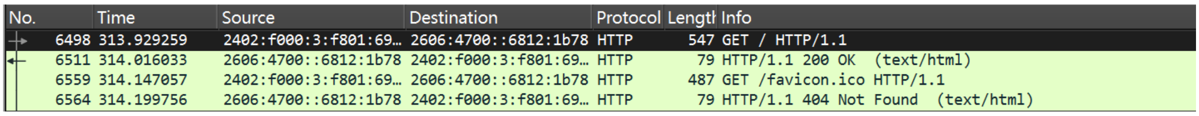
第五次实验报告

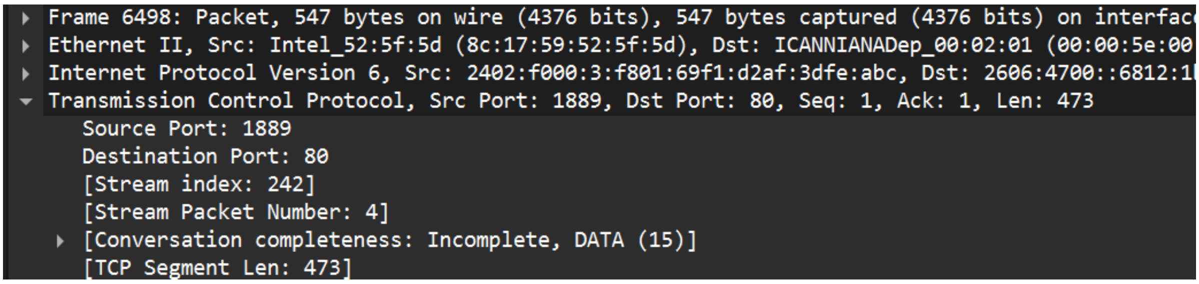
张峰源 2023010859

抓包实验1



No.	Time	Source	Destination	Protocol	Lengtł	Info
6498	313.929259	2402:f000:3:f801:69…	2606:4700::6812:1b78	HTTP	547	GET / HTTP/1.1
6511	314.016033	2606:4700::6812:1b78	2402:f000:3:f801:69…	HTTP	79	HTTP/1.1 200 OK (text/html)
6559	314.147057	2402:f000:3:f801:69…	2606:4700::6812:1b78	HTTP	487	GET /favicon.ico HTTP/1.1
6564	314.199756	2606:4700::6812:1b78	2402:f000:3:f801:69…	HTTP	79	HTTP/1.1 404 Not Found (text/html)

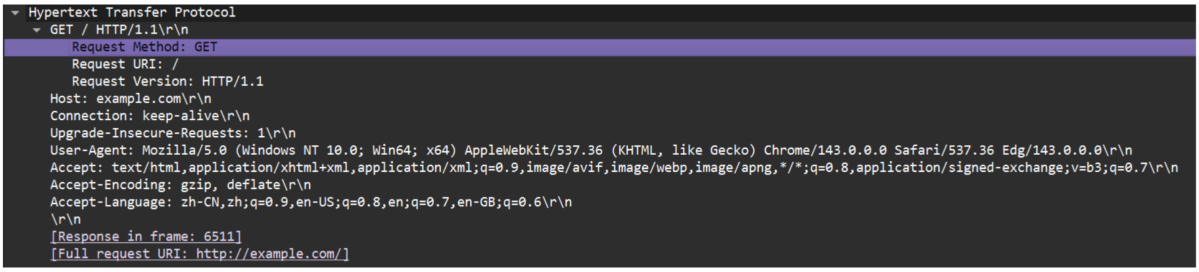
（1）HTTP使用的传输层协议是什么?



```
Frame 6498: Packet, 547 bytes on wire (4376 bits), 547 bytes captured (4376 bits) on interfac
Ethernet II, Src: Intel_52:5f:5d (8c:17:59:52:5f:5d), Dst: ICANNIANADep_00:02:01 (00:00:5e:00
Internet Protocol Version 6, Src: 2402:f000:3:f801:69f1:d2af:3dfe:abc, Dst: 2606:4700::6812:1
Transmission Control Protocol, Src Port: 1889, Dst Port: 80, Seq: 1, Ack: 1, Len: 473
    Source Port: 1889
    Destination Port: 80
    [Stream index: 242]
    [Stream Packet Number: 4]
    [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 473]
```

