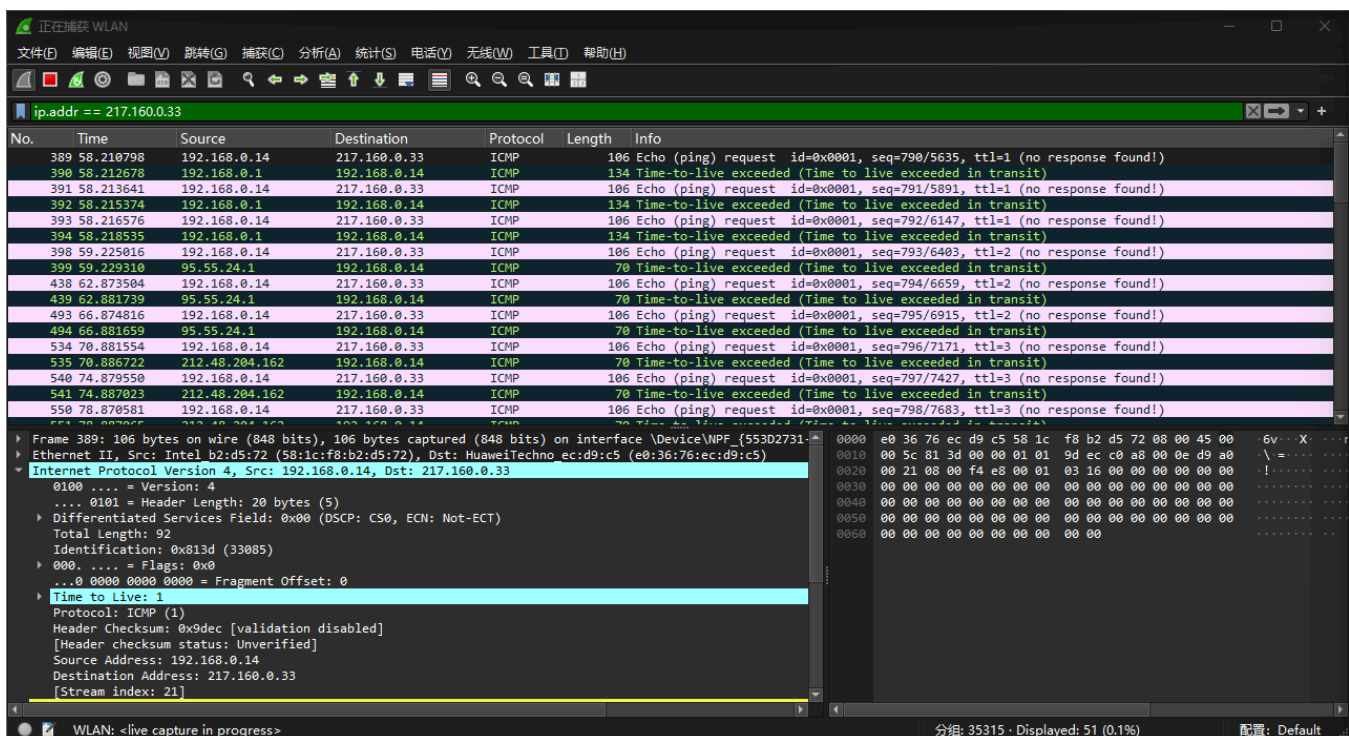
C:\Users\Tolia>tracert -d zhousfive.de

通过最多 30 个跃点跟踪
到 zhousfive.de [217.160.0.33] 的路由:

 1      1 ms      1 ms      1 ms    192.168.0.1
 2      *        *        *        请求超时。
 3      *        *        *        请求超时。
 4      *        *        *        请求超时。
 5      *        *        *        请求超时。
 6      *        *        *        请求超时。
 7      *        *        *        请求超时。
 8      *        *        *        请求超时。
 9     55 ms     54 ms     53 ms    217.160.0.33

