# 21级计算机科学与技术2班

# 计算机网络期末综合实验

## 学号:21307174 姓名:刘俊杰

## 一、实验内容

**重新启动一台配置好可上互联网的计算机后，请关闭无关的网络应用。**

**（1） 运行wireshark 软件准备分析上网行为，启动抓取网络数据。**

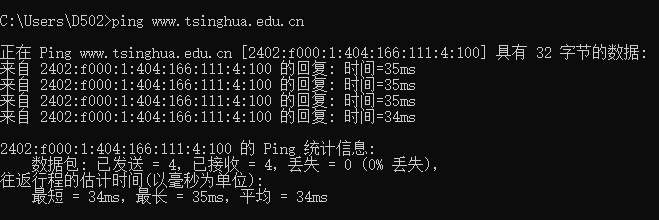
**（2） 输入一个 网址 <http://www.tsinghua.edu.cn>， 一直到整个网页显示。**

**请记录并分析：**

1. **具体分析应用层协议的运行的过程。依据网络数据说明相关协议的基本运行情况、顺序及其作用。**
2. **分析传输层运行的协议过程。依据网络数据说明相关协议的基本运行情况、顺序及其作用。**
3. **按实际测试，网页自动跳转运行https协议。请分析其应用层与传输层协议的消息交换过程。**
4. **分析网络层运行的协议。依据网络数据说明相关协议的基本运行情况、顺序及其作用。**
5. **分析数据链路层运行的协议。依据网络数据说明相关协议的基本运行情况、顺序及其作用。**

## 二、实验操作

**①首先通过ping查询<http://www.tsinghua.edu.cn> 的IP地址:**

****

**②输入ipconfig /all查询本机信息:**

****

**可以看到本机的IPv4地址为172.16.5.2，默认网关为172.16.0.1，Mac地址为18-60-24-8C-17-60**

**③重新启动计算机。**

**④在本地执行ipconfig /flushdns和arp -d分别清除dns缓存和arp缓存**

**⑤运行wireshark 软件准备分析上网行为，启动抓取网络数据。**

**⑥输入 网址 <http://www.tsinghua.edu.cn>， 一直到整个网页显示。**

****

**⑦最后关闭网页**

## 三、实验结果及分析

### (1)具体分析应用层协议的运行的过程。依据网络数据说明相关协议的基本运行情况、顺序及其作用。

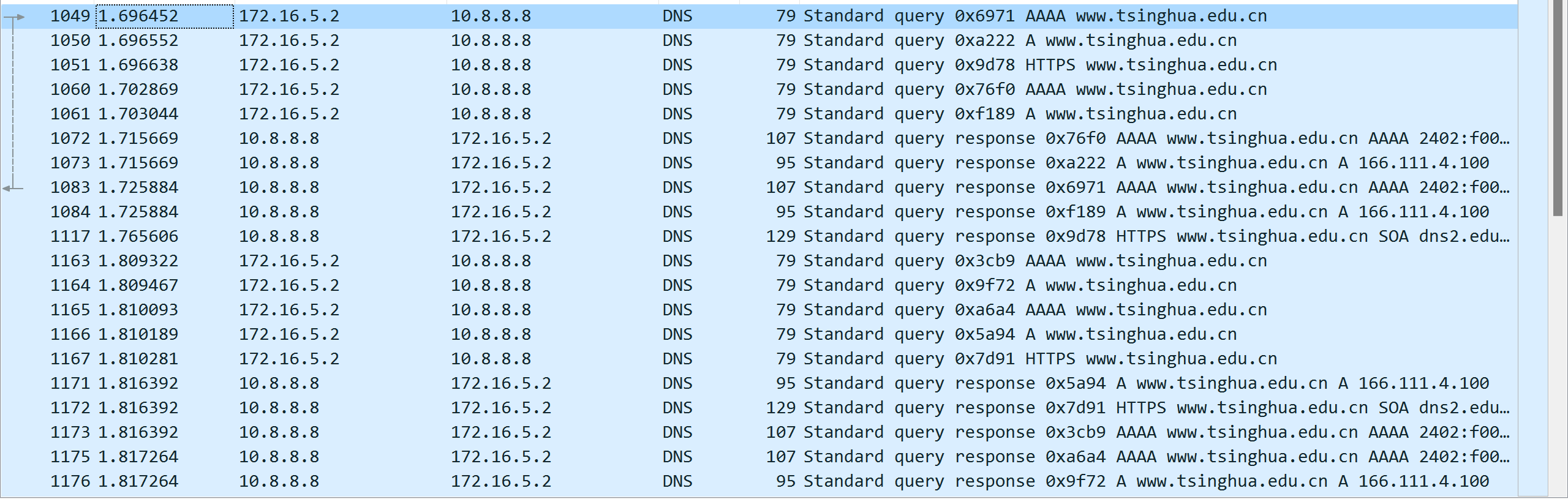
### DNS

**首先我们先从之前的实验准备中得知DNS服务器的IP地址为10.8.8.8**

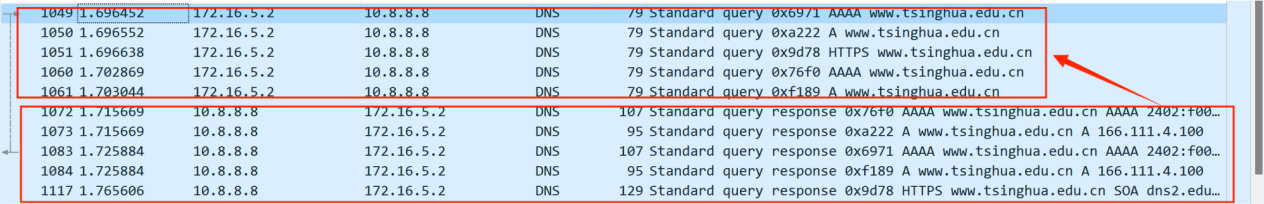
**打开网页之前本机此时只知道清华大学官网的域名www.tsinghua.edu.cn，还需要知道清华大学官网的IP地址，因为之前没有访问过清华大学官网所以本机的DNS缓存中没有对应的IP地址，所以通过向DNS服务器发送DNS查询报文查询这个网址映射关系，**

**可以看到下列抓包结果中有DNS的查询报文(172.16.5.2本机发往10.8.8.8DNS服务器)和回复报文。**

**以下展示的是所有与www.tsinghua.edu.cn的有关DNS报文:**

****

**请求和响应关系:**

****

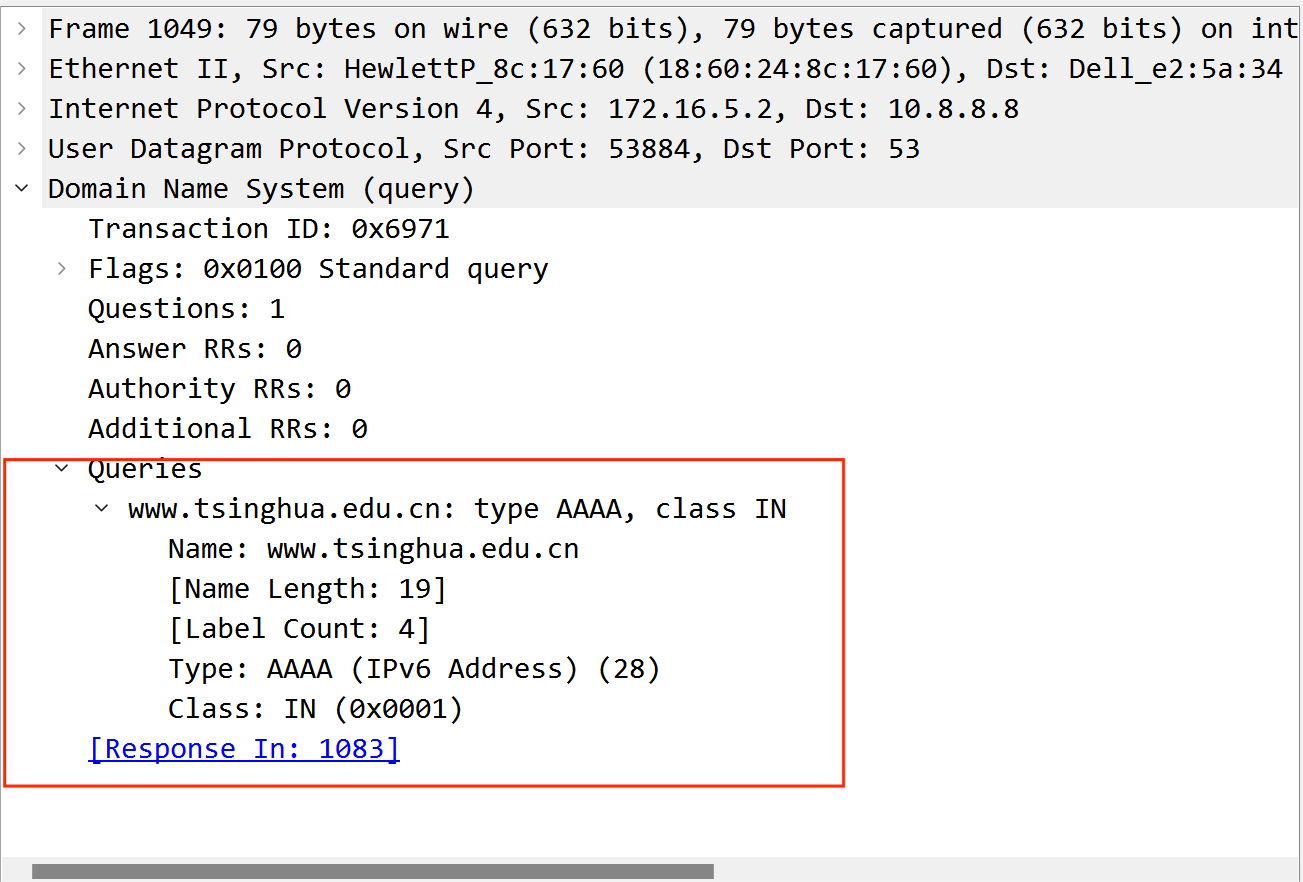
**可以看到DNS请求并回复了www.tsinghua.edu.cn的IPv4地址和IPv6地址 HTTPS记录**