TCP协议

（2）HTTP请求包请求的方法是什么？请求的Host和URL是什么？使用的HTTP版本是什么?



```
Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
  Host: example.com\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36 Edg/143.0.0.0\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7\r\n
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6\r\n
  \r\n
  [Response in frame: 6511]
  [Full request URI: http://example.com/]
```

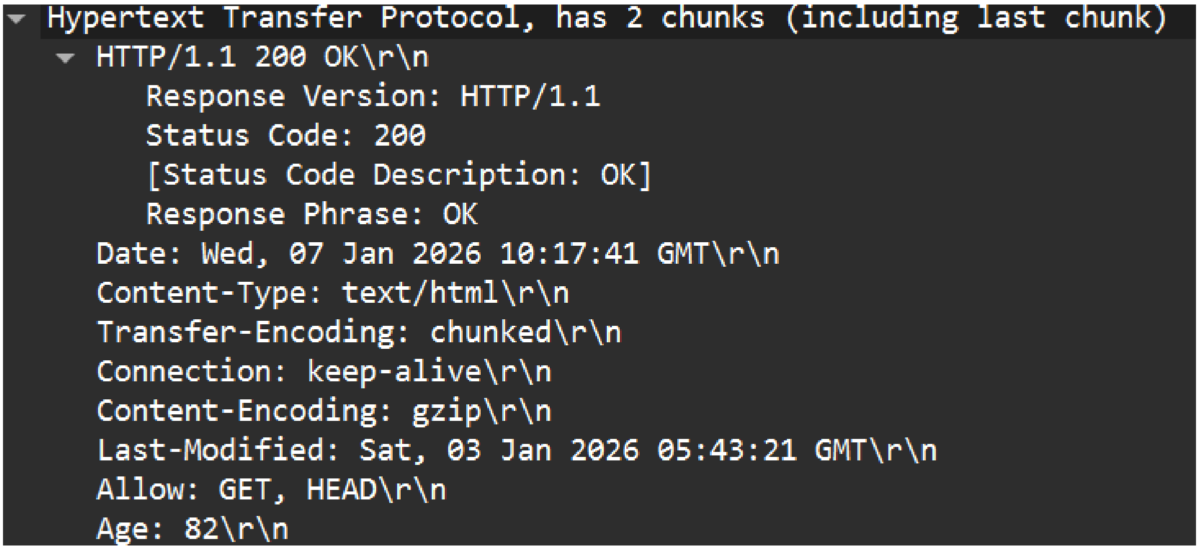
请求方法：GET

HOST：example.com

URL：Request URI: /

HTTP版本：HTTP/1.1

（3）HTTP响应包的状态码是什么？响应的Content-Type是什么?

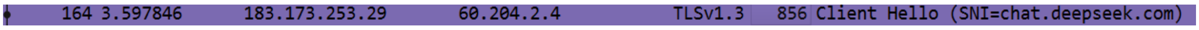


```
Hypertext Transfer Protocol, has 2 chunks (including last chunk)
  HTTP/1.1 200 OK\r\n
    Response Version: HTTP/1.1
    Status Code: 200
    [Status Code Description: OK]
    Response Phrase: OK
  Date: Wed, 07 Jan 2026 10:17:41 GMT\r\n
  Content-Type: text/html\r\n
  Transfer-Encoding: chunked\r\n
  Connection: keep-alive\r\n
  Content-Encoding: gzip\r\n
  Last-Modified: Sat, 03 Jan 2026 05:43:21 GMT\r\n
  Allow: GET, HEAD\r\n
  Age: 82\r\n
```