跟踪完成。

C:\Users\Tolia>
```



正在捕获 WLAN

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(M) 无线(W) 工具(T) 帮助(H)

ip.addr == 217.160.0.33

No.	Time	Source	Destination	Protocol	Length	Info
389	58.210798	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=790/5635, ttl=1 (no response found!)
390	58.212678	192.168.0.14	217.160.0.33	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
391	58.213641	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=791/5891, ttl=1 (no response found!)
392	58.215374	192.168.0.14	217.160.0.33	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
393	58.216576	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=792/6147, ttl=1 (no response found!)
394	58.218535	192.168.0.14	217.160.0.33	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
398	59.225016	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=793/6403, ttl=2 (no response found!)
399	59.229310	95.55.24.1	192.168.0.14	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
438	62.873504	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=794/6659, ttl=2 (no response found!)
439	62.881739	95.55.24.1	192.168.0.14	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
493	66.874816	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=795/6915, ttl=2 (no response found!)
494	66.881659	95.55.24.1	192.168.0.14	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
534	70.881554	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=796/7171, ttl=3 (no response found!)
535	70.886722	212.48.204.162	192.168.0.14	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
540	74.879550	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=797/7427, ttl=3 (no response found!)
541	74.887023	212.48.204.162	192.168.0.14	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
550	78.870581	192.168.0.14	217.160.0.33	ICMP	106	Echo (ping) request id=0x0001, seq=798/7683, ttl=3 (no response found!)

Frame 389: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface \Device\NPF... (553D2731-...)

Ethernet II, Src: Intel_b2:d5:72 (58:1c:f8:b2:d5:72), Dst: HuaweiTechno_ec:d9:c5 (e0:36:76:ec:d9:c5)

Internet Protocol Version 4, Src: 192.168.0.14, Dst: 217.160.0.33

0100 = Version: 4

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

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

Total Length: 92

Identification: 0x813d (33085)

0000 = Flags: 0x0

...0 0000 0000 0000 = Fragment Offset: 0

Time to Live: 1

Protocol: ICMP (1)

Header Checksum: 0x9dec [validation disabled]

[Header checksum status: Unverified]

Source Address: 192.168.0.14

Destination Address: 217.160.0.33

[Stream index: 21]

WLAN: <live capture in progress>

分组: 35315 · Displayed: 51 (0.1%)

配置: Default

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

955	142.882569	192.168.0.14	217.168.0.33	ICMP	106 Echo (ping) request	id=0x0001, seq=814/11779, ttl=9 (reply in 956)
956	142.937674	217.168.0.33	192.168.0.14	ICMP	106 Echo (ping) reply	id=0x0001, seq=814/11779, ttl=51 (request in 955)
957	142.939138	192.168.0.14	217.168.0.33	ICMP	106 Echo (ping) request	id=0x0001, seq=815/12035, ttl=9 (reply in 958)
958	142.993576	217.168.0.33	192.168.0.14	ICMP	106 Echo (ping) reply	id=0x0001, seq=815/12035, ttl=51 (request in 957)
959	142.994838	192.168.0.14	217.168.0.33	ICMP	106 Echo (ping) request	id=0x0001, seq=816/12291, ttl=9 (reply in 960)
960	143.048586	217.168.0.33	192.168.0.14	ICMP	106 Echo (ping) reply	id=0x0001, seq=816/12291, ttl=51 (request in 959)

Frame 960: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface \Device\NPF_{553D2731-8B1C-F8B2-D572-E03676EC:D9:C5}, Dst: Intel_b2:d5:72 (58:1c:f8:b2:d5:72)

