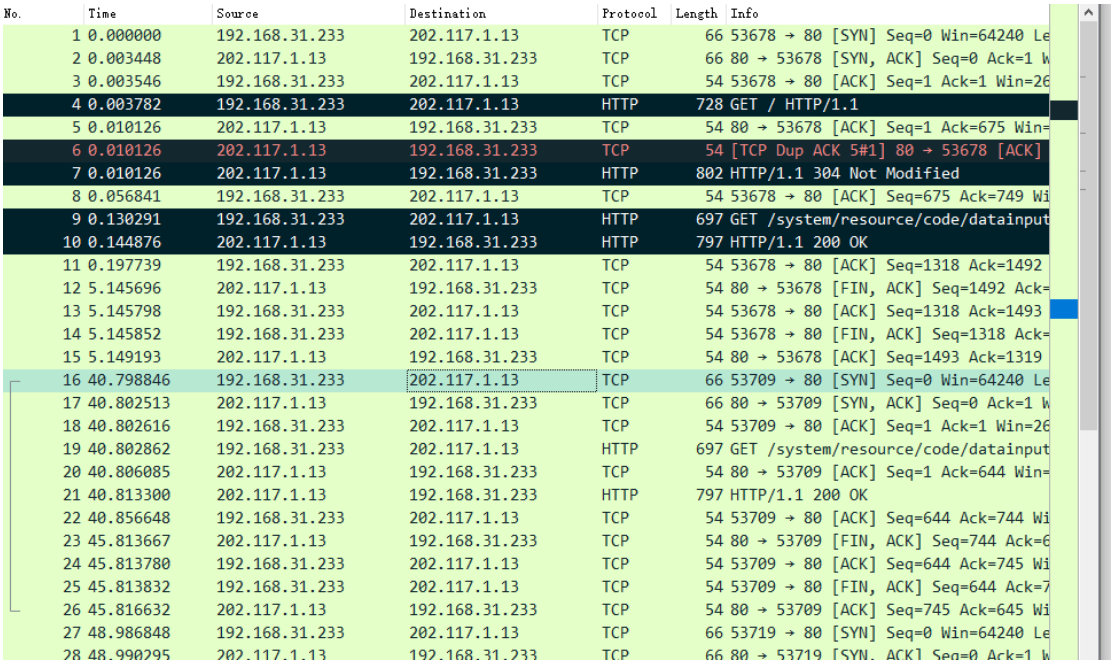
分析协议 HTTP 分组嗅探器（wireshark）

访问 www.xjtu.edu.cn(202.117.1.13)，过滤器设置为 ip.addr == 202.117.1.13，得到如下图所示的结果。其中部分 HTTP 报文已标黑。



