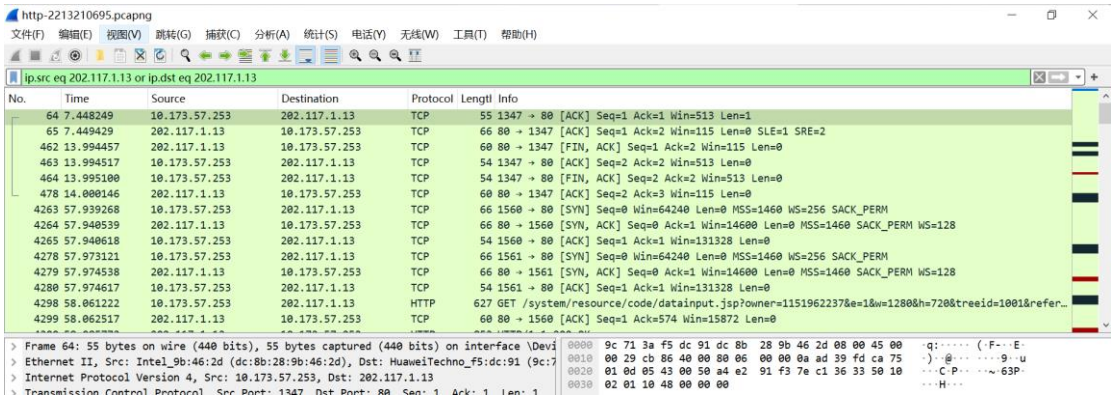
WIRESHARK 抓包

```
C:\Users\23363>ping www.xjtu.edu.cn

正在 Ping www.xjtu.edu.cn [202.117.1.13] 具有 32 字节的数据:
来自 202.117.1.13 的回复: 字节=32 时间=1ms TTL=61
来自 202.117.1.13 的回复: 字节=32 时间=1ms TTL=61
来自 202.117.1.13 的回复: 字节=32 时间=1ms TTL=61
来自 202.117.1.13 的回复: 字节=32 时间=1ms TTL=61

202.117.1.13 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 = 1ms, 最长 = 1ms, 平均 = 1ms

C:\Users\23363>
```

通过 Ping 查出 www.xjtu.end.cn 的 ip 地址为
www.xjtu.edu.cn 的 IP:202.117.1.13

抓到的包



Http 分析

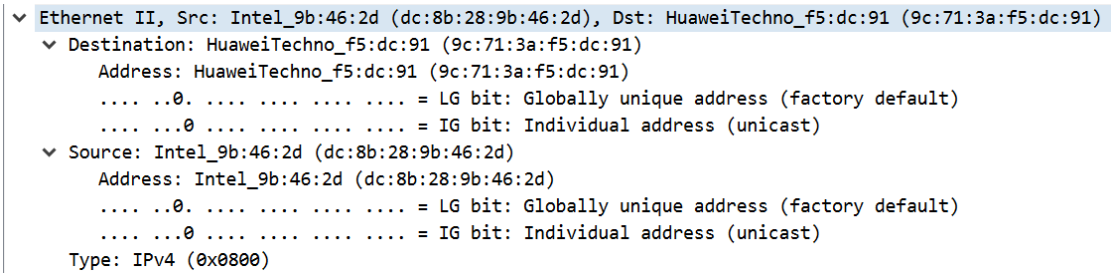
Http 请求报文

物理层数据帧详细概要

可以看出以下信息：

为 64 号帧，线路字节 440bit，实际捕获 55bit，端口号为 0 号端口，端口在计算机中明明为 WLAN 2，采用的封装方式为 Ethernet(1)，捕获的时间为 2024 年 1 月 5 号，09：31：53 左右。以及一些和帧有关的时间信息。该帧没有被标记，也没有被忽略。该帧的协议为 http，端口号为 80。