**DNS中AAAA记录请求的是www.tsinghua.edu.cn的IPv6地址**

**DNS中A记录请求的是www.tsinghua.edu.cn的IPv4地址**

**DNS中HTTP记录查询www.tsinghua.edu.cn的HTTPS记录**

**点开1049的DNS查询报文:**



**DNS查询部分：**

**事务ID（Transaction ID）： 0x6971**

**标志（Flags）： 0x0100（标准查询）**

**问题数（Questions）： 1**

**应答资源记录数（Answer RRs）： 0**

**权威资源记录数（Authority RRs）： 0**

**附加资源记录数（Additional RRs）： 0**

**DNS查询详细信息：**

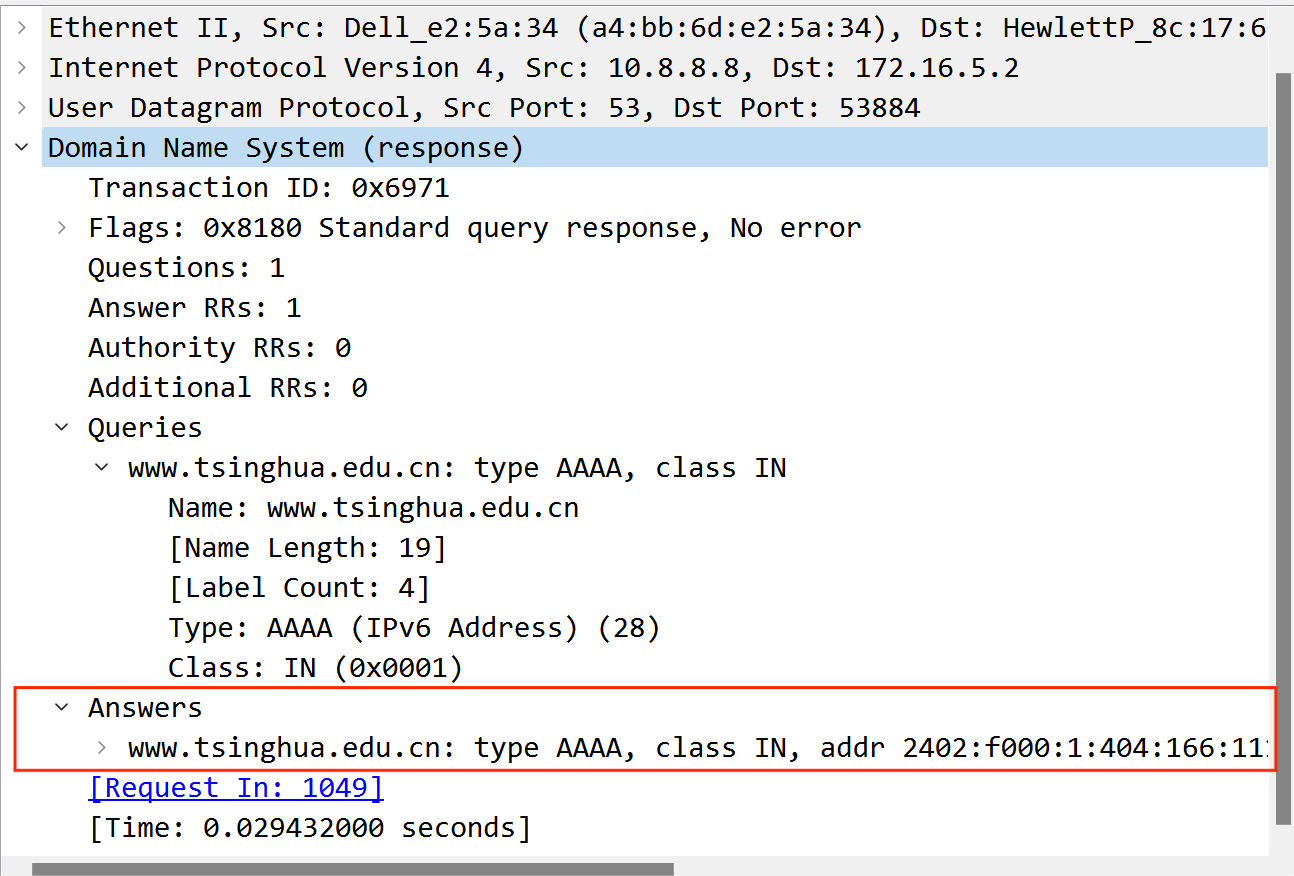
**查询名称（Queries）： www.tsinghua.edu.cn**

**类型（Type）： AAAA（IPv6地址）**

**类别（Class）： IN（互联网）**

**这个报文表示对域名"www.tsinghua.edu.cn"的IPv6地址（AAAA记录）的DNS查询，源IP为172.16.5.2，目标IP为10.8.8.8，在UDP的53端口上进行通信。**

**再点开其对应的序号为1083的DNS回复报文:**



**DNS响应部分：**

**事务ID（Transaction ID）： 0x6971**

**标志（Flags）： 0x8180（标准查询响应，无错误）**

**问题数（Questions）： 1**

**应答资源记录数（Answer RRs）： 1**

**权威资源记录数（Authority RRs）： 0**

**附加资源记录数（Additional RRs）： 0**

**DNS查询详细信息：**

**响应信息：**

**请求ID： 1049（与之前的请求相关联）**

**时间： 0.029432000秒**

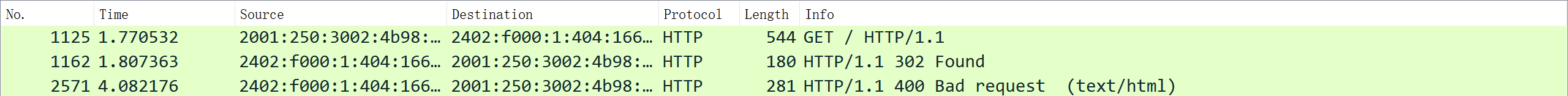
**在这个DNS响应中：**

**事务ID与之前的DNS查询相同，用于关联查询和响应。**

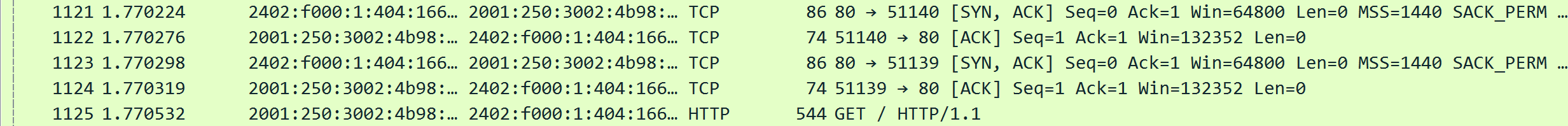
**在响应信息中的Answer中返回了之前查询的DNS请求报文的www.tsinghua.edu.cn的IPv6地址。**

### HTTP

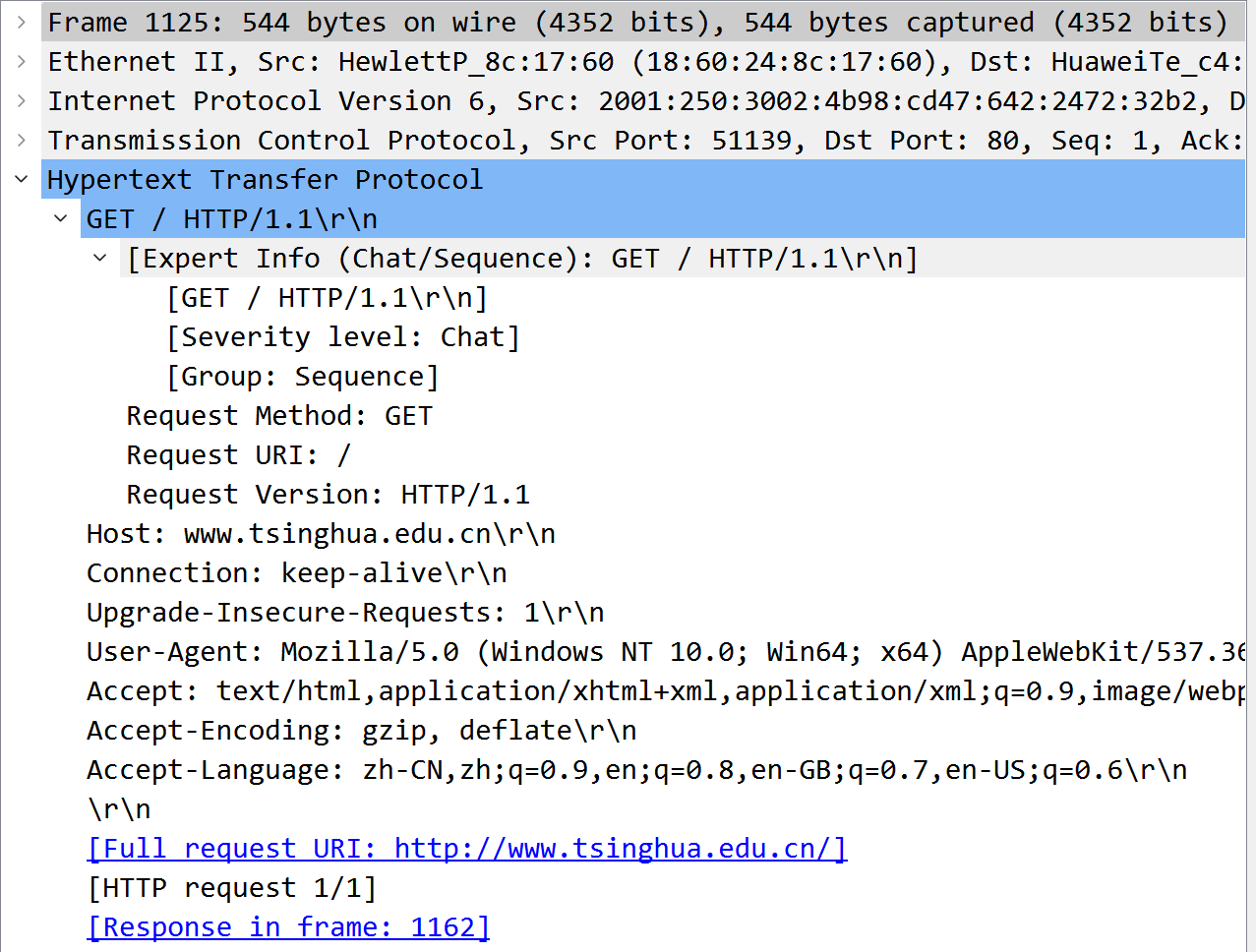
**Web的应用层协议是HTTP,本机和服务端通过交换HTTP报文进行会话,HTTP使用TCP作为它的支撑运输协议，一旦建立起TCP连接客户通过套接字接口发送HTTP并在接口收到回复**