状态码：200

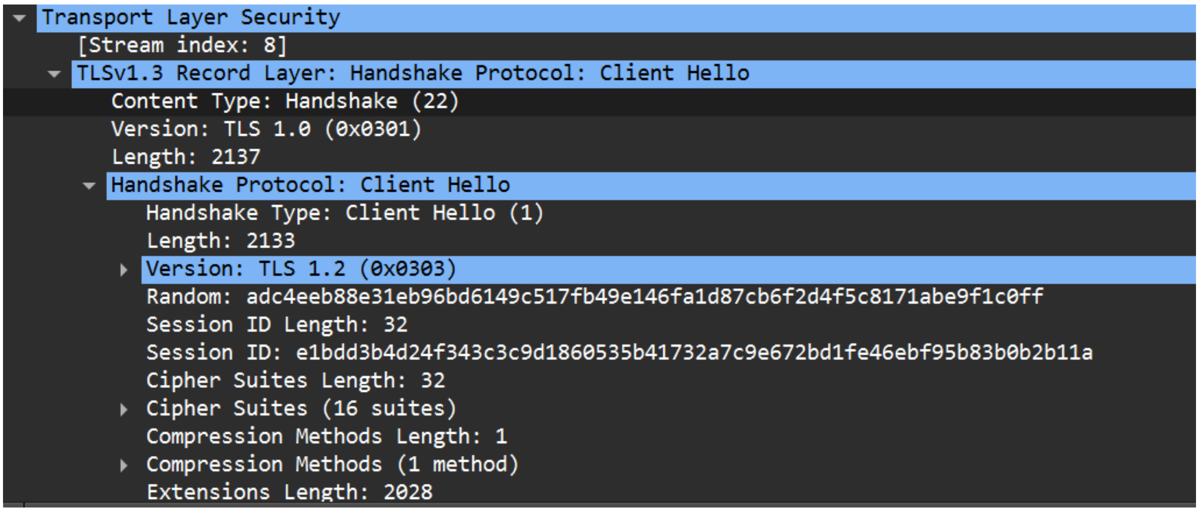
content-Type：text/html

deepseek



```
164 3.597846    183.173.253.29    60.204.2.4    TLSv1.3    856 Client Hello (SNI=chat.deepseek.com)
```