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.31.233	202.117.1.13	TCP	66	53678 → 80 [SYN] Seq=0 Win=64240 Le
2	0.003448	202.117.1.13	192.168.31.233	TCP	66	80 → 53678 [SYN, ACK] Seq=0 Ack=1 W
3	0.003546	192.168.31.233	202.117.1.13	TCP	54	53678 → 80 [ACK] Seq=1 Ack=1 Win=26
4	0.003782	192.168.31.233	202.117.1.13	HTTP	728	GET / HTTP/1.1
5	0.010126	202.117.1.13	192.168.31.233	TCP	54	80 → 53678 [ACK] Seq=1 Ack=675 Win=
6	0.010126	202.117.1.13	192.168.31.233	TCP	54	[TCP Dup ACK 5#1] 80 → 53678 [ACK]
7	0.010126	202.117.1.13	192.168.31.233	HTTP	802	HTTP/1.1 304 Not Modified
8	0.056841	192.168.31.233	202.117.1.13	TCP	54	53678 → 80 [ACK] Seq=675 Ack=749 Wi
9	0.130291	192.168.31.233	202.117.1.13	HTTP	697	GET /system/resource/code/datainput
10	0.144876	202.117.1.13	192.168.31.233	HTTP	797	HTTP/1.1 200 OK
11	0.197739	192.168.31.233	202.117.1.13	TCP	54	53678 → 80 [ACK] Seq=1318 Ack=1492
12	5.145696	202.117.1.13	192.168.31.233	TCP	54	80 → 53678 [FIN, ACK] Seq=1492 Ack=
13	5.145798	192.168.31.233	202.117.1.13	TCP	54	53678 → 80 [ACK] Seq=1318 Ack=1493
14	5.145852	192.168.31.233	202.117.1.13	TCP	54	53678 → 80 [FIN, ACK] Seq=1318 Ack=
15	5.149193	202.117.1.13	192.168.31.233	TCP	54	80 → 53678 [ACK] Seq=1493 Ack=1319
16	40.798846	192.168.31.233	202.117.1.13	TCP	66	53709 → 80 [SYN] Seq=0 Win=64240 Le
17	40.802513	202.117.1.13	192.168.31.233	TCP	66	80 → 53709 [SYN, ACK] Seq=0 Ack=1 W
18	40.802616	192.168.31.233	202.117.1.13	TCP	54	53709 → 80 [ACK] Seq=1 Ack=1 Win=26
19	40.802862	192.168.31.233	202.117.1.13	HTTP	697	GET /system/resource/code/datainput
20	40.806085	202.117.1.13	192.168.31.233	TCP	54	80 → 53709 [ACK] Seq=1 Ack=644 Win=
21	40.813300	202.117.1.13	192.168.31.233	HTTP	797	HTTP/1.1 200 OK
22	40.856648	192.168.31.233	202.117.1.13	TCP	54	53709 → 80 [ACK] Seq=644 Ack=744 Wi
23	45.813667	202.117.1.13	192.168.31.233	TCP	54	80 → 53709 [FIN, ACK] Seq=744 Ack=6
24	45.813780	192.168.31.233	202.117.1.13	TCP	54	53709 → 80 [ACK] Seq=644 Ack=745 Wi
25	45.813832	192.168.31.233	202.117.1.13	TCP	54	53709 → 80 [FIN, ACK] Seq=644 Ack=7
26	45.816632	202.117.1.13	192.168.31.233	TCP	54	80 → 53709 [ACK] Seq=745 Ack=645 Wi
27	48.986848	192.168.31.233	202.117.1.13	TCP	66	53719 → 80 [SYN] Seq=0 Win=64240 Le
28	48.990295	202.117.1.13	192.168.31.233	TCP	66	80 → 53719 [SYN, ACK] Seq=0 Ack=1 W

发现出现了 TCP 协议以及 HTTP 协议，下面对它们进行分析。

一、HTTP 协议使用 TCP 协议进行工作的方式

HTTP 协议是建立在 TCP 协议之上的应用层协议。当客户端需要与服务器进行 HTTP 通信时，它们之间会首先建立一个 TCP 连接，然后通过这个 TCP 连接进行 HTTP 通信。以下是 HTTP 协议如何使用 TCP 协议的基本流程：

- 建立 TCP 连接：这是通过 TCP 的三次握手完成的。详情可见下文的分析。
- 发送 HTTP 请求：客户端将 HTTP 请求封装在 TCP 数据包中，并通过已建立的 TCP 连接发送到服务器。
- 服务器处理请求：服务器收到客户端的 TCP 数据包后，解析其中的 HTTP 请求。服务器根据请求的内容执行相应的操作，可能是获取资源、处理数据等。此时二者通过 TCP 报文交流、传送数据。
- 发送 HTTP 响应：服务器通过 TCP 连接将 HTTP 响应封装在 TCP 数据包中发送给客户端。
- 关闭 TCP 连接。可见下文。