****

**可以看到前两条HTTP报文发生在TCP连接建立之后**

****

**点开序号为1125的HTTP请求报文:**

****

**可以看到该报文是使用TCP来支撑运输，并装载在IPv6报文中。可以看到该报文是http://www.tsinghua.edu.cn/ 的 HTTP GET 请求。并且对应的回复报文在序号1162处。**

**分析HTTP请求行：**

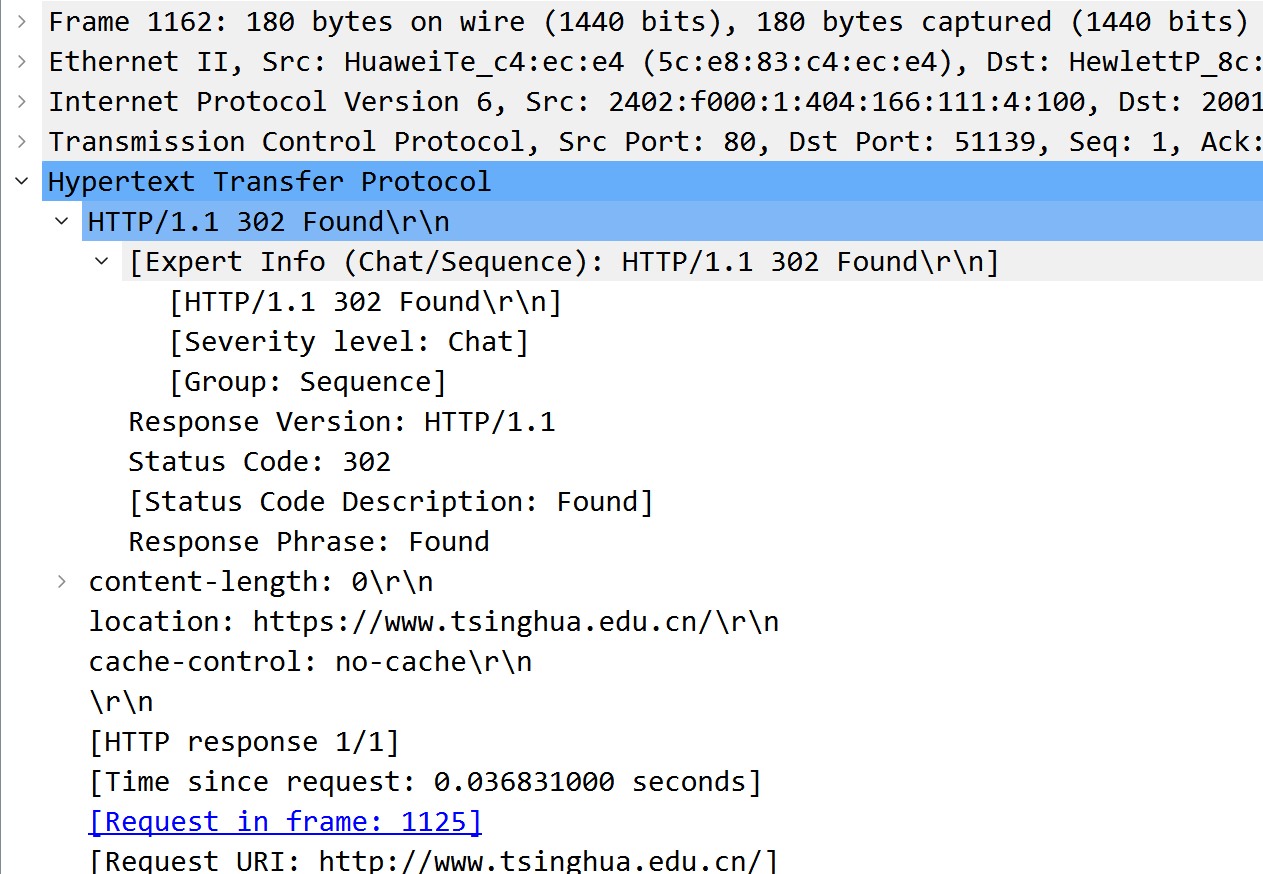
**请求方法（Request Method）： GET**

**请求URI（Request URI）： /**

**请求版本（Request Version）： HTTP/1.1**

**这表明是一个GET请求，请求的资源路径是"/"，使用的HTTP版本是1.1。**

**再点开序号为1162的HTTP回复报文：**

****

**HTTP响应行：**

**响应版本（Response Version）： HTTP/1.1**

**状态码（Status Code）： 302**

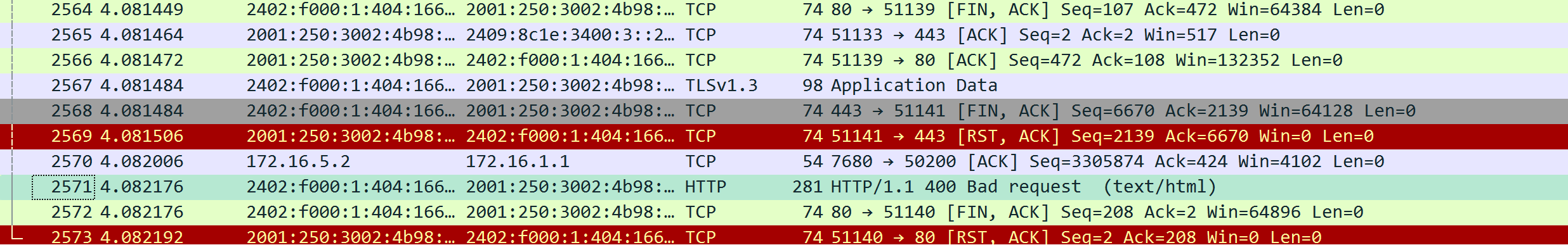
**状态码描述（Status Code Description）： Found**

**响应短语（Response Phrase）： Found**

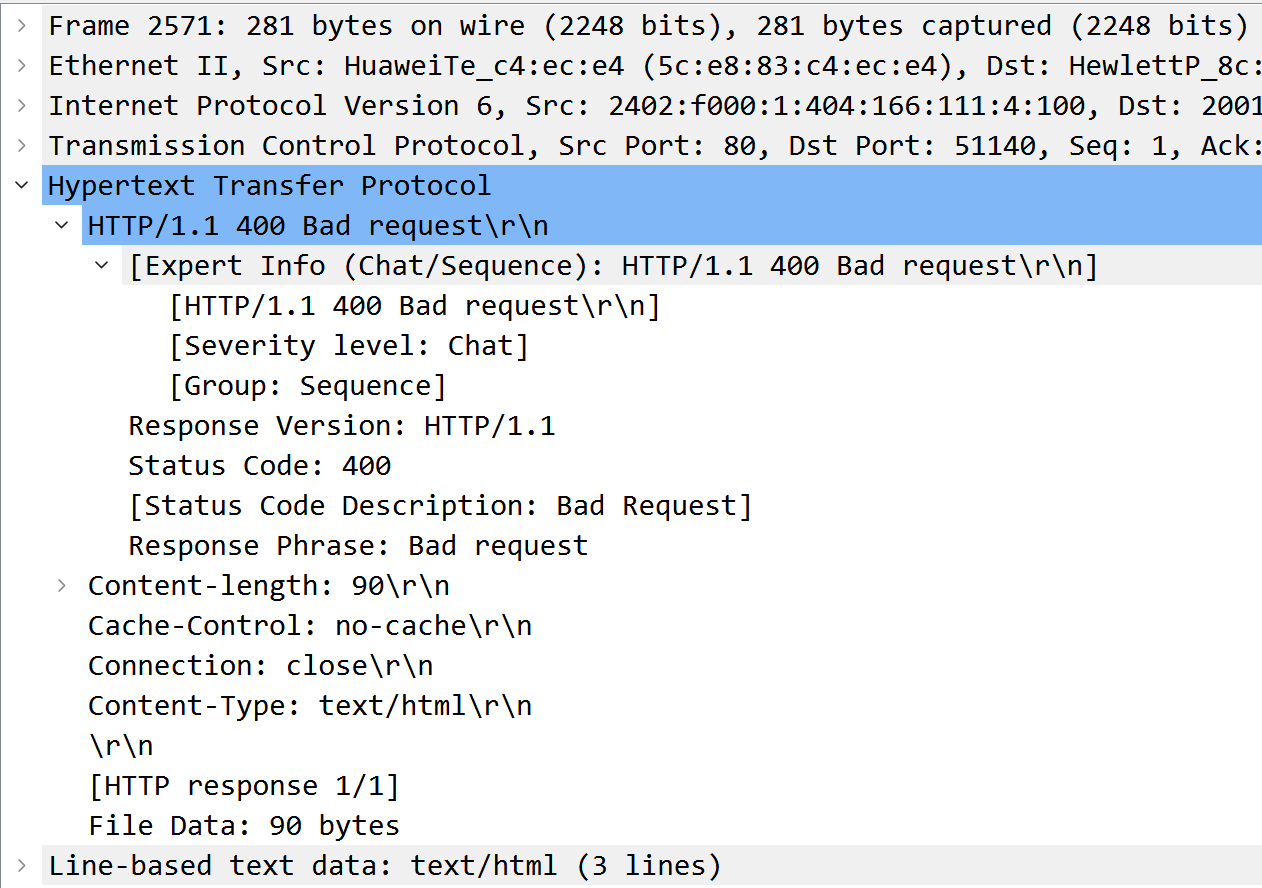
**这表明这是一个HTTP/1.1版本的302 Found响应，表示请求的资源已被临时移动。**