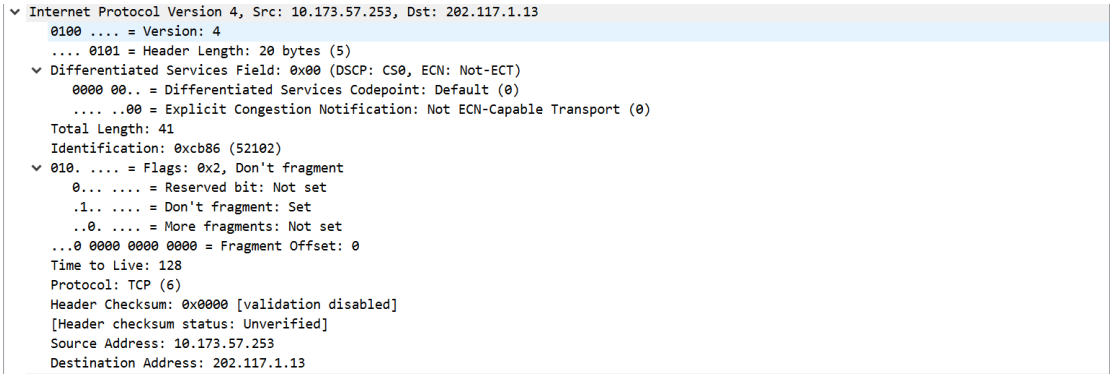
数据链路层以太帧的首部信息：



目的地的 MAC 地址: 9c:71:3a:f5:dc:91

源 MAC 地址:dc:8b:28:9b:46:2d

网络层 ip 包首部信息：



第 1 行看出使用了 IPv4 协议

首部长 20 字节。IP 包共长 41 字节，标记字段为 52102，生存周期为 128。

源地址为 10.173.57.253

目的地址为 202.117.1.13

传输层数据包首部信息

```
Transmission Control Protocol, Src Port: 1347, Dst Port: 80, Seq: 1, Ack: 1, Len: 1
    Source Port: 1347
    Destination Port: 80
    [Stream index: 8]
  > [Conversation completeness: Incomplete (28)]
    [TCP Segment Len: 1]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 2766311923
    [Next Sequence Number: 2    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 2126591539
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
    Window: 513
    [Calculated window size: 513]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x1048 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
    TCP payload (1 byte)
```

可以看出以下信息：
源端口号：1347
目的端口号：80
流量控制端口号：513

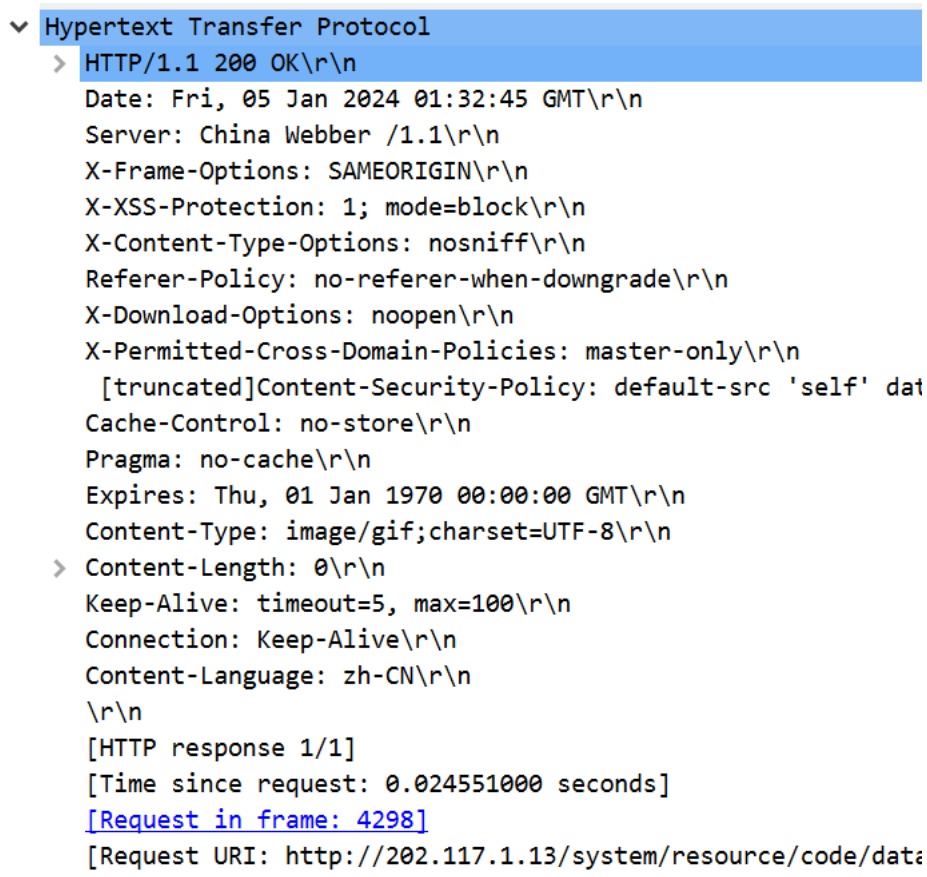
应用层分析：

```
Hypertext Transfer Protocol
  > GET /system/resource/code/datainput.jsp?owner=1151962237&e=1&w=1280&h=720&treeid=1001&refer=&pagename=L2luZGV4LmpzcA%3D%3D&newsid=-1 HTTP/1.1\r\n
    Host: 202.117.1.13\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/120.0.0.0 Safari/537.36 Edg/120.0.0.0\r\n
    Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8\r\n
    Referer: http://202.117.1.13/\r\n
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6\r\n
  > Cookie: JSESSIONID=AE07DE40B844BBE1C14A0067706BB739\r\n
    \r\n
    [Full request URI: http://202.117.1.13/system/resource/code/datainput.jsp?owner=1151962237&e=1&w=1280&h=720&treeid=1001&refer=&pagename=L2luZGV4…
    [HTTP request 1/1]
    [Response in frame: 4300]
```