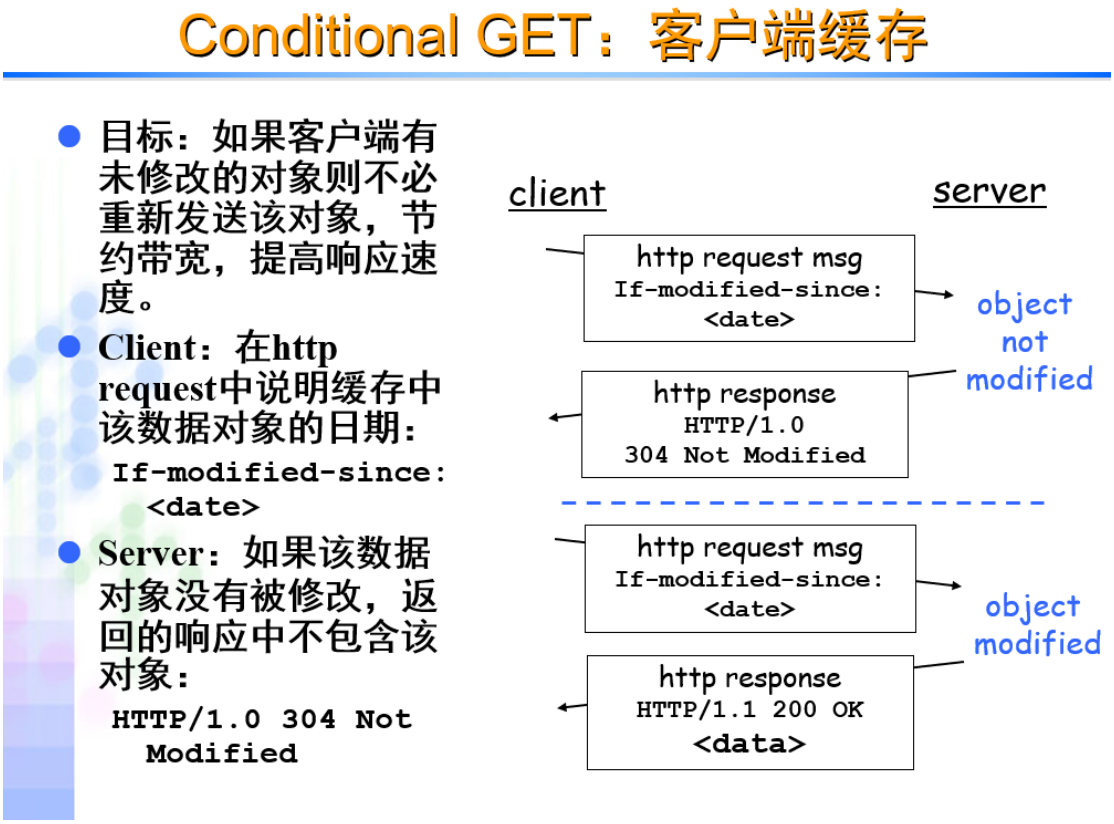
二、HTTP 请求报文（即 get 报文）和 HTTP 应答报文

报文分析

4	0.003782	192.168.31.233	202.117.1.13	HTTP	728 GET / HTTP/1.1
5	0.010126	202.117.1.13	192.168.31.233	TCP	54 80 → 53678 [ACK] Seq=1 Ack=675 Win=
6	0.010126	202.117.1.13	192.168.31.233	TCP	54 [TCP Dup ACK 5#1] 80 → 53678 [ACK]
7	0.010126	202.117.1.13	192.168.31.233	HTTP	802 HTTP/1.1 304 Not Modified

以上图为例。
出现了 HTTP 请求报文，其 info 显示 GET / HTTP/1.1。可以看出，其请求方法为 GET，请求的 URI 路径为 /（表示默认资源，通常是网站的首页），HTTP 版本为 HTTP/1.1。

最后出现 HTTP 响应包，状态码为 304，表示所请求的资源自上次请求以来未被修改。这与我之前打开过该网站有关。具体说明可以参见下图：



中间的第一个包是一个 ACK 包，其 Seq=1，Ack=675，这指明发送方期望收到下一个字节的序列号是 675。其 Len=0，表示这是一个空的确认包。
中间的第二个包一个重复确认的包，这可能是由于出现了网络故障。

HTTP1.1 报文格式

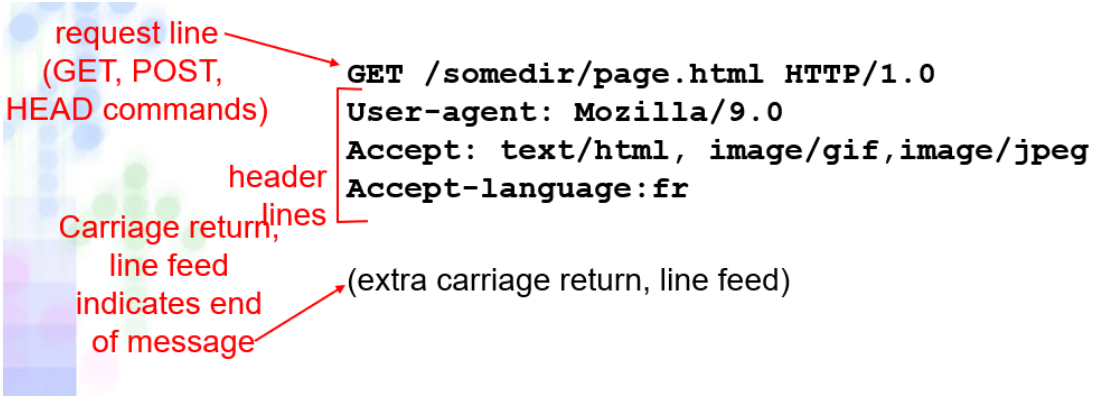
HTTP 请求报文格式包括请求行、请求头部和请求正文。其中请求正文是可选部分，在 GET 报文中看不到。