Internet Protocol Version 4, Src: 217.168.0.33, Dst: 192.168.0.14

0100 = Version: 4

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

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

Total Length: 92

Identification: 0xbb02 (47874)

000. = Flags: 0x0

...0 0000 0000 0000 = Fragment Offset: 0

Time to Live: 51

0000 58 1c f8 b2 d5 72 e0 36 76 ec d9 c5 08 00 45 00 X.....6 v

0010 00 5c bb 02 00 00 33 01 32 27 d9 a0 00 21 c0 a83: 2'

0020 00 00 00 00 00 00 fc ce 00 01 03 30 00 00 00 000

0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 000

0040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 000

0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 000

0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 000

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

The screenshot shows two ICMP Echo (ping) packets in Wireshark. The top packet is a request from 217.168.0.33 to 192.168.0.14. The bottom packet is a reply from 192.168.0.14 to 217.168.0.33. The interface is '正在捕获 WLAN'.

Top Packet (Request):

- No. 80
- Time: 7.174048
- Source: 217.168.0.33
- Destination: 192.168.0.14
- Protocol: ICMP
- Length: 1162
- Info: Echo (ping) request id=0x0001, seq=1131/27396, ttl=128 (reply in 93)

Bottom Packet (Reply):

- No. 79
- Time: 6.166930
- Source: 95.55.24.1
- Destination: 192.168.0.14
- Protocol: ICMP
- Length: 70
- Info: Time-to-live exceeded (Time to live exceeded in transit)

- Утилита ping отправляет ICMP Echo Request пакеты на конкретный хост и ожидает ICMP Echo Reply.
- Утилита tracert отправляет серию ICMP Echo Request пакетов с

увеличивающимся TTL для отслеживания маршрута к удаленному хосту и в поле данных содержат нули

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

ICMP Echo Reply: Это стандартный ответ на ICMP Echo-запрос. Когда хост получает Echo-запрос (например, от команды Ping), он отвечает Echo-ответом, указывая, что он доступен.

ICMP Error (Time Exceeded): Этот пакет генерируется маршрутизатором, когда TTL пакета достигает 0. Он сообщает отправителю, что пакет был уничтожен из-за превышения времени жизни. Tracert использует эти пакеты для определения промежуточных маршрутизаторов.

Зачем нужны оба: Echo Reply используется для подтверждения доступности узла. Time Exceeded пакеты нужны для диагностики сети, они позволяют узнать о промежуточных маршрутизаторах на пути к целевому хосту.

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

Если убрать ключ -d: Tracert попытается выполнить обратное разрешение IP-адресов в DNS-имена (то есть преобразовать IP-адреса маршрутизаторов на пути в имена хостов).

Дополнительный трафик: При этом генерируются дополнительные DNS-запросы для каждого IP-адреса, который tracert обнаруживает на пути к конечной точке. Это приводит к небольшому увеличению времени выполнения команды и дополнительной нагрузке на DNS-сервер.

4.3. Анализ HTTP-трафика

Потому что сайт у варианта не использует HTTP, поэтому мы здесь используем другой сайт: <http://www.moe.gov.cn/>

正在捕获 WLAN

文件(F) 编辑(E) 视图(V) 捕获(C) 分析(A) 统计(S) 电话(V) 无线(W) 工具(I) 帮助(H)

http

No.	Time	Source	Destination	Protocol	Length	Info
431	5.360421	192.168.0.14	211.97.84.77	HTTP	576	GET / HTTP/1.1
473	5.582401	211.97.84.77	192.168.0.14	HTTP	800	HTTP/1.1 200 OK (text/html)
483	5.609615	192.168.0.14	211.97.84.77	HTTP	558	GET /images/moe.css HTTP/1.1
484	5.611079	192.168.0.14	211.97.84.77	HTTP	561	GET /images/moe.nav.css HTTP/1.1
498	5.801965	192.168.0.14	211.97.84.77	HTTP	570	GET /images/moe.module.list.css HTTP/1.1
502	5.807116	192.168.0.14	211.97.84.77	HTTP	564	GET /images/moe.index.css HTTP/1.1
503	5.807529	211.97.84.77	192.168.0.14	HTTP	420	HTTP/1.1 304 Not Modified