（4）TLS数据包使用的TLS版本是什么？你还能看到HTTP请求或响应的内容吗？为什么？

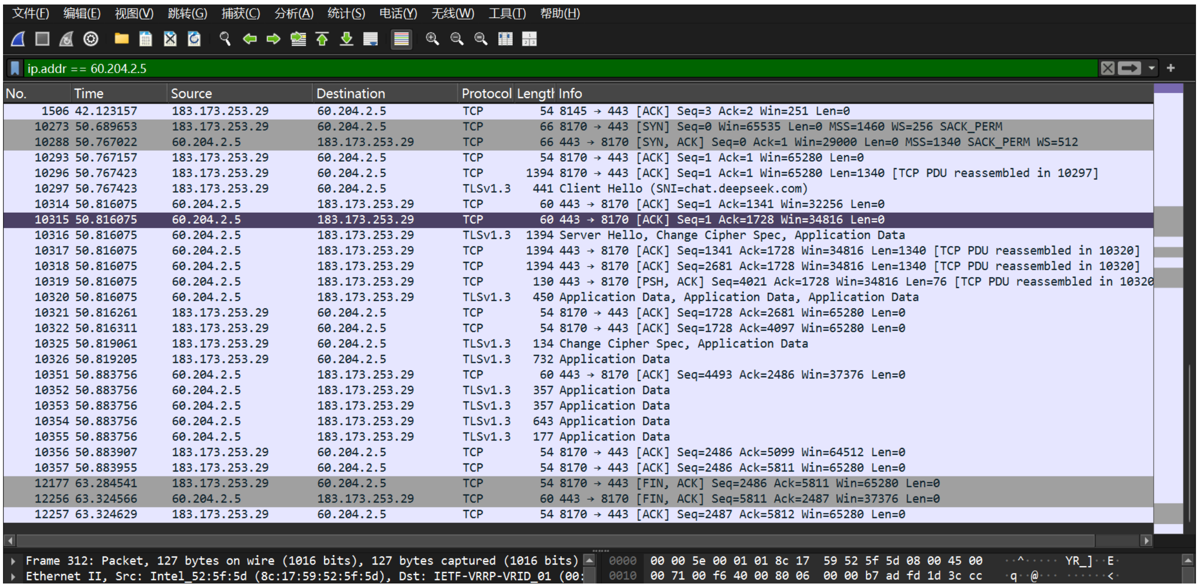


```
Transport Layer Security
  [Stream index: 8]
  TLSv1.3 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 2137
    Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 2133
      Version: TLS 1.2 (0x0303)
      Random: adc4eeb88e31eb96bd6149c517fb49e146fa1d87cb6f2d4f5c8171abe9f1c0ff
      Session ID Length: 32
      Session ID: e1bdd3b4d24f343c3c9d1860535b41732a7c9e672bd1fe46ebf95b83b0b2b11a
      Cipher Suites Length: 32
      Cipher Suites (16 suites)
      Compression Methods Length: 1
      Compression Methods (1 method)
      Extensions Length: 2028
```

TLSv1.3。

不能。因为被TSL加密了。

（5）简要展示下HTTPS整体的交互流程（言之有理即可，无需过于细节）



```
ip.addr == 60.204.2.5
```

```
No.   Time      Source          Destination     Protocol Length Info
 1506 42.123157 183.173.253.29  60.204.2.5      TCP        54 8145 → 443 [ACK] Seq=3 Ack=2 Win=251 Len=0
10273 50.689653 183.173.253.29  60.204.2.5      TCP        66 8170 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
10288 50.767022 60.204.2.5      183.173.253.29  TCP        66 443 → 8170 [SYN, ACK] Seq=0 Ack=1 Win=29000 Len=0 MSS=1340 SACK_PERM WS=512
10293 50.767157 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
10296 50.767423 183.173.253.29  60.204.2.5      TCP      1394 8170 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=1340 [TCP PDU reassembled in 10297]
10297 50.767423 183.173.253.29  60.204.2.5      TLSv1.3   441 Client Hello (SNI=chat.deepseek.com)
10314 50.816075 60.204.2.5      183.173.253.29  TCP        60 443 → 8170 [ACK] Seq=1 Ack=1341 Win=32256 Len=0
10315 50.816075 60.204.2.5      183.173.253.29  TCP        60 443 → 8170 [ACK] Seq=1 Ack=1728 Win=34816 Len=0
10316 50.816075 60.204.2.5      183.173.253.29  TLSv1.3  1394 Server Hello, Change Cipher Spec, Application Data
10317 50.816075 60.204.2.5      183.173.253.29  TCP      1394 443 → 8170 [ACK] Seq=1341 Ack=1728 Win=34816 Len=1340 [TCP PDU reassembled in 10320]
10318 50.816075 60.204.2.5      183.173.253.29  TCP      1394 443 → 8170 [ACK] Seq=2681 Ack=1728 Win=34816 Len=1340 [TCP PDU reassembled in 10320]
10319 50.816075 60.204.2.5      183.173.253.29  TCP       130 443 → 8170 [PSH, ACK] Seq=4021 Ack=1728 Win=34816 Len=76 [TCP PDU reassembled in 10320
10320 50.816075 60.204.2.5      183.173.253.29  TLSv1.3   450 Application Data, Application Data, Application Data
10321 50.816261 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [ACK] Seq=1728 Ack=2681 Win=65280 Len=0
10322 50.816311 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [ACK] Seq=1728 Ack=4097 Win=65280 Len=0
10325 50.819061 183.173.253.29  60.204.2.5      TLSv1.3   134 Change Cipher Spec, Application Data
10326 50.819205 183.173.253.29  60.204.2.5      TLSv1.3   732 Application Data
10351 50.883756 60.204.2.5      183.173.253.29  TCP        60 443 → 8170 [ACK] Seq=4493 Ack=2486 Win=37376 Len=0
10352 50.883756 60.204.2.5      183.173.253.29  TLSv1.3   357 Application Data
10353 50.883756 60.204.2.5      183.173.253.29  TLSv1.3   357 Application Data
10354 50.883756 60.204.2.5      183.173.253.29  TLSv1.3   643 Application Data
10355 50.883756 60.204.2.5      183.173.253.29  TLSv1.3   177 Application Data
10356 50.883907 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [ACK] Seq=2486 Ack=5099 Win=64512 Len=0
10357 50.883955 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [ACK] Seq=2486 Ack=5811 Win=65280 Len=0
12177 63.284541 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [FIN, ACK] Seq=2486 Ack=5811 Win=65280 Len=0
12256 63.324566 60.204.2.5      183.173.253.29  TCP        60 443 → 8170 [FIN, ACK] Seq=5811 Ack=2487 Win=37376 Len=0
12257 63.324629 183.173.253.29  60.204.2.5      TCP        54 8170 → 443 [ACK] Seq=2487 Ack=5812 Win=65280 Len=0
```

```
▸ Frame 312: Packet, 127 bytes on wire (1016 bits), 127 bytes captured (1016 bits)    0000  00 00 5e 00 01 01 8c 17  59 52 5f 5d 08 00 45 00   ··^·····YR_]··E·
▸ Ethernet II, Src: Intel_52:5f:5d (8c:17:59:52:5f:5d), Dst: IETF-VRRP-VRID_01 (00:   0010  00 71 00 f6 40 00 80 06  00 00 b7 ad fd 1d 3c cc   ·q··@·········<·
```