• 请求行由 3 部分组成，分别为：请求方法、请求目标以及协议版本，之间由空格分隔。

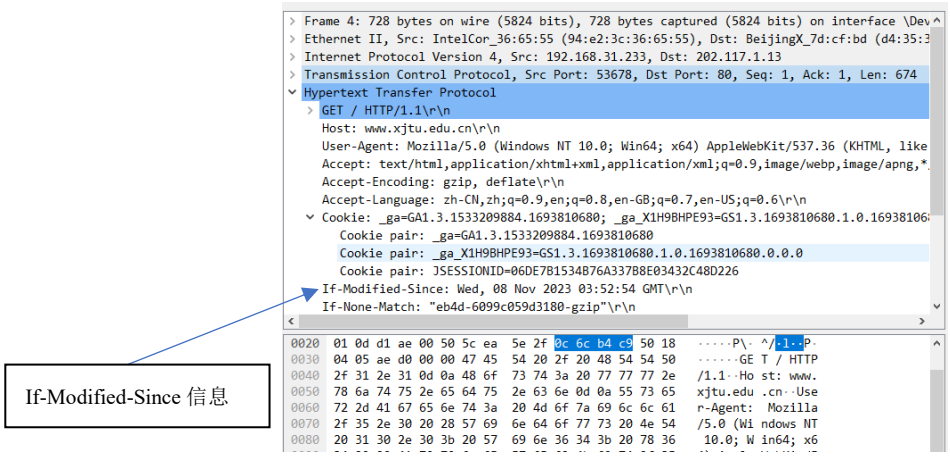
• 请求头部包含若干个属性，格式为“属性名:属性值”，服务端据此获取客户端的信息。

• 请求正文包含请求的数据。

示意图如下：



所抓的包中的信息如下图，与示意图中的信息是相似的。



特别在此例中，If-Modified-Since 中记录的日期（即缓存中主页的日期）的主页内容与服务器当时的主页内容是一致的（即数据对象未更改），因此响应状态码为 304。

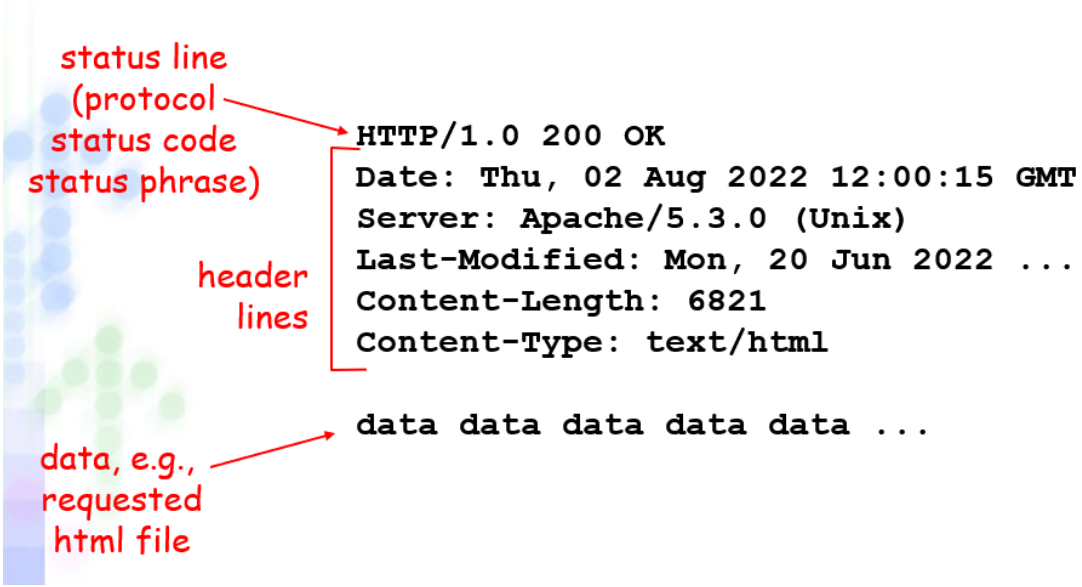
HTTP 响应报文格式包含

• 状态行（Status Line），包含 HTTP 版本、状态码和相应的状态信息;