即具体的 http 报文，
给出了请求主机的地址：202.117.1.13
给出了一些自身信息：这是一个在 Windows 10 上运行的 64 位系统的浏览器，基于 WebKit 引擎，其浏览器核心为 Chrome（版本号 120.0.0.0）和 Microsoft Edge（版本号也是 120.0.0.0）。客户端支持两种压缩算法：gzip 和 deflate。如果可能的话，首选返回中文内容。如果中文不可用，可以考虑返回英语内容，依次考虑英国英语和美国英语。以及 cookie。

http 应答报文：

前面的物理层，数据链路层，网络层，传输层，http 请求报文大同小异，直接看应用层 http 报文。



第一行 200 ok 表示准备好了，后面是一些设定，与请求报文相呼应。

http 协议工作过程：

1. 建立连接：客户端与服务器之间通过 TCP（通常是 TCP）建立连接。默认的 HTTP 端口是 80，或者使用安全的 HTTPS 协议时为 443。
2. 发送请求：客户端发送一个 HTTP 请求给服务器。这个请求通常包含以下部分：

   请求行：包含请求的方法（GET、POST 等）、请求的资源路径和使用的协议版本。

   请求头部：包含一系列的键值对，描述客户端的信息、所需的文档类型、支持的压缩算法等信息。

   请求主体：仅在使用 POST、PUT 等方法时存在，包含发送给服务器的数据。

3. 处理请求：服务器接收并解析请求，然后根据请求执行相应的操作。这可能涉及到从文件系统中获取文件、查询数据库，或执行其他服务器端的逻辑。
4. 发送响应：服务器生成一个 HTTP 响应，包含以下部分：