**可以看到此报文同样转载在TCP和IPv6之中，是对序号为1125的请求报文的回复，可以看到请求的对象的资源已被找到并响应。**

**最后一条HTTP报文发生在页面关闭、TCP连接断开的时候**

****

**点开查看详细信息:**

****

**这是一个 HTTP 错误响应，状态码为 400，表示客户端发送了一个无效的请求。详细分析如下：**

**HTTP 协议：**

**HTTP 版本：HTTP/1.1**

**状态码：400 Bad Request**

**响应头部和正文：未提供，但状态码 400 通常会伴随一些错误信息。**

**状态码 400 表示客户端发出的请求在服务器端无效，可能是由于请求语法错误、无效的请求消息、请求参数不完整等原因导致的。**

#### 分析传输层运行的协议过程。依据网络数据说明相关协议的基本运行情况、顺序及其作用。

**传输层运行的协议过程**

**①DNS解析：**

**在打开网页之前，浏览器首先要解析域名，获取服务器的IP地址。这一步使用的是DNS协议，通常在传输层上使用UDP进行域名解析。**

**②建立TCP连接 - 三次握手：**

**一旦知道了服务器的IP地址，浏览器就需要与服务器建立TCP连接。这是通过TCP的三次握手过程完成的：**

**客户端发送SYN报文： 浏览器向服务器发送一个TCP报文，其中包含SYN标志位，表示请求建立连接，并选择一个初始序列号。**

**服务器回应SYN+ACK报文： 服务器收到客户端的请求后，回应一个包含SYN和ACK标志位的TCP报文，表示确认连接请求，并选择自己的初始序列号。**

**客户端发送ACK报文： 最后，客户端发送一个包含ACK标志位的TCP报文，表示确认服务器的响应。此时，TCP连接建立成功。**

**③HTTP请求：**

**一旦建立了TCP连接，浏览器通过该连接发送HTTP请求。HTTP请求是在应用层定义的，它包含了请求的方法（GET、POST等）、资源路径、HTTP版本等信息。**

**④服务器处理请求并返回HTTP响应：**

**服务器接收到HTTP请求后，处理请求并生成HTTP响应。HTTP响应也包含在TCP连接中传输。**

**服务器通过TCP连接将HTTP响应发送回客户端，包含状态码、响应头和响应体等信息。**

**⑤断开TCP连接 - 四次挥手：**

**一旦HTTP响应传输完成，客户端和服务器需要断开TCP连接。这是通过TCP的四次挥手过程完成的：**

**客户端发送FIN报文： 客户端通知服务器它已经完成了数据的发送。**

**服务器回应ACK报文： 服务器确认客户端的FIN，并向客户端发送一个ACK，表示服务器也准备关闭连接。**

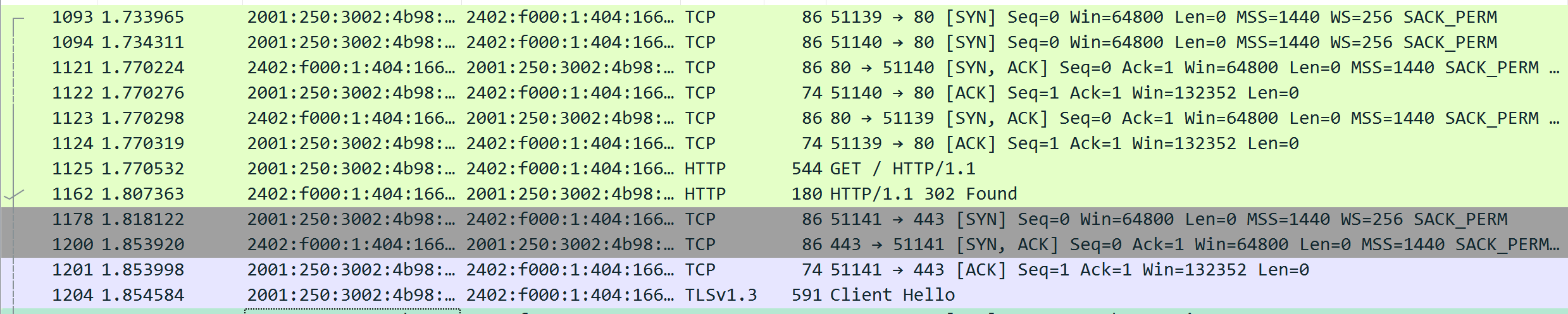
**服务器发送FIN报文： 服务器通知客户端它也完成了数据的发送。**

**客户端回应ACK报文： 客户端确认服务器的FIN，双方的连接就正式关闭了。**

**由于其他几个协议应用的传输层运行的协议过程在其部分已展示，所以接下来重点分析TCP建立连接的三次握手和取消连接的四次挥手:**

### TCP建立连接的三次握手

**查看捕获到的三次握手:**

****

**其实可以看到有三组TCP的三次握手,查看报文内容得到三次握手的报文:**

**①1093[SYN] 1123[SYN,ACK] 1124[SYN]**

**②1094[SYN] 1121[SYN,ACK] 1122[SYN]**

**③1178[SYN] 1200[SYN,ACK] 1201[SYN]**

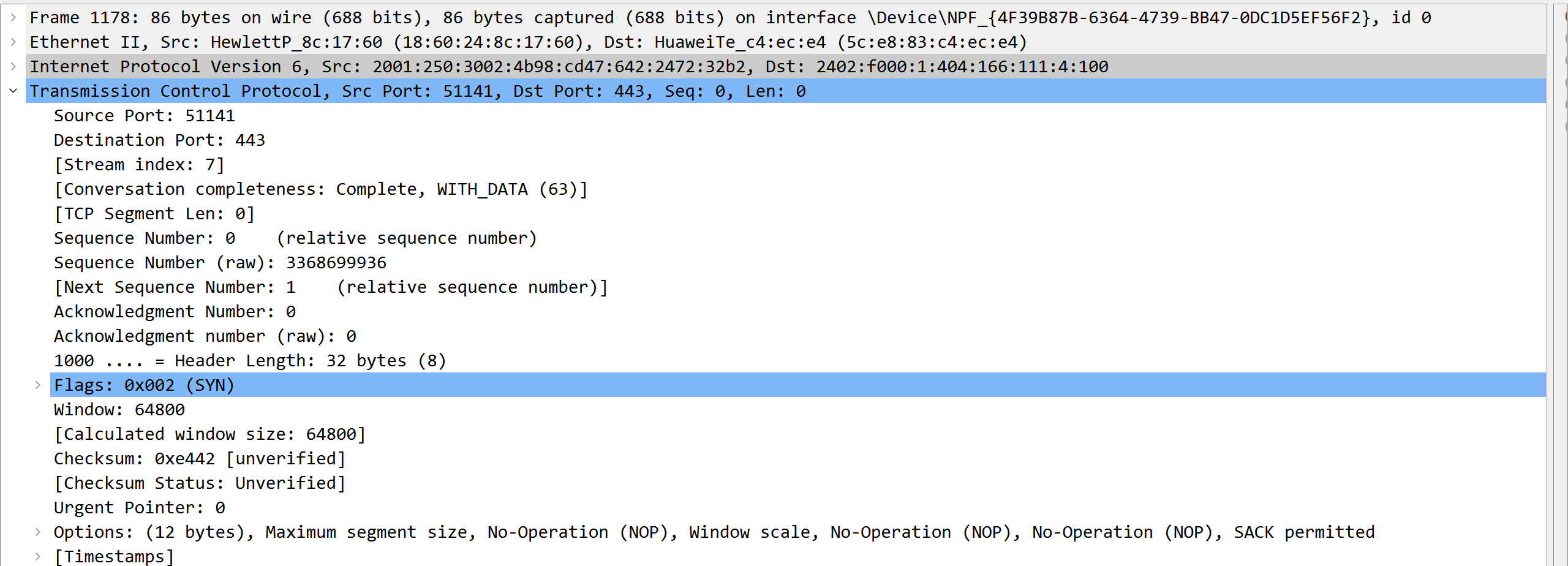
**之所以有三组TCP的三次握手，这是因为**

**①前两组是因为HTTP的非持续连接。在非持续连接中，每个HTTP请求都会建立一个新的TCP连接。每个请求完成后，TCP连接会立即关闭，不会被重用。所以每个请求都需要建立新的连接、进行TCP的三次握手。**

**②而最后一组握手在持续连接中，这样多个HTTP请求可以在同一个TCP连接上完成。TCP连接在完成所有请求后才会关闭，而不是在每个请求结束时关闭。**

**取1178[SYN] 1200[SYN,ACK] 1201[SYN]的握手来分析:**

**1178[SYN]:**



**分析:**

**源端口（Source Port）： 51141**

**目标端口（Destination Port）： 443**

**序列号（Sequence Number）： 0（这是TCP连接中第一个数据字节的序列号。在这个包中，它的值是0，表示这是一个初始的序列号。）**

**确认号（Acknowledgment Number）： 0表示对方期望接收的下一个序列号。在这个初始的SYN包中，通常是0。**

**头部长度（Header Length）： 32 bytes（8）TCP头部长度的单位是32位字（4字节），这里是32 bytes，表示8个32位字，即8 \* 4 = 32字节。**

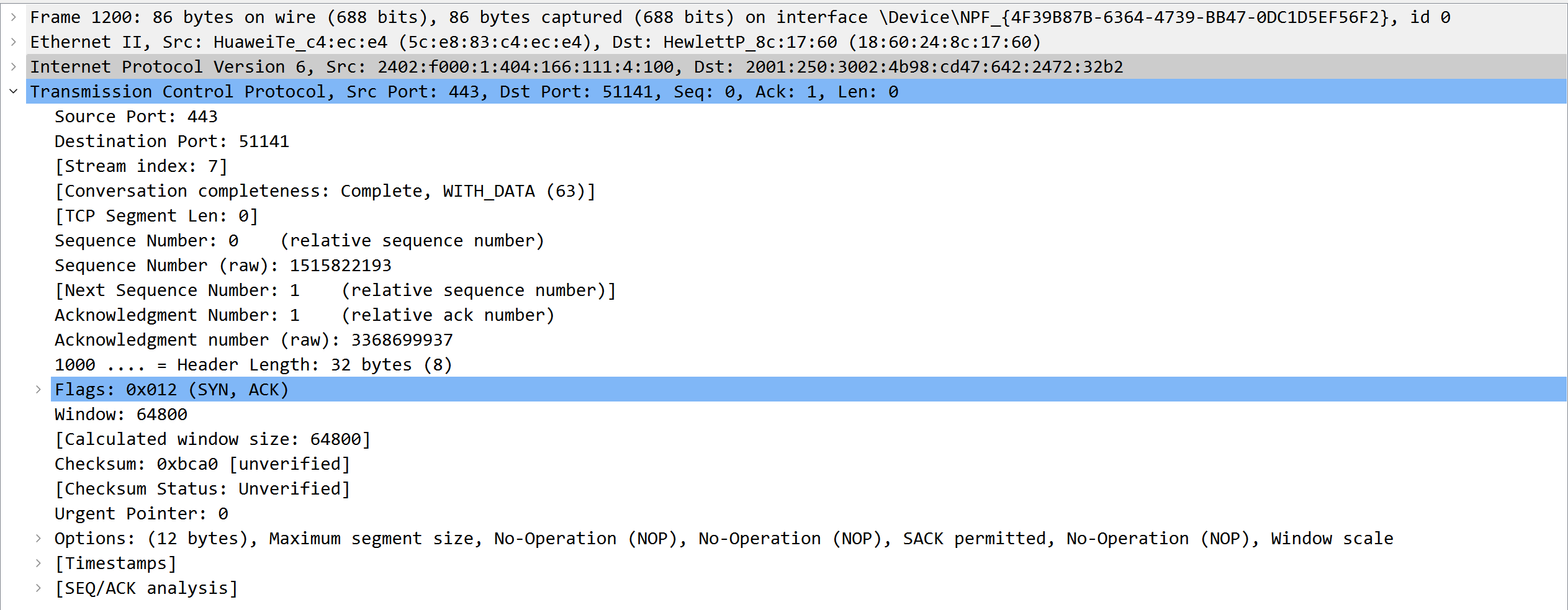
**标志位（Flags）： 0x002（SYN）这是一个SYN标志位，表示发起一个连接请求。**