510	5.808681	192.168.0.14	211.97.84.77	HTTP	554	GET /images/moe.doMobile.min.js HTTP/1.1
514	5.813350	192.168.0.14	211.97.84.77	HTTP	550	GET /images/jquery.min.js HTTP/1.1
517	5.813818	211.97.84.77	192.168.0.14	HTTP	1187	HTTP/1.1 200 OK (text/css)
519	5.814586	192.168.0.14	211.97.84.77	HTTP	540	GET /images/h5.js HTTP/1.1
526	5.989836	211.97.84.77	192.168.0.14	HTTP	443	HTTP/1.1 304 Not Modified
537	5.999266	211.97.84.77	192.168.0.14	HTTP	1173	HTTP/1.1 200 OK (text/css)
541	5.999664	192.168.0.14	211.97.84.77	HTTP	549	GET /images/jquery.002.js HTTP/1.1
542	5.999733	192.168.0.14	211.97.84.77	HTTP	560	GET /images/ldangerous.swiper.min.js HTTP/1.1
544	6.003271	211.97.84.77	192.168.0.14	HTTP	433	HTTP/1.1 304 Not Modified
550	6.010734	211.97.84.77	192.168.0.14	HTTP	436	HTTP/1.1 304 Not Modified
554	6.011762	211.97.84.77	192.168.0.14	HTTP	433	HTTP/1.1 304 Not Modified

Frame 519: 540 bytes on wire (4320 bits), 540 bytes captured (4320 bits) on interface \Device\NPF_{553D2731-...}

Ethernet II, Src: Intel_b2:d5:72 (58:1c:f8:b2:d5:72), Dst: HuaweiTechno_ec:d9:c5 (e0:36:76:ec:d9:c5)

Internet Protocol Version 4, Src: 192.168.0.14, Dst: 211.97.84.77

0100 = Version: 4

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

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

Total Length: 526

Identification: 0x963b (38459)

010. = Flags: 0x2, Don't fragment

...0 0000 0000 0000 = Fragment Offset: 0

Time to Live: 128

Protocol: TCP (6)

Header Checksum: 0x7a49 [validation disabled]

[Header checksum status: Unverified]

Source Address: 192.168.0.14

Destination Address: 211.97.84.77

[Stream index: 8]

Transmission Control Protocol, Src Port: 12614, Dst Port: 80, Seq: 1027, Ack: 25379, Len: 486

Hypertext Transfer Protocol

GET /images/h5.js HTTP/1.1\r\n

Request Method: GET

Request URI: /images/h5.js

Request Version: HTTP/1.1

Host: www.moe.gov.cn\r\n

Connection: keep-alive\r\n

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6613.122 (x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6613.122 Safari/537.36\r\n

Accept: */*\r\n

Referer: http://www.moe.gov.cn/\r\n

Accept-Encoding: gzip, deflate\r\n

Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,ru;q=0.7\r\n

Cookie: wdcid=6a8962c0bf761465; wdses=5c16b0f25ea588c4; wdlast=1726393987\r\n

Cookie pair: wdcid=6a8962c0bf761465

Cookie pair: wdses=5c16b0f25ea588c4

Cookie pair: wdlast=1726393987

If-None-Match: "66e258bf-c5f"\r\n

If-Modified-Since: Thu, 12 Sep 2024 02:58:07 GMT\r\n

\r\n

[Response in frame: 554]

[Full request URI: http://www.moe.gov.cn/images/h5.js]

0000 e0 36 76 ec d9 c5 58 1c f8 b2 d5 72 08 00 45 00 6v...X...r

0010 02 0e 96 3b 40 00 80 06 7a 49 c0 a8 00 0e d3 61 ...;@...zI..

0020 54 4d 31 46 00 50 b2 cc 3e 6b 66 80 6c 03 50 18 TM1F P...>kf

0030 01 03 fa c4 00 00 47 45 54 20 2f 69 6d 61 67 65GE T /j

0040 73 2f 68 35 2e 6a 73 20 48 54 50 2f 31 2e 31 s/h5.js HTTP

0050 0d 0a 48 6f 73 74 3a 20 77 77 77 2e 6d 6f 65 2e ..Host: www.

0060 67 6f 76 2e 63 6e 0d 0a 43 6f 6e 6e 65 63 74 69 gov.cn...Conr

0070 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a on: keep -al

0080 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 User-Age nt:

0090 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 lla/5.0 (Wir

00a0 20 4e 54 20 31 30 2e 30 3b 20 57 69 6e 36 34 3b NT 10.0 ; W

00b0 20 78 36 34 29 20 41 70 70 6c 65 57 65 62 4b 69 x64) Ap ple

00c0 74 2f 35 33 37 2e 33 36 20 28 4b 48 54 4d 4c 2c t/537.36 (K

00d0 20 6c 69 6b 65 20 47 65 63 6b 6f 29 20 43 68 72 like Ge cko

00e0 6f 6d 65 2f 31 32 38 2e 30 2e 30 2e 30 20 53 61 ome/128. 0.0