- TCP三次握手建立连接：SYN→SYN+ACK→ACK

- TLS握手：Client Hello → Server Hello， Change Cipher Spec（通知切换到加密模式）

- 加密数据传输：application data

- 四次挥手：

  第一次挥手（客户端→服务器）：包`12177`

  - Info：`8170 → 443 [FIN, ACK] Seq=2486 Ack=5811`

  - 标志位：`FIN=1 + ACK=1`，请求关闭连接。

  第二次 + 第三次挥手（服务器→客户端）：包`12256`

  - Info：`443 → 8170 [FIN, ACK] Seq=5811 Ack=2487`

    服务器将ACK 和 自己的关闭请求FIN 合并发送，对应四次挥手中的第二次和第三次。

  第四次挥手（客户端→服务器）：包`12257`

  - Info：`8170 → 443 [ACK] Seq=2487 Ack=5812`

  - 标志位：`ACK=1`，确认服务器的关闭请求，完成连接关闭。

简述题

（1）根据实验观察到的结果和课程内容，分析HTTP协议的头部与IP头或TCP头的设计思路差异。

答:

1.**设计目标的差异**：HTTP 头部服务于应用层的业务逻辑，比如实验中 HTTP 请求头里的`Host`字段，是为了让一台服务器能托管多个网站；而 IP 头（如`Source/Destination Address`）是为了网络层的路由定位，TCP 头（如`Sequence Number`）是为了传输层的可靠传输，二者的目标是让数据稳定、准确地跨网络送达，不涉及上层业务语义。

2.**字段灵活性的差异**：HTTP 头部是可变长、可扩展的，实验里能看到`Cookie`、`User-Agent`等自定义字段，可随业务需求新增；但 IP/TCP 头以固定结构为主，可选字段仅作补充（如 TCP 的 MSS 选项），这种设计是为了避免字段冗余，保证传输效率。