**窗口大小（Window Size）： 64800表示接收方的窗口大小，即可以接收的字节数。在这个包中，窗口大小为64800字节。**

**校验和（Checksum）： 0xe442TCP头部和数据的校验和，用于检测数据在传输过程中是否发生了错误。在这个包中，校验和值是0xe442。**

**这个包是一个TCP的连接建立请求（SYN），它的目的是发起一个与目标端口为443的服务建立连接的请求,这是TCP连接的初始阶段。**

**1200[SYN,ACK]:**



**分析:**

**源端口（Source Port）： 443**

**目标端口（Destination Port）： 51141**

**序列号（Sequence Number）： 0这是TCP连接中第一个数据字节的序列号。在这个包中，它的值是0，表示这是一个初始的序列号。**

**确认号（Acknowledgment Number）： 1表示对方期望接收的下一个序列号。在这个SYN-ACK包中，确认号是1，表示已经接收到客户端的SYN请求，并且期望接收客户端的下一个数据字节。**

**头部长度（Header Length）： 32 bytes（8）**

**TCP头部长度的单位是32位字（4字节），这里是32 bytes，表示8个32位字，即8 \* 4 = 32字节。**

**标志位（Flags）： 0x012（SYN, ACK）**

**这是一个SYN和ACK标志位，表示建立连接并确认收到客户端的SYN请求。**

**窗口大小（Window Size）： 64800**

**表示接收方的窗口大小，即可以接收的字节数。在这个包中，窗口大小为64800字节。**

**校验和（Checksum）： 0xbca0**

**TCP头部和数据的校验和，用于检测数据在传输过程中是否发生了错误。在这个包中，校验和值是0xbca0。**

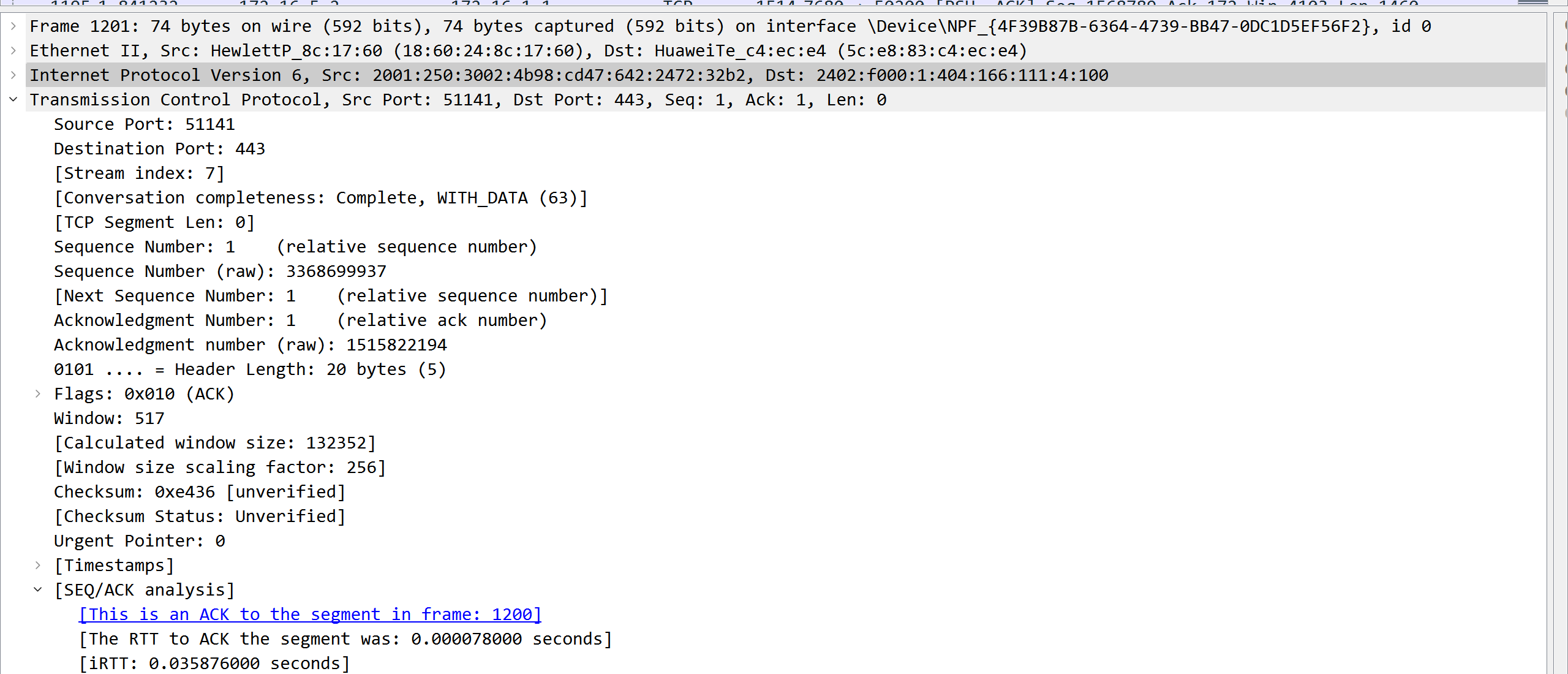
**时间戳（Timestamps）： 启用了时间戳选项。**

**还有几个NOP（No-Operation）选项。**

**[SEQ/ACK analysis]：**

**This is an ACK to the segment in frame: 1178说明这是对帧1178中的TCP段的确认。**

**这个包是服务器回应客户端的TCP连接建立请求的SYN-ACK包。服务器告知客户端已经接收到了连接请求，并确认客户端的初始序列号。1201[SYN]:**



**分析:**

**源端口（Source Port）： 51141**

**目标端口（Destination Port）： 443**

**序列号（Sequence Number）： 1这是TCP连接中的第一个数据字节的序列号。在这个包中，它的值是1，表示这是客户端的下一个要发送的数据字节。**

**确认号（Acknowledgment Number）： 1表示对方期望接收的下一个序列号。在这个ACK包中，确认号是1，表示客户端已经接收到了服务器的SYN-ACK，并确认了服务器的初始序列号。**

**头部长度（Header Length）： 20 bytes（5）**

**TCP头部长度的单位是32位字（4字节），这里是20 bytes，表示5个32位字，即5 \* 4 = 20字节。**

**标志位（Flags）： 0x010（ACK）**

**这是一个ACK标志位，表示确认收到了对方的数据。这是三次握手的最后一个步骤。**

**窗口大小（Window Size）： 517**

**表示接收方的窗口大小，即可以接收的字节数。在这个包中，窗口大小为517字节。**

**校验和（Checksum）： 0xe436**

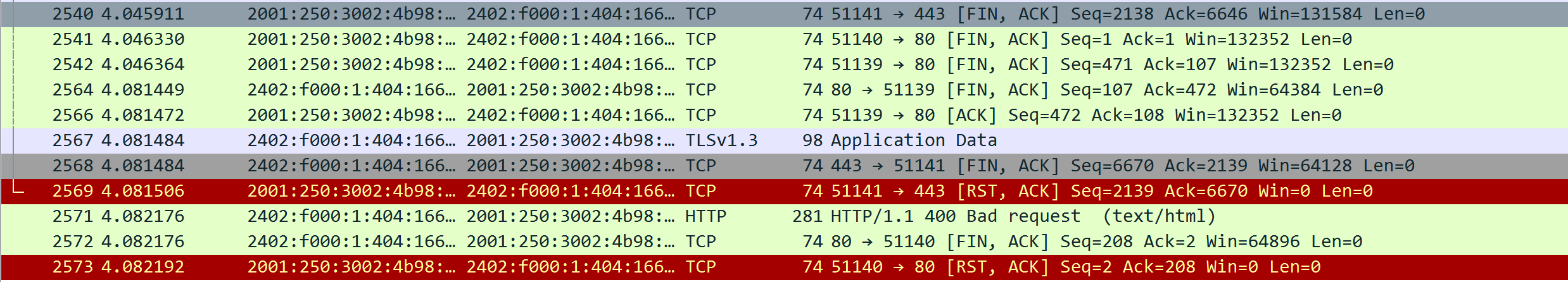
**TCP头部和数据的校验和，用于检测数据在传输过程中是否发生了错误。在这个包中，校验和值是0xe436。**