00f0 66 61 72 69 2f 35 33 37 2e 33 36 0d 0a 41 63 63 fari/537 .36

0100 65 70 74 3a 20 2a 2f 2a 0d 0a 52 65 66 65 72 65 ept: /*...Re

0110 72 3a 20 68 74 74 70 3a 2f 2f 77 77 77 2e 6d 6f ri: http: //w

0120 65 2e 67 6f 76 2e 63 6e 2f 0d 0a 41 63 63 65 70 t-Encodi ng:

0130 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a 69 70 e.gov.cn /.../

0140 2c 20 64 65 66 6c 61 74 65 0d 0a 41 63 63 65 70 , deflate e.../

0150 74 2d 4c 61 6e 67 75 61 67 65 3a 20 7a 68 2d 43 t-Langua ge:

0160 4e 2c 7a 68 3b 71 3d 30 2e 39 2c 65 6e 3b 71 3d N,zh;q=0 .9,e

0170 30 2e 38 2c 72 75 3b 71 3d 30 2e 37 0d 0a 43 6f 0.8,ru;q =0.7

0180 6f 6b 69 65 3a 20 77 64 63 69 64 3d 36 61 38 39 okie: wd cid

0190 36 32 63 64 62 66 37 36 31 34 36 35 3b 20 77 64 62c0bf76 1465

01a0 73 65 73 3d 35 63 31 36 62 30 66 32 35 65 61 35 ses=5c16 b0f

01b0 38 38 63 34 3b 20 77 64 6c 61 73 74 3d 31 37 32 88c4; wd last

01c0 36 33 39 33 39 38 37 0d 0a 49 66 2d 4e 6f 6e 65 6393987- If-

01d0 2d 4d 61 74 63 68 3a 20 22 36 36 65 32 35 38 62 -Match: "66e

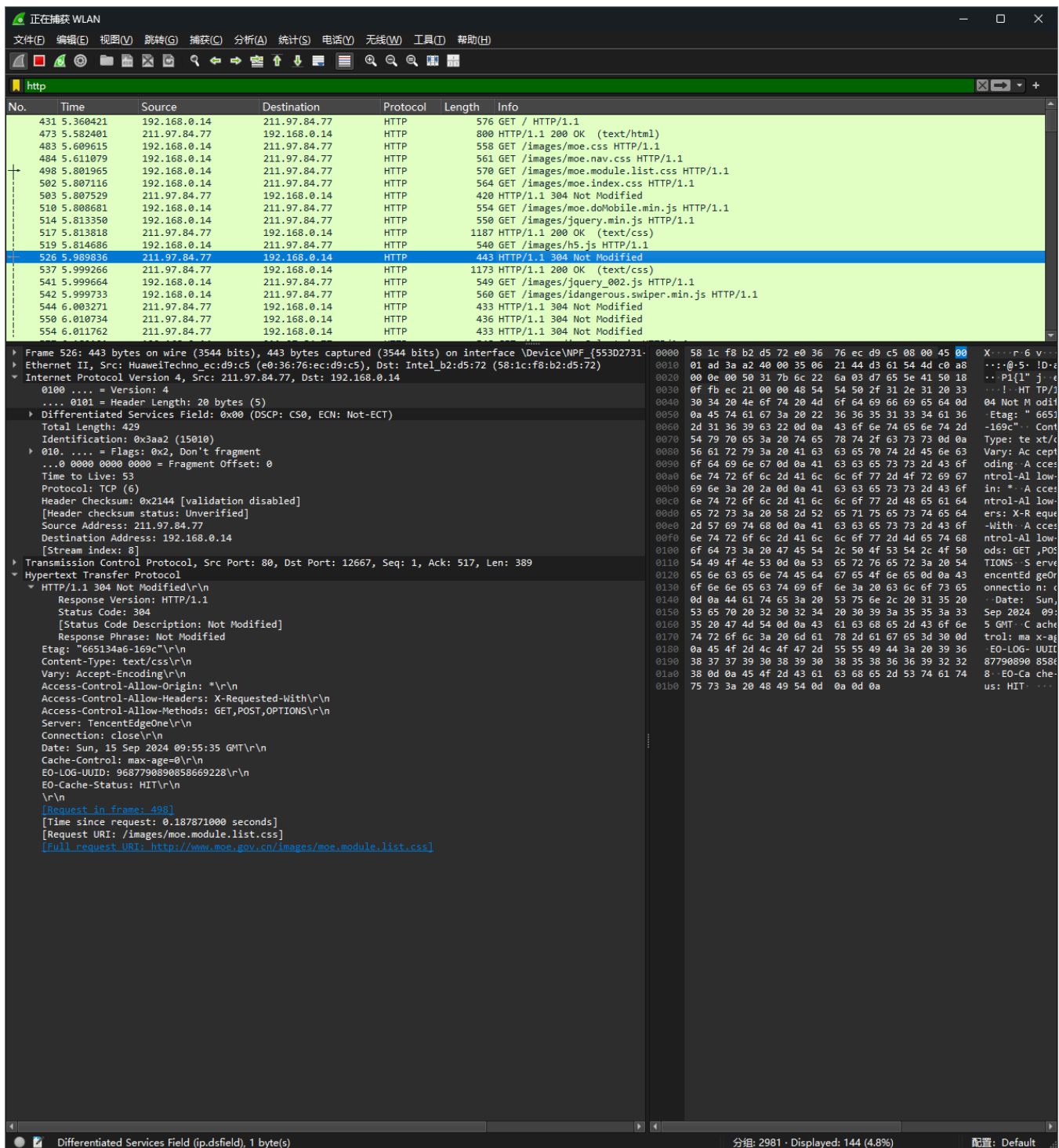
01e0 66 2d 63 35 66 22 0d 0a 49 66 2d 4d 6f 64 69 66 f-c5f"... If-

01f0 69 65 64 2d 53 69 6e 63 65 3a 20 54 68 75 2c 20 ied-Sinc e: 1

0200 31 32 20 53 65 70 20 32 30 32 3a 20 30 32 3a 35 12 Sep 2 024

0210 38 3a 30 37 20 47 4d 54 0d 0a 0d 0a 8:07 GMT

Differentiated Services Field (ip.dsfield), 1 byte(s) 分组: 2863 · Displayed: 144 (5.0%) 配置: Default



По результатам анализа трассы можно увидеть, как HTTP протокол передает содержимое страницы во время первичного и вторичного (обновленного) запроса.

- **Первичное посещение страницы**

- Отправка GET-запроса: Браузер отправляет HTTP GET-запрос к серверу, чтобы получить содержимое страницы. Запрос содержит

информацию о запрашиваемом ресурсе

- ii. Получение ответа от сервера: Сервер обрабатывает запрос и отправляет HTTP-ответ, который содержит код состояния, заголовки и тело ответа. Обычно при первом запросе сервер возвращает полный ответ со всем содержимым:
 - Заголовки: Информация о типе контента, длине, поддержке кеширования и т. д.
 - Тело: Само содержимое страницы, например HTML-код.
- iii. Загрузка дополнительных ресурсов: Браузер анализирует HTML и отправляет дополнительные GET-запросы для загрузки встроенных ресурсов, таких как CSS, JavaScript, изображения и т. д.
- iv. Кеширование: Браузер кэширует полученные ресурсы (если это разрешено заголовками ответа, такими как `Cache-Control` , `Expires` , `ETag`), чтобы ускорить дальнейшие загрузки этой страницы.

- **Вторичный запрос-обновление (Обновление страницы)**

При повторном посещении страницы или обновлении (нажатие F5) браузер сначала проверяет кэш и затем решает, как обрабатывать запросы:

- i. Использование кэша:
 - **Прямая загрузка из кэша (Strong Caching):** Если ресурсы в кэше все еще действительны (например, `Cache-Control: max-age` еще не истек), браузер использует кэшированные версии этих ресурсов, не отправляя запросы на сервер. В этом случае браузер просто загружает ресурсы из локального кэша.
- ii. Условный запрос к серверу (Conditional Request):
 - Если ресурс истек в кэше, браузер отправляет условный GET-запрос с заголовками, такими как `If-Modified-Since` или `If-None-Match` (использует `ETag`), чтобы проверить, изменился ли ресурс на сервере.
 - Ответ сервера:
 - `304 Not Modified` : Если ресурс не

изменился, сервер возвращает ответ

304 Not Modified без тела контента.

Браузер использует кэшированную копию.

- 200 OK : Если ресурс изменился, сервер отправляет новый ресурс с кодом 200 OK и браузер обновляет кэш.

4.4. Анализ DNS-трафика

The image shows a Windows command prompt window with the following text:

```
C:\Users\Tolia>ipconfig /flushdns

Windows IP 配置

已成功刷新 DNS 解析缓存。

C:\Users\Tolia>
```

Below the command prompt is a Wireshark packet capture window showing DNS traffic. The packet list shows several DNS queries and responses. The selected packet is a DNS query (No. 88967) from source fe80::dad1:7500:3b6... to destination fe80::1. The packet details show a Standard query for zhoufive.de. The packet bytes show the raw DNS data.

По результатам анализа собранной трассы, ответьте на следующие вопросы.

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

DNS-запросы отправляются на специальные серверы, называемые DNS-

серверами (или резолверами), а не непосредственно на адрес посещаемого сайта.

2. Какие бывают типы DNS-запросов?

- i. **A-запрос (Address Record):** Используется для получения IPv4-адреса, соответствующего доменному имени. Например, запрос A-записи для `example.com` вернет его IP-адрес.
- ii. **AAAA-запрос (IPv6 Address Record):** Похож на A-запрос, но используется для получения IPv6-адреса, соответствующего доменному имени.
- iii. **CNAME-запрос (Canonical Name Record):** Используется для получения канонического имени для домена. Он позволяет одному доменному имени быть псевдонимом для другого. Например, `www.example.com` может быть CNAME для `example.com`.
- iv. **MX-запрос (Mail Exchange Record):** Используется для определения почтового сервера, ответственного за прием электронной почты для домена.
- v. **NS-запрос (Name Server Record):** Указывает на авторитетные DNS-серверы, обслуживающие конкретный домен.
- vi. **TXT-запрос:** Возвращает текстовые данные, связанные с доменом. Используется, например, для SPF-записей (борьба со спамом) или подтверждения владения доменом.
- vii. **PTR-запрос (Pointer Record):** Используется для обратного DNS-разрешения — получения доменного имени по IP-адресу.

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

Независимые DNS-запросы для получения изображений с сайта необходимы в ситуациях, когда изображения находятся на других доменах, отличных от домена основного сайта.

4.5. Анализ ARP-трафика

```
C:\Windows\System32>arp -a
```

```
接口: 192.168.0.14 --- 0x6
```

Internet 地址	物理地址	类型
192.168.0.1	e0-36-76-ec-d9-c5	动态
192.168.0.255	ff-ff-ff-ff-ff-ff	静态
224.0.0.2	01-00-5e-00-00-02	静态
224.0.0.22	01-00-5e-00-00-16	静态
224.0.0.251	01-00-5e-00-00-fb	静态
224.0.0.252	01-00-5e-00-00-fc	静态
239.192.152.143	01-00-5e-40-98-8f	静态
239.255.255.250	01-00-5e-7f-ff-fa	静态
255.255.255.255	ff-ff-ff-ff-ff-ff	静态

```
接口: 192.168.241.1 --- 0x12
```

Internet 地址	物理地址	类型
192.168.241.255	ff-ff-ff-ff-ff-ff	静态
224.0.0.2	01-00-5e-00-00-02	静态
224.0.0.22	01-00-5e-00-00-16	静态
224.0.0.251	01-00-5e-00-00-fb	静态
224.0.0.252	01-00-5e-00-00-fc	静态
239.192.152.143	01-00-5e-40-98-8f	静态
239.255.255.250	01-00-5e-7f-ff-fa	静态
255.255.255.255	ff-ff-ff-ff-ff-ff	静态

```
接口: 192.168.133.1 --- 0x15
```

Internet 地址	物理地址	类型
192.168.133.255	ff-ff-ff-ff-ff-ff	静态
224.0.0.2	01-00-5e-00-00-02	静态
224.0.0.22	01-00-5e-00-00-16	静态
224.0.0.251	01-00-5e-00-00-fb	静态
224.0.0.252	01-00-5e-00-00-fc	静态
239.192.152.143	01-00-5e-40-98-8f	静态
239.255.255.250	01-00-5e-7f-ff-fa	静态
255.255.255.255	ff-ff-ff-ff-ff-ff	静态