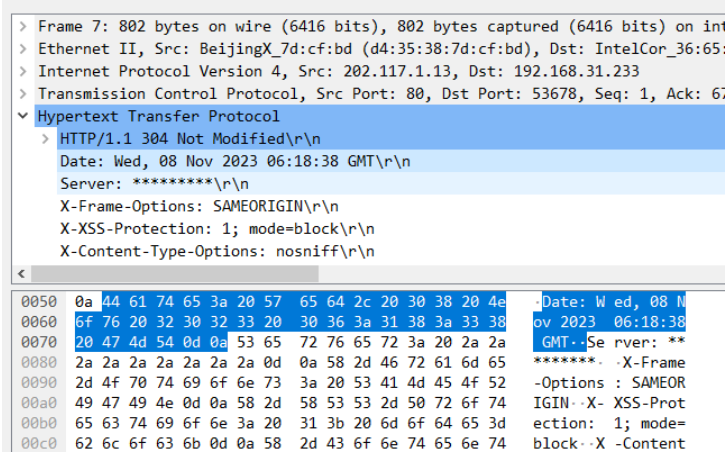
• 响应头部（Headers）： 包含关于响应的元信息，如服务器类型、日期、内容类型等;

• 消息体（Body）： 包含响应的数据，例如 HTML 页面内容。

示意图如下：



所抓的包中的信息如下图，与示意图中的信息是相似的



三、TCP 协议的三次握手(见 No.1~No.3）：

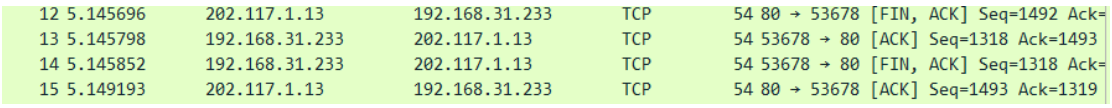
1	0.000000	192.168.31.233	202.117.1.13	TCP	66 53678 → 80 [SYN] Seq=0 Win=64240 Len=0
2	0.003448	202.117.1.13	192.168.31.233	TCP	66 80 → 53678 [SYN, ACK] Seq=0 Ack=1 Win=
3	0.003546	192.168.31.233	202.117.1.13	TCP	54 53678 → 80 [ACK] Seq=1 Ack=1 Win=26342

第一次握手：客户机向服务器发送 SYN 置 1 的报文，其$SEQ_client = 0$。作用是通知服务器客户端的存在，以便服务器知道客户端希望建立连接。

第二次握手：服务器向主机发送 SYN 和 ACK 都置 1 的报文，服务器$SEQ_{server} = 0$，$ACK = SEQ_{client} + 1 = 1$。第二次握手的作用是确认服务器已经接收到客户端的连接请求，并告知客户机服务器也愿意建立连接。

第三次握手：客户机发送一个 ACK 置 1 的报文，其$SEQ_{client} = SEQ_{client} + 1 = 1$，$ACK = SEQ_{server} + 1 = 1$。第三次握手的作用是确认双方都已知道连接已经建立，并且可以开始进行数据传输，这是连接建立的最后一步。

四、TCP 的释放：



12	5.145696	202.117.1.13	192.168.31.233	TCP	54 80 → 53678 [FIN, ACK] Seq=1492 Ack=
13	5.145798	192.168.31.233	202.117.1.13	TCP	54 53678 → 80 [ACK] Seq=1318 Ack=1493
14	5.145852	192.168.31.233	202.117.1.13	TCP	54 53678 → 80 [FIN, ACK] Seq=1318 Ack=
15	5.149193	202.117.1.13	192.168.31.233	TCP	54 80 → 53678 [ACK] Seq=1493 Ack=1319

第 12 条数据包 FIN, ACK 都置 1，其 Seq=1492、 Ack=1318、 Win=17408、Len=0，表示发起方数据发送结束。

第 13 条数据包 ACK 置 1，这是对第 12 条数据包的确认，表示接收到了发起方的关闭请求。其$Seq = 1318$，$Ack = 1492 + 1 = 1493$。

第 14 条数据包 FIN, ACK 都置 1，这是接收方发起的关闭请求，表示接收方也没有更多数据要发送了。其 Seq 和 ACK 与上一条相同，这可能是合并 ACK 的结果。

第 15 条数据包 ACK 置 1，表示发起方接收到了接收方的关闭请求。其 Seq=1493，Ack=1319。