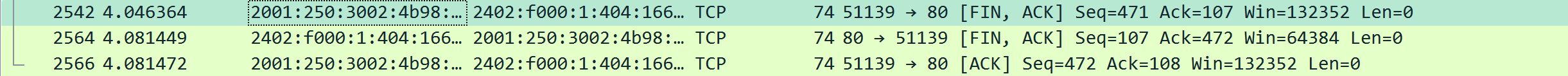
**[SEQ/ACK analysis]：**

**This is an ACK to the segment in frame: 1200说明这是对帧1200中的TCP段的确认。**

**这个包是客户端对服务器的SYN-ACK的确认，完成了TCP连接的三次握手过程。双方现在建立了连接，可以开始进行数据的传输。**

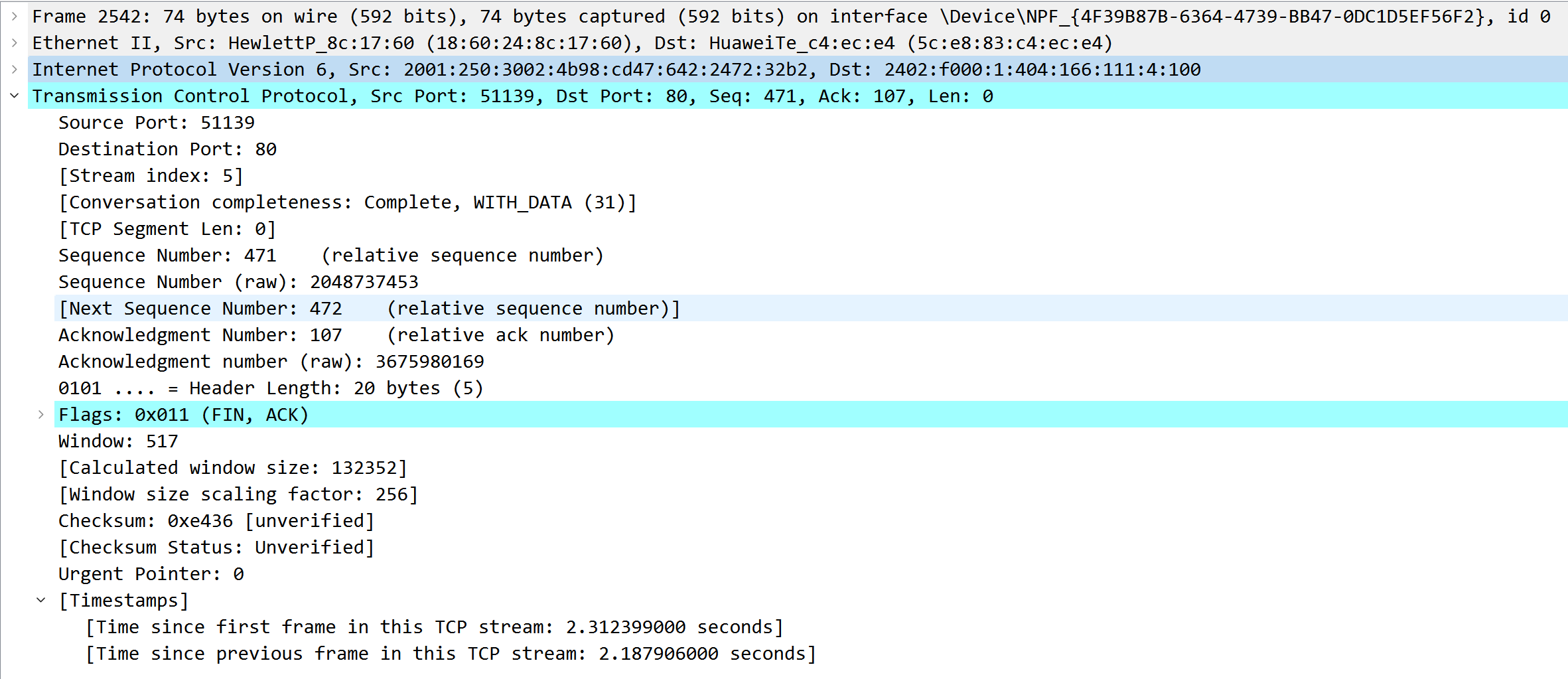
### TCP取消连接的四次挥手





**可以看到只有三条报文，这是因为四次挥手的第二和第三步被捕获到同一个报文中**

**第一次挥手:**



**分析:**

**源端口（Source Port）： 51139**

**目标端口（Destination Port）： 80**

**序列号（Sequence Number）： 471）这是TCP连接中的第471个数据字节的序列号。在这个包中，它的值是471，表示这是客户端的下一个要发送的数据字节。**

**确认号（Acknowledgment Number）： 107表示对方期望接收的下一个序列号。在这个FIN+ACK包中，确认号是107，表示服务器已经接收到了客户端的FIN请求，并且期望接收客户端的下一个数据字节。**

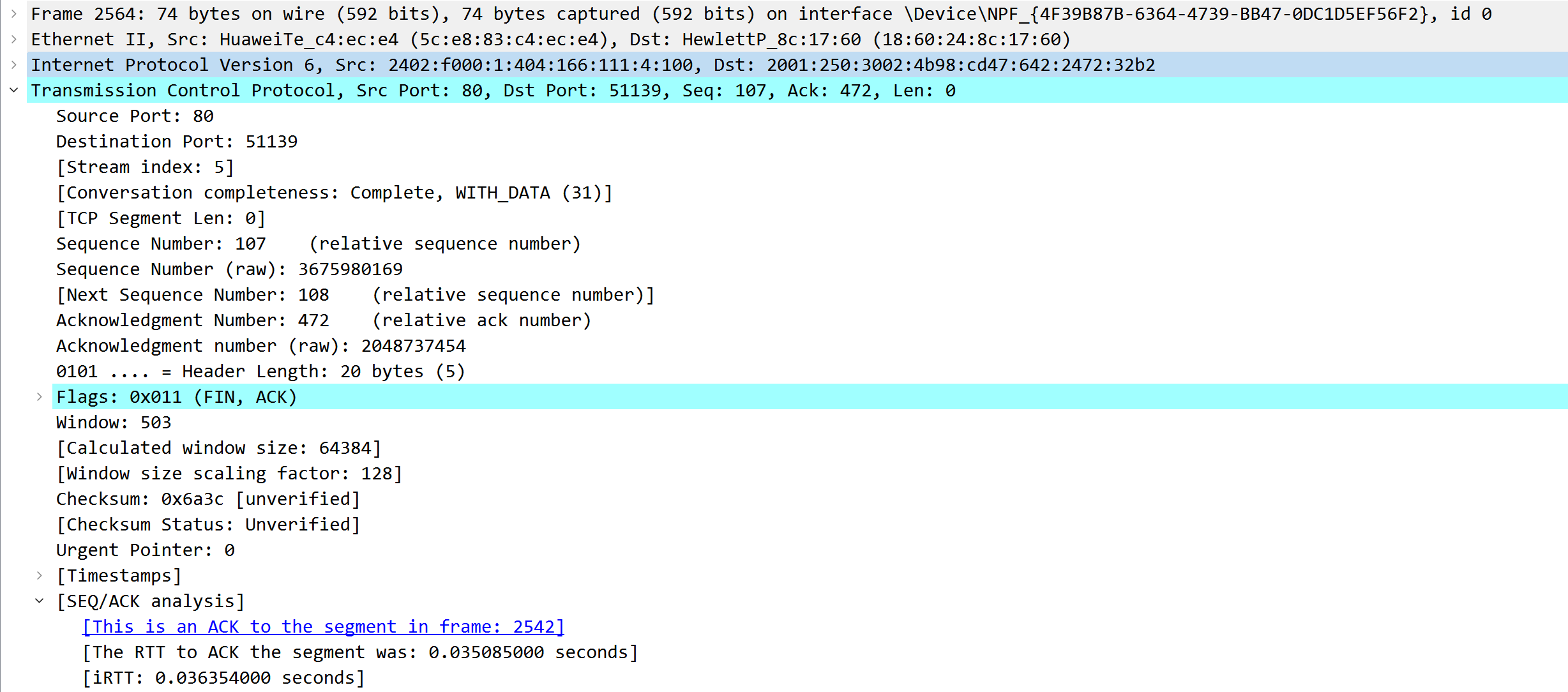
**头部长度（Header Length）： 20 bytes（5）TCP头部长度的单位是32位字（4字节），这里是20 bytes，表示5个32位字，即5 \* 4 = 20字节。**