```
C:\Windows\System32>netsh interface ip delete arpcache  
确定。
```

```
C:\Windows\System32>arp -a
```

```
接口: 192.168.0.14 --- 0x6
```

Internet 地址	物理地址	类型
192.168.0.1	e0-36-76-ec-d9-c5	动态
192.168.0.255	ff-ff-ff-ff-ff-ff	静态
224.0.0.2	01-00-5e-00-00-02	静态
224.0.0.22	01-00-5e-00-00-16	静态

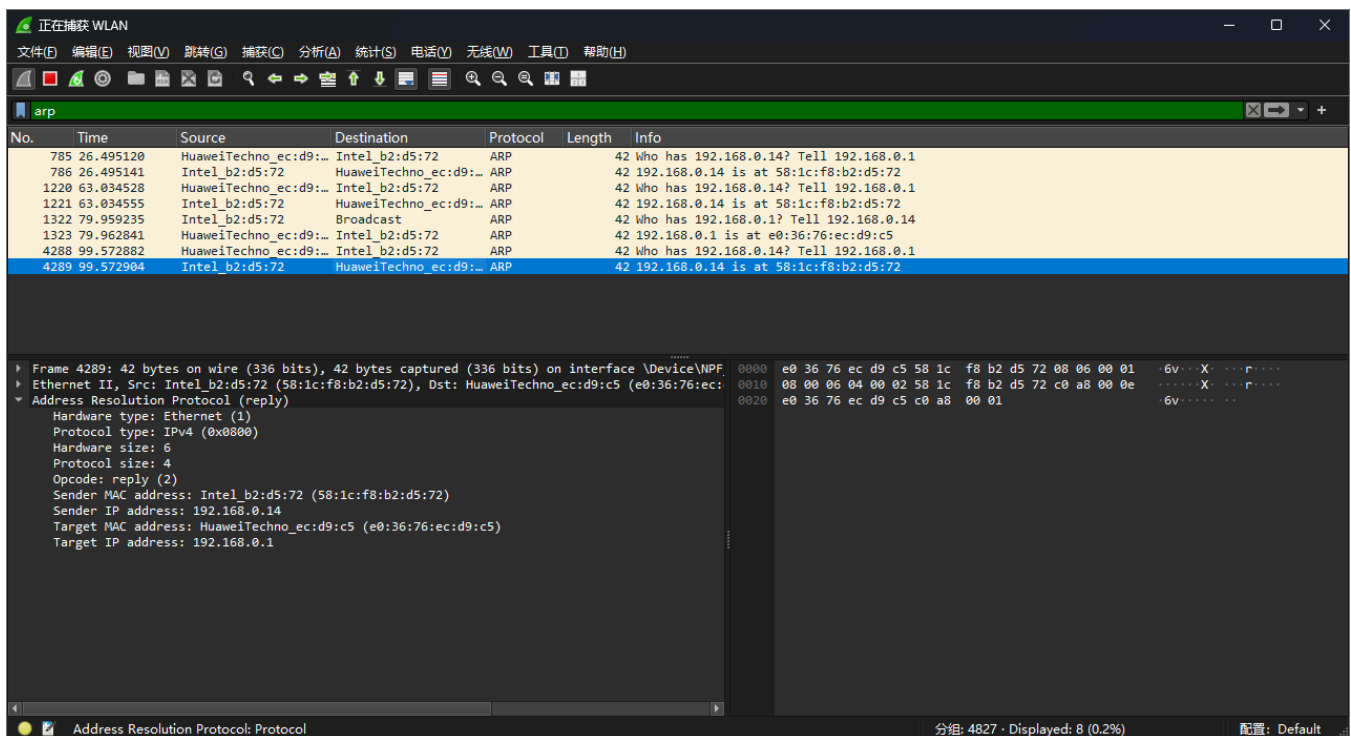
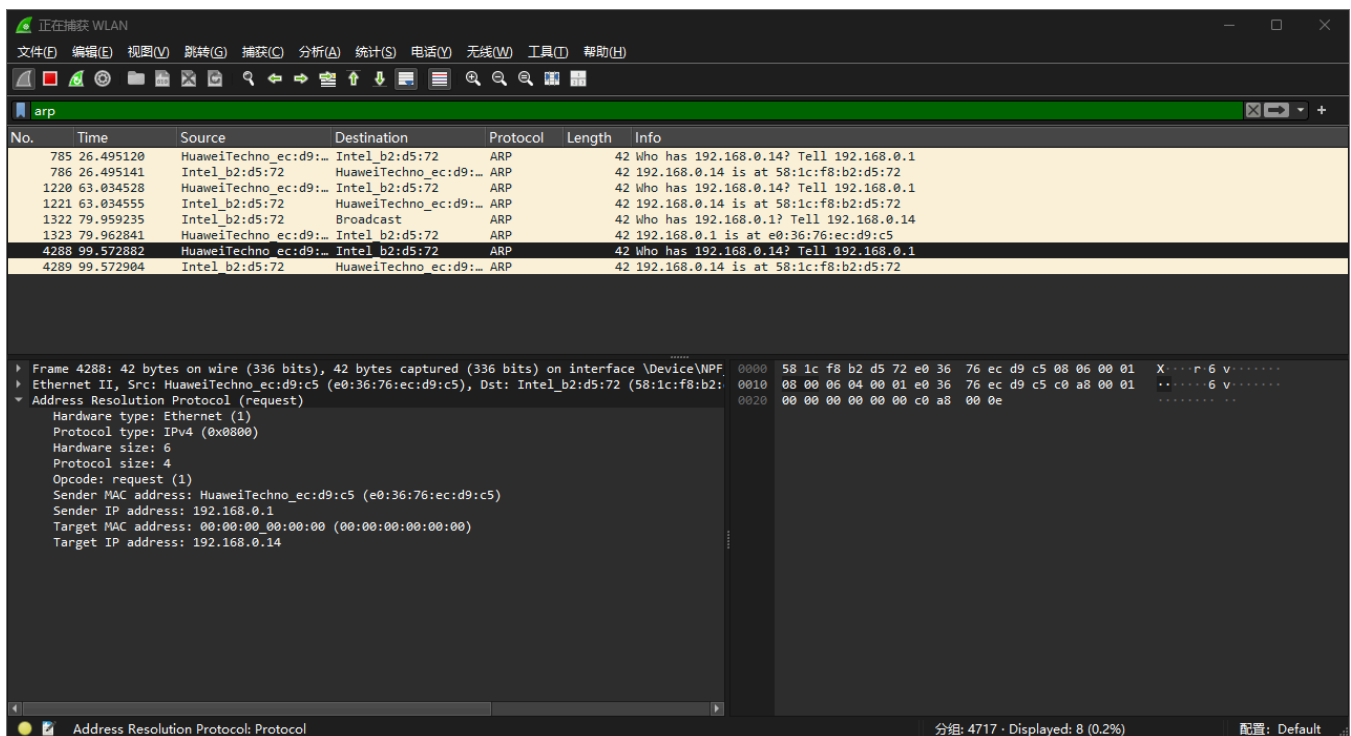
```
接口: 192.168.241.1 --- 0x12
```

Internet 地址	物理地址	类型
224.0.0.22	01-00-5e-00-00-16	静态