   状态行（Status Line）：包含协议版本、状态码和状态消息。

   响应头部（Response Headers）：与请求头类似，包含服务器信息、响应的文档类型、

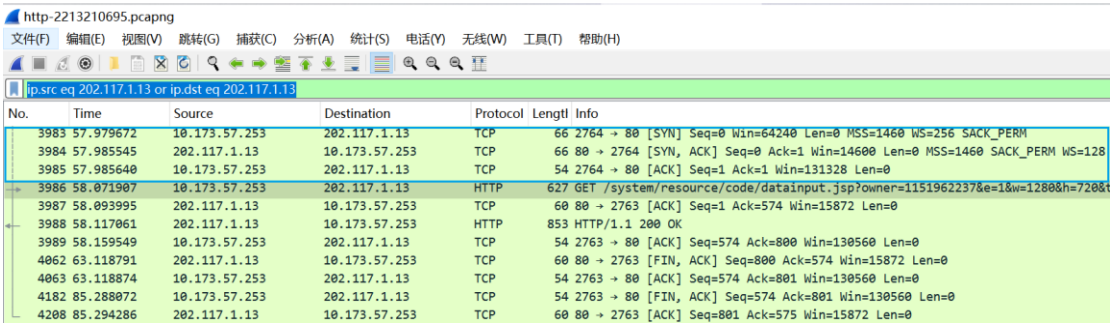
日期等信息。

响应主体（Response Body）：包含实际的数据，比如 HTML 页面、图像、文本等。

5. 传输数据：服务器将响应数据通过建立的 TCP 连接传输到客户端。
6. 关闭连接：客户端接收到响应后，如果没有指定连接保持活动（HTTP/1.1 默认保持活动，除非指定关闭），则客户端或服务器将关闭连接。

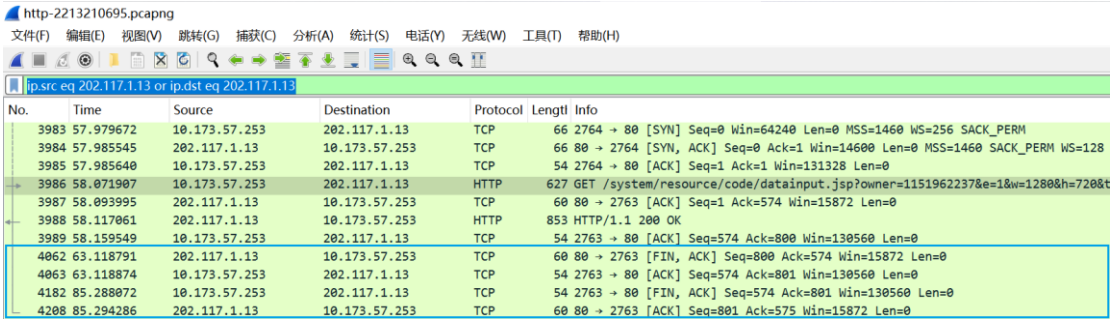
TCP 分析：

TCP 建立连接



tcp 建立时由客户端端口 2764 向服务器 80 端口发送第一次握手 SYN=1 seq=0
主机端返回第二次握手 SYN=1 ACK=1 seq=0 ack=1
客户端收到后第三次握手 ACK=1，seq=1，ack=1
至此，tcp 成功建立连接

TCP 释放链接



主机发起请求 FIN,ACK（第一次挥手），
本机接收到并返回两个包，一个为 ACK（第二次挥手），
另一个为 FIN,ACK（第三次挥手），
最后主机发一个 ACK 返回给本机（第四次挥手）然后结束。

TCP 过程

1. 建立连接：

客户端向服务器发送 SYN（同步）报文：客户端选择一个随机的初始序列号（ISN）并发

送一个带有 SYN 标志的 TCP 报文给服务器。

服务器收到 SYN 报文并回应：服务器接收到客户端的 SYN 报文后，选择自己的随机 ISN，并发送一个带有 SYN 和 ACK（确认）标志的报文给客户端。

客户端发送 ACK 报文：客户端收到服务器的响应后，发送一个带有 ACK 标志的报文给服务器，完成连接的建立。

这个过程称为三次握手，建立了双方的连接，确保双方都能够接收和发送数据。

2. 数据传输：

客户端和服务器通过已建立的连接传输数据。数据被分割成 TCP 段，并分别被封装成 TCP 报文，然后通过网络传输。

3. 关闭连接：

客户端发送 FIN 报文：当一方（通常是客户端）希望关闭连接时，它发送一个带有 FIN（结束）标志的 TCP 报文。

服务器收到 FIN 并回应：服务器收到 FIN 后，发送一个带有 ACK 标志的报文给客户端，表示确认收到关闭请求。

服务器发送 FIN 报文：服务器也 7 希望关闭连接，因此发送一个带有 FIN 标志的报文给客户端。

客户端收到 FIN 并回应：客户端收到服务器的 FIN 后，发送一个带有 ACK 标志的报文，确认收到服务器的关闭请求。

这个过程称为四次挥手，确保双方都完成了数据的传输并准备好关闭连接。

TCP 通过这样的握手和挥手过程，提供了可靠的连接，保证了数据的完整性和有序性。这种可靠性建立在序列号、确认和重传机制等基础上，确保数据在传输过程中不会丢失、损坏或乱序。