**标志位（Flags）： 0x011（FIN, ACK）这是一个FIN和ACK标志位，表示发送方已经完成数据的发送，而且确认收到了对方的数据。这是四次挥手的第三步。**

**窗口大小（Window Size）： 517表示接收方的窗口大小，即可以接收的字节数。在这个包中，窗口大小为517字节。**

**校验和（Checksum）： 0xe436TCP头部和数据的校验和，用于检测数据在传输过程中是否发生了错误。在这个包中，校验和值是0xe436。**

**这个包是客户端发送的带有FIN和ACK标志的TCP报文，表示客户端已经完成数据的发送，并请求断开连接。这是四次挥手的第一步。**



**分析：**

**源端口（Source Port）： 80**

**目标端口（Destination Port）： 51139**

**序列号（Sequence Number）： 107这是TCP连接中的第107个数据字节的序列号。在这个包中，它的值是107，表示这是服务器的下一个要发送的数据字节。**

**确认号（Acknowledgment Number）： 472表示对方期望接收的下一个序列号。在这个FIN+ACK包中，确认号是472，表示客户端已经接收到了服务器的FIN请求，并且期望接收服务器的下一个数据字节。**

**头部长度（Header Length）： 20 bytes（5）**

**TCP头部长度的单位是32位字（4字节），这里是20 bytes，表示5个32位字，即5 \* 4 = 20字节。**

**标志位（Flags）： 0x011（FIN, ACK）**

**这是一个FIN和ACK标志位，表示发送方已经完成数据的发送，而且确认收到了对方的数据。这是四次挥手的第三步。**

**窗口大小（Window Size）： 503**

**表示接收方的窗口大小，即可以接收的字节数。在这个包中，窗口大小为503字节。**

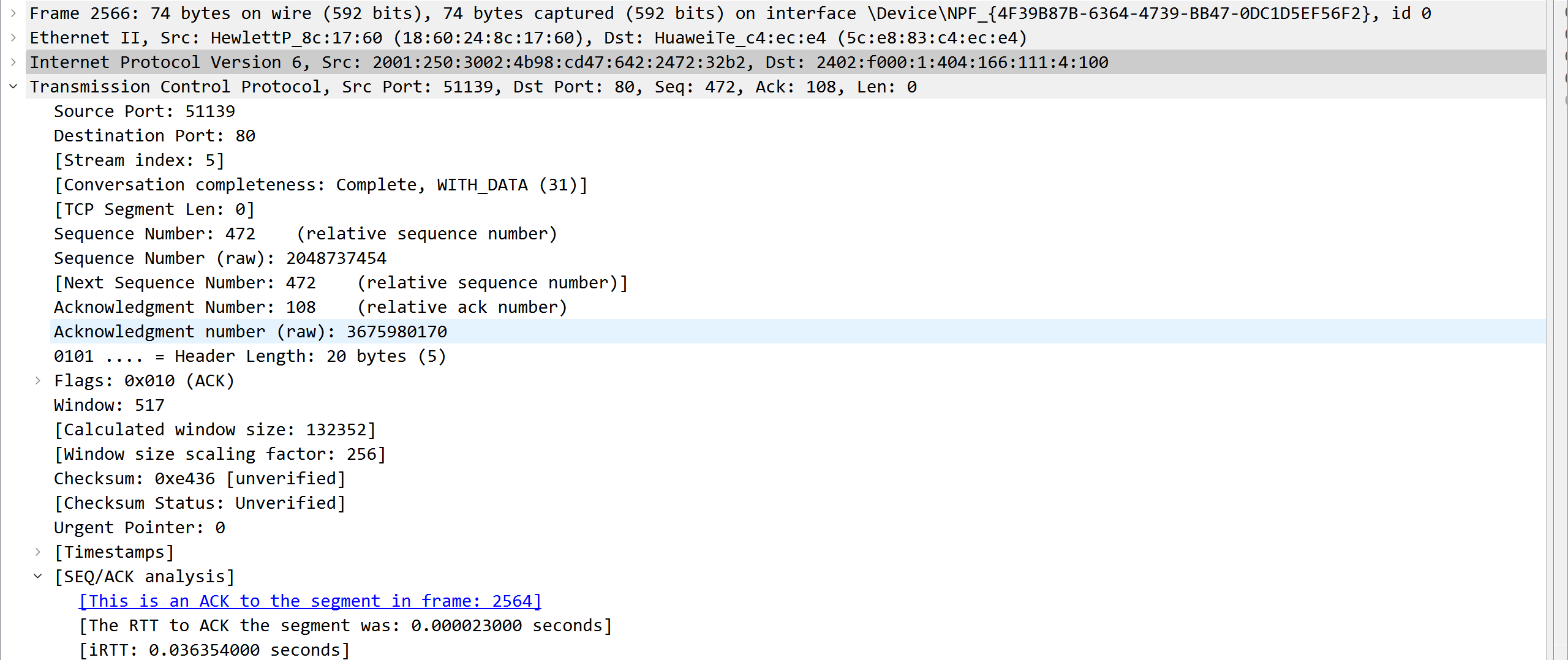
**校验和（Checksum）： 0x6a3c**

**TCP头部和数据的校验和，用于检测数据在传输过程中是否发生了错误。在这个包中，校验和值是0x6a3c。**

**[SEQ/ACK analysis]：**

**This is an ACK to the segment in frame: 2542说明这是对帧2542中的TCP段的确认。**

**这个包是服务器回应客户端的带有FIN和ACK标志的TCP报文，表示服务器已经完成数据的发送，并请求断开连接。这是四次挥手的第二步和第三步。**



**分析:**

**源端口（Source Port）： 51139**

**目标端口（Destination Port）： 80**

**序列号（Sequence Number）： 472这是TCP连接中的第472个数据字节的序列号。在这个包中，它的值是472，表示这是客户端的下一个要发送的数据字节。**

**确认号（Acknowledgment Number）： 108表示对方期望接收的下一个序列号。在这个ACK包中，确认号是108，表示服务器已经接收到了客户端的FIN请求，并且期望接收客户端的下一个数据字节。**

**头部长度（Header Length）： 20 bytes（5）**

**TCP头部长度的单位是32位字（4字节），这里是20 bytes，表示5个32位字，即5 \* 4 = 20字节。**

**标志位（Flags）： 0x010（ACK）**

**这是一个ACK标志位，表示确认收到了对方的数据。这是四次挥手的最后一步。**

**窗口大小（Window Size）： 517**

**表示接收方的窗口大小，即可以接收的字节数。在这个包中，窗口大小为517字节。**

**校验和（Checksum）： 0xe436**

**TCP头部和数据的校验和，用于检测数据在传输过程中是否发生了错误。在这个包中，校验和值是0xe436。**

**[SEQ/ACK analysis]：**

**This is an ACK to the segment in frame: 2564说明这是对帧2564中的TCP段的确认。**

**这个包是客户端回应服务器的带有ACK标志的TCP报文，表示客户端确认了服务器的关闭请求。这是四次挥手的最后一步。双方都完成了连接的关闭，可以安全地释放资源。**

### 按实际测试，网页自动跳转运行https协议。请分析其应用层与传输层协议的消息交换过程。

**当网页自动跳转到HTTPS协议时，涉及应用层的HTTP协议和传输层的TLS/SSL协议。以下是应用层与传输层协议的消息交换过程：**

**①应用层（HTTP）：**

**客户端发起HTTP请求： 初始阶段，客户端通过HTTP协议发起一个HTTP请求，通常是使用HTTP的GET请求。**

**服务器回应重定向： 服务器收到HTTP请求后，如果需要将连接升级到HTTPS，会在HTTP响应头中添加一个重定向（Redirect）指令。**

**②应用层（HTTP）：**

**客户端再次发起HTTP请求： 客户端收到服务器的HTTP响应后，根据重定向指令再次发起一个HTTP请求，这次请求的目标是服务器指定的HTTPS地址。**

**③传输层（TLS/SSL）：**

**TLS握手过程： 在发起HTTPS请求时，客户端和服务器进行TLS握手。**

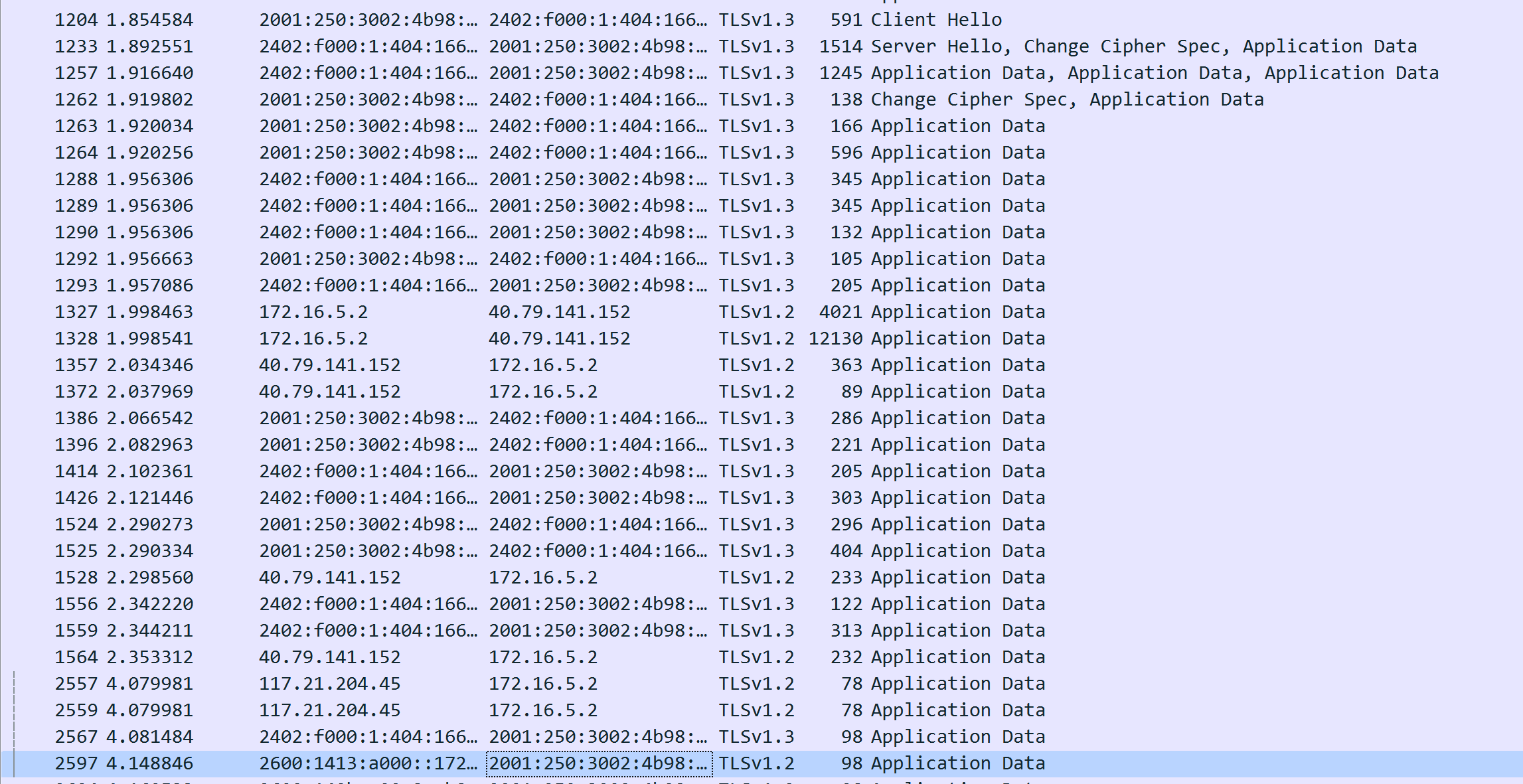
**④传输层（TLS/SSL）：**

**客户端和服务器发送ChangeCipherSpec消息，表示接下来的通信将使用新协商的密钥。客户端和服务器通过TLS记录层传输应用层数据，这些数据在传输过程中被加密和完整性保护。**

**关于应用HTTP的分析已展示过了,接下来重点分析传输层（TLS/SSL）**

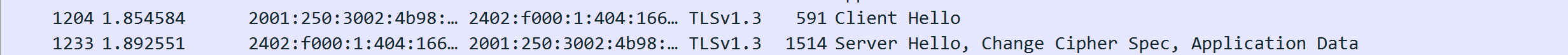
### **TLS**

**可以看到使用的是TLSv1.3和TLS1.2**

****

**TLS（Transport Layer Security，安全传输层)，TLS是建立在传输层TCP协议之上的协议，服务于应用层，它的前身是SSL（Secure Socket Layer，安全套接字层），它实现了将应用层的报文进行加密后再交由TCP进行传输的功能。**

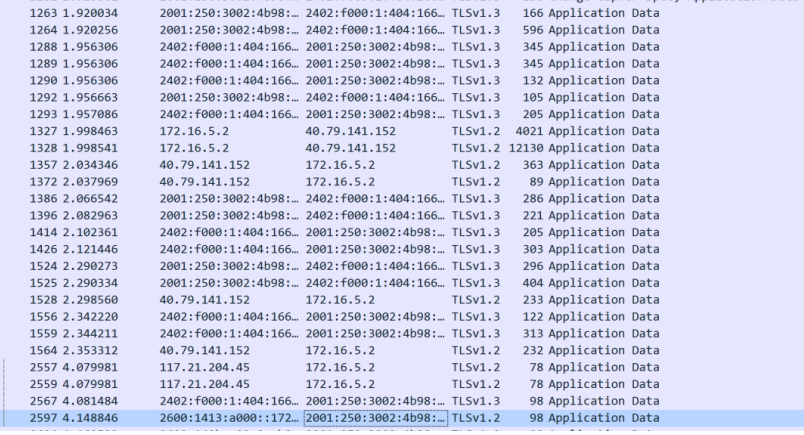
**举例来分析:**

****

**ClientHello： 客户端向服务器发起握手请求，提供支持的加密算法和其他相关信息。**

**ServerHello： 服务器选择一个加密算法和其他参数，并向客户端发送响应。**

**ChangeCipherSpec(准备切换密码) 服务器端告诉客户端表示接下来的通信将使用新协商**



**Application Data： 客户端和服务器通过TLS记录层传输应用层数据，这些数据被加密和完整性保护。**

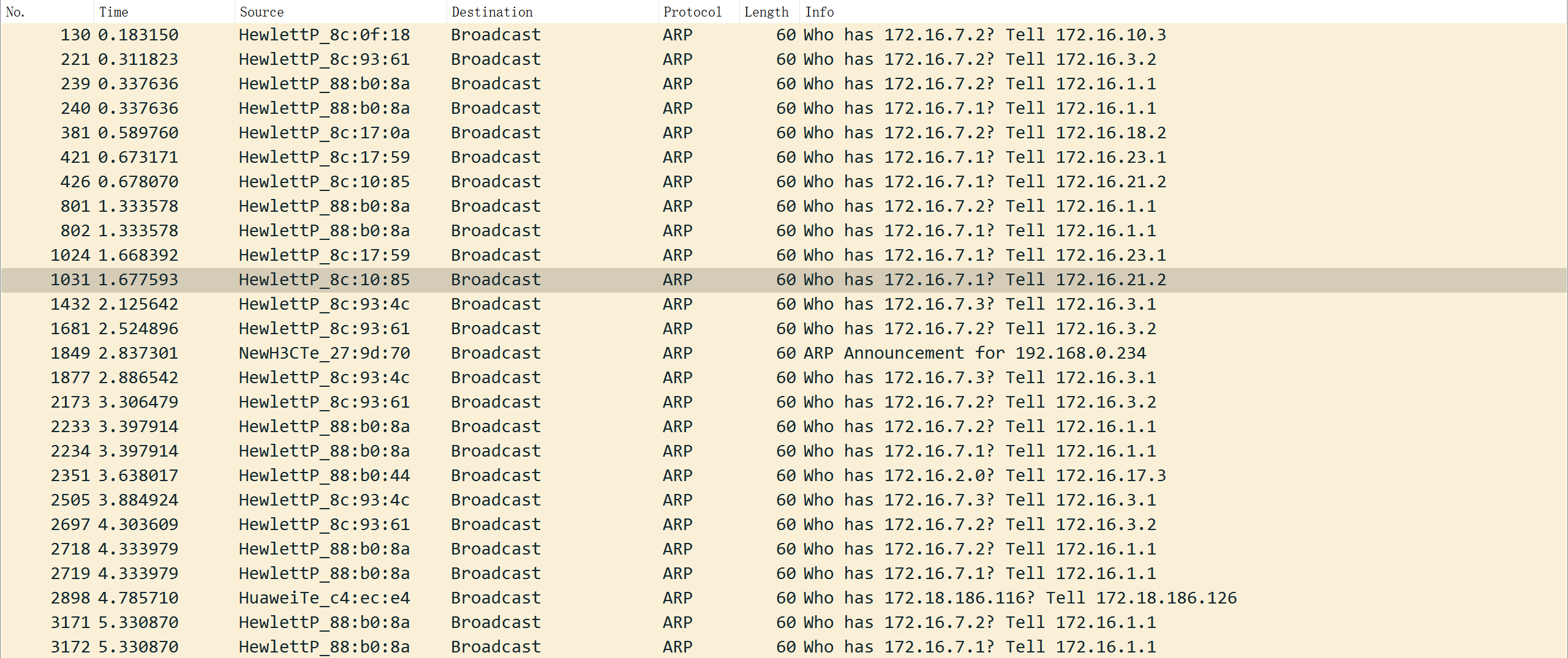
#### 分析网络层运行的协议。依据网络数据说明相关协议的基本运行情况、顺序及其作用。

### ARP

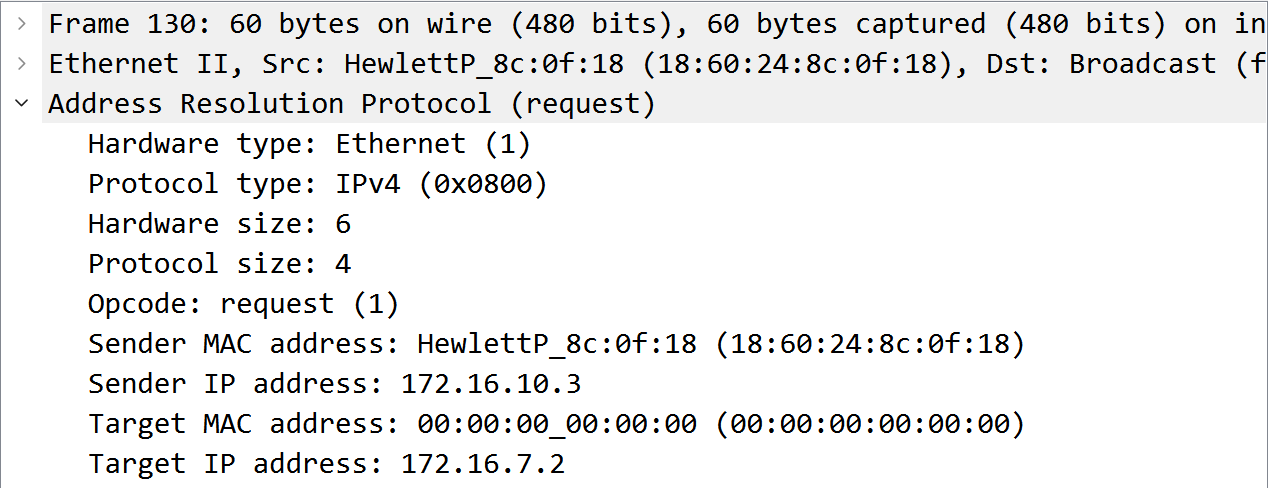
**ARP网络协议是用于将IP地址映射到物理硬件地址（如MAC地址）。在TCP/IP网络中，数据包在发送之前需要知道目标设备的物理地址，而ARP正是负责解决这个问题的协议。**