```
接口: 192.168.133.1 --- 0x15
```

Internet 地址	物理地址	类型
224.0.0.22	01-00-5e-00-00-16	静态

```
C:\Windows\System32>_
```

По результатам анализа собранной трассы, ответьте на следующие вопросы.

- Какие MAC-адреса присутствуют в захваченных пакетах ARP-протокола? Что означают эти адреса? Какие устройства они идентифицируют?
 - e0:36:76:ec:d9:c5 - адрес отправителя (маршрутизатор) (ip=192.168.0.1)

- 58:1c:f8:b2:d5:72 - адрес устройства получателя (компьютер, с которого производится запрос на сайт) (ip = 192.168.0.4)
 - 00:00:00:00:00:00 - broadcast-адрес
2. **Какие MAC-адреса присутствуют в захваченных HTTP-пакетах и что означают эти адреса? Что означают эти адреса? Какие устройства они идентифицируют?**
- В HTTP-пакетах присутствуют те же самые MAC-адреса, что и в ARP запросе. Они используются для идентификации отправителя и получателя HTTP-пакета.
3. **Для чего ARP-запрос содержит IP-адрес источника?**
- i. Идентификация отправителя: IP-адрес источника позволяет устройствам в сети определить, кто инициировал ARP-запрос. Это необходимо, чтобы знать, куда отправить ответ.
 - ii. Ответ на запрос: Устройства, получившие ARP-запрос, используют IP-адрес источника для формирования ответа. ARP-ответ содержит MAC-адрес устройства, отправившего запрос, и этот ответ отправляется на указанный IP-адрес.
 - iii. Кеширование: IP-адрес источника помогает устройствам обновить свои ARP-таблицы, добавляя или обновляя записи о том, какой MAC-адрес соответствует определенному IP-адресу. Это необходимо для оптимизации сетевых операций и ускорения последующих взаимодействий.

4.6. 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».

```

C:\Windows\System32>nslookup -type=NS zhousfive.de
服务器: Unknown
Address: fe80::1

非权威应答:
zhousfive.de      nameserver = ns1029.ui-dns.biz
zhousfive.de      nameserver = ns1029.ui-dns.de
zhousfive.de      nameserver = ns1029.ui-dns.com
zhousfive.de      nameserver = ns1029.ui-dns.org

ns1029.ui-dns.org      internet address = 217.160.83.29
ns1029.ui-dns.biz      internet address = 217.160.81.29
ns1029.ui-dns.de      internet address = 217.160.80.29
ns1029.ui-dns.com      internet address = 217.160.82.29

C:\Windows\System32>nslookup zhousfive.de
服务器: Unknown
Address: fe80::1

非权威应答:
名称:      zhousfive.de
Addresses: 2001:8d8:100f:f000::230
           217.160.0.33

C:\Windows\System32>nslookup -type=NS zhousfive.de
服务器: Unknown
Address: fe80::1

非权威应答:
zhousfive.de      nameserver = ns1029.ui-dns.de
zhousfive.de      nameserver = ns1029.ui-dns.com
zhousfive.de      nameserver = ns1029.ui-dns.org
zhousfive.de      nameserver = ns1029.ui-dns.biz

ns1029.ui-dns.org      internet address = 217.160.83.29
ns1029.ui-dns.biz      internet address = 217.160.81.29
ns1029.ui-dns.de      internet address = 217.160.80.29
ns1029.ui-dns.com      internet address = 217.160.82.29

C:\Windows\System32>

```

[illegible]



1. Чем различается трасса трафика в п.2 и п.4, указанных выше?
Type: A (1) (Host Address)
Type: NS(2) (authoritative Name Server)
2. Что содержится в поле «Answers» DNS-ответа?

```

▼ Answers
  ▼ zhoufive.de: type NS, class IN, ns ns1029.ui-dns.de
    Name: zhoufive.de
    Type: NS (2) (authoritative Name Server)
    Class: IN (0x0001)
    Time to live: 3607 (1 hour, 7 seconds)
    Data length: 16
    Name Server: ns1029.ui-dns.de
  ▼ zhoufive.de: type NS, class IN, ns ns1029.ui-dns.com
    Name: zhoufive.de
    Type: NS (2) (authoritative Name Server)
    Class: IN (0x0001)
    Time to live: 3607 (1 hour, 7 seconds)
    Data length: 19
    Name Server: ns1029.ui-dns.com
  ▼ zhoufive.de: type NS, class IN, ns ns1029.ui-dns.org
    Name: zhoufive.de
    Type: NS (2) (authoritative Name Server)
    Class: IN (0x0001)
    Time to live: 3607 (1 hour, 7 seconds)
    Data length: 19
    Name Server: ns1029.ui-dns.org
  ▼ zhoufive.de: type NS, class IN, ns ns1029.ui-dns.biz
    Name: zhoufive.de
    Type: NS (2) (authoritative Name Server)
    Class: IN (0x0001)
    Time to live: 3607 (1 hour, 7 seconds)
    Data length: 19
    Name Server: ns1029.ui-dns.biz

```

- 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.
- RDLLENGTH — длина поля данных.
- RDATA — поле данных, формат и содержание которого зависит от типа записи.

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

Серверов, возвращающих авторитативный отклик, нет