**当一台计算机需要发送数据到另一台计算机时，它首先检查如果ARP表中已经有了目标IP地址的映射关系，计算机就可以直接使用这个映射关系，而无需发送ARP请求 。 如果ARP表中没有目标IP地址的映射关系，计算机会发送一个ARP请求广播到本地网络上的所有设备，询问目标IP地址对应的MAC地址。目标设备收到ARP请求后，会发送一个ARP应答包，包含自己的MAC地址。**

**查看本次抓包所抓到的ARP数据包:**

****

**例如分析序号为130的ARP报文:**

****

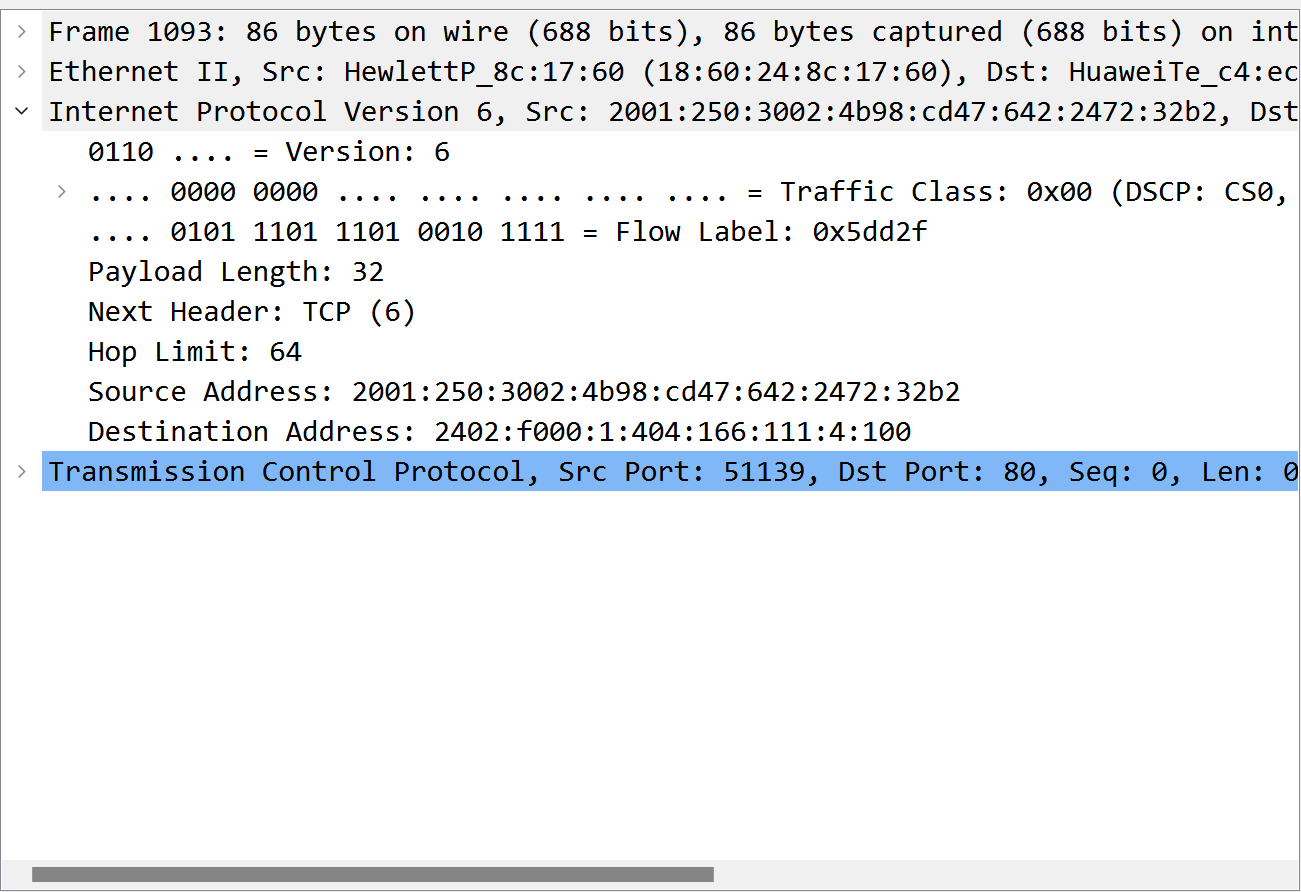
**可以看到在这个特定的 ARP 请求中，设备 with IP 地址 172.16.10.3 想要知道 172.16.7.2 对应的 MAC 地址。ARP 请求中的 "Who has 172.16.7.2? Tell 172.16.10.3" 表示了这个查询，172.16.10.3向广播发送这个ARP请求报文,广播向子网广播这条报文，对应的172.168.7.2收到后将发送回一个对应映射的响应ARP分组。**

### IP

**IP协议提供了网络层的基本通信机制，负责定位设备和在网络上传递数据包。不同的协议在IP的基础上运行，构建了不同的网络服务。**

**下面选取TCP连接时发送SYN的IP数据信息分析:**





**IPv6基本信息：**

**IPv6地址：**

**源地址（Source Address）： 2001:250:3002:4b98:cd47:642:2472:32b2**

**目标地址（Destination Address）： 2402:f000:1:404:166:111:4:100**

**版本（Version）： 表明这是IPv6协议。**

**Traffic Class（流量类别）： 用于指定数据包的服务质量，此处为0x00。**

**Flow Label（流标签）： 用于标识数据包所属的流，便于路由器进行流级别的处理。**

**Payload Length（负载长度）： 负载的长度为32字节，即IPv6头部后面的TCP头和数据部分。**

**Next Header（下一个协议类型）： 表示紧随IPv6头部的下一个协议，这里是TCP（协议号为6）。**

**Hop Limit（跳数限制）： 跳数限制（TTL），此处为64，表示数据包最多可以经过64个路由器。**

**可以看到TCP协议则建立在IP协议之上，提供了更高层次的、可靠的端到端通信机制，确保数据的有序、可靠传输。在互联网通信中，IP和TCP通常一起使用，形成了TCP/IP协议栈，为应用层提供了可靠的网络通信服务。**

### OSPF

**查看捕捉到的OSPF报文:**

#### 

**点击查看详细信息:**

#### 

**OSPF头部：**

**版本：3**

**消息类型：Hello Packet (1)**

**包长度：36**

**源OSPF路由器：10.44.185.254**

**区域ID：0.0.0.4**

**校验和：0xa631**

**实例ID：IPv6单播AF (0)**

**保留字段：00**

**OSPF Hello Packet：**

**接口ID：115**

**路由器优先级：1**

**选项：0x000013，标志包括 R（Router）、E（External Routing）、V6（IPv6）**

**Hello间隔：10**

**路由器死亡间隔：40**

**指定路由器：10.44.185.254**

**备用指定路由器：0.0.0.0**

**这个OSPF数据包是一个IPv6环境下的OSPF Hello消息，用于OSPF协议的邻居关系维护。OSPF是一种路由协议，用于在IP网络中计算最短路径，并支持IPv6。该消息表明这个设备正在通过OSPF协议与其他设备进行Hello协商，以维护路由信息。**

#### 分析数据链路层运行的协议。依据网络数据说明相关协议的基本**运行情况、顺序及其作用。**

**在通过ARP协议得到对应的物理地址后，链路层利用以太网协议数据帧通过MAC地址进行寻址和传输。**

### Ethernet

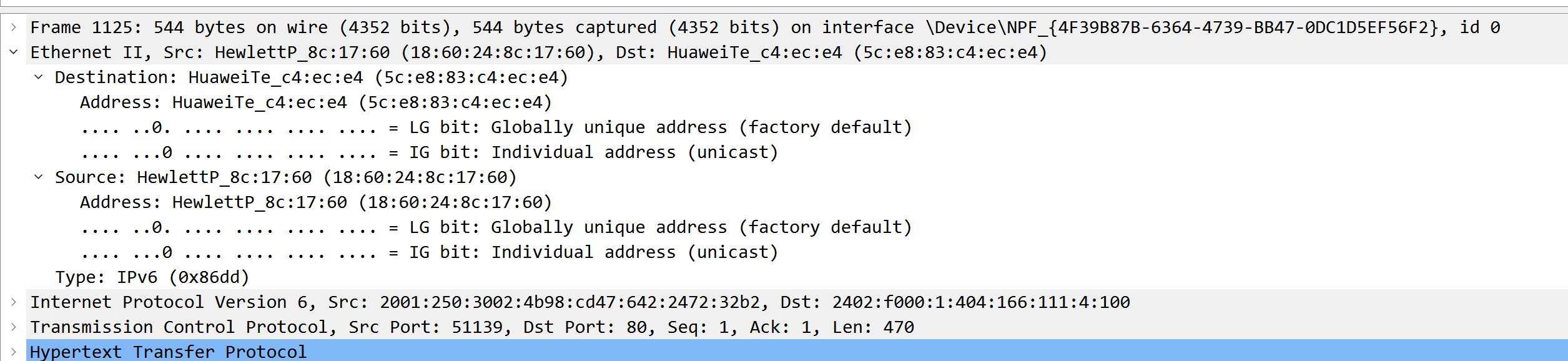
**以太网协议（Ethernet）：**

**首先数据被封装到以太网帧中，帧包括源和目标MAC地址、数据部分以及控制字段。**

**然后帧通过物理介质（例如，网线）传输到网络中的目标设备。**

**最后目标设备接收帧后，通过检查目标MAC地址确定是否接收帧，然后解封装数据。**

**例如分析HTTP请求的以太网协议:**



**目的地地址（Destination Address）：**

**地址：HuaweiTe\_c4:ec:e4（5c:e8:83:c4:ec:e4）**

**LG位：0（Globally unique address - 全局唯一地址，出厂默认设置）**

**IG位：0（Individual address - 个体地址，单播）**

**源地址（Source Address）：**

**地址：HewlettP\_8c:17:60（18:60:24:8c:17:60）**

**LG位：0（Globally unique address - 全局唯一地址，出厂默认设置）**

**IG位：0（Individual address - 个体地址，单播）**

**帧类型（Type）：**

**类型值：IPv6（0x86dd）表示该以太网帧中封装的上层协议是IPv6。**

**该以太网II帧源地址是 HewlettP\_8c:17:60，目标地址是 HuaweiTe\_c4:ec:e4，而上层协议是IPv6。这种类型的帧通常用于在局域网中进行数据帧的传输。**