深 圳 大 学 实 验 报 告

课 程 名　称：计算机网络

实验项目名称：数据包抓取与分析

学　　院：计算机与软件学院

专　　业：计算机科学与技术

指 导 教　师：邹永攀

报告人：刘睿辰　学号：**2018152051**　班级：数计班

实 验 时　间：**2021 年 3 月 23 日**

实验报告提交时间：**2021 年 4 月 6 日**

教务处制

一、实验目的：

学习安装、使用协议分析软件，掌握基本的数据报捕获、过滤和协议的分析技巧，能对抓取数据包进行分析。

二、实验环境：

1. 操作系统：Windows 10 操作系统；
2. 电脑具有 Internet 连接；
3. 电脑配置 Wireshark 抓包软件。

三、实验内容：

协议分析软件的安装和使用、学会抓取数据包的方法并对对抓取数据包进行分析。

四、实验步骤：

1. 网络协议分析软件 Wireshark 的使用。

1.1 网络嗅探器软件（以 Wireshark 为例）的功能

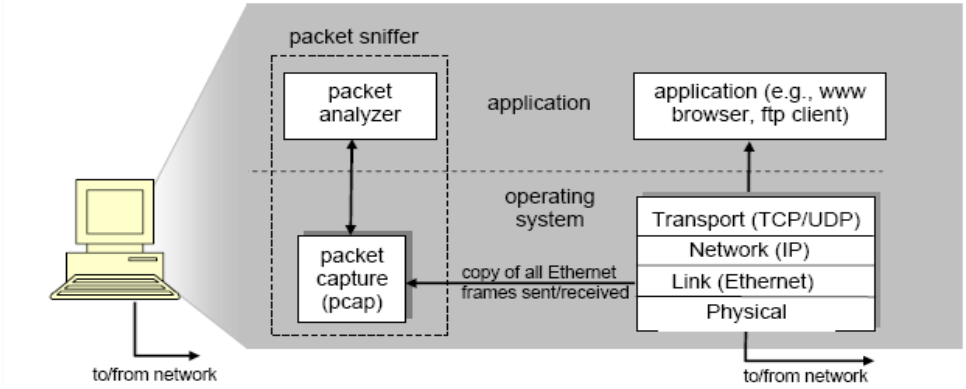


图 1. 网络嗅探器结构

如图 1 所示，网络嗅探器主要由包捕获器(packet capture)和包分析器(packet analyze)两个部分组成。包捕获器用于接收每一个网络上(发送接收/经过)的链路层帧，并复制至内部缓冲区。而包分析器用于显示协议消息中的所有字段的内容。为了能做到这一点，包分析器必须“懂得”协议所交换的所有信息的结构。例如，如果要显示 HTTP 协议所交换的信息中的每一个字段的内容，包分析器首先解析以太网帧，抽取出其中的 IP 数据报。再对 IP 数据报进行解析，抽取出其中的 TCP 段。下一步再对 TCP 段进行解析，抽取出其中的 HTTP 报文。最后，将 HTTP 报文中的各个字段分解，如报文开始的 GET、POST、或 HEAD 字符串等。

1.2 安装配置网络嗅探器，了解嗅探器的命令菜单的功能。

我们登录 Wireshark 官网 https://www.wireshark.org/download.html， 选择 windows installer(64-bit)选项进行下载，即可安装 Wireshark 软件。

打开 Wireshark 之后，可以看到有很多的本地接口，如图 2 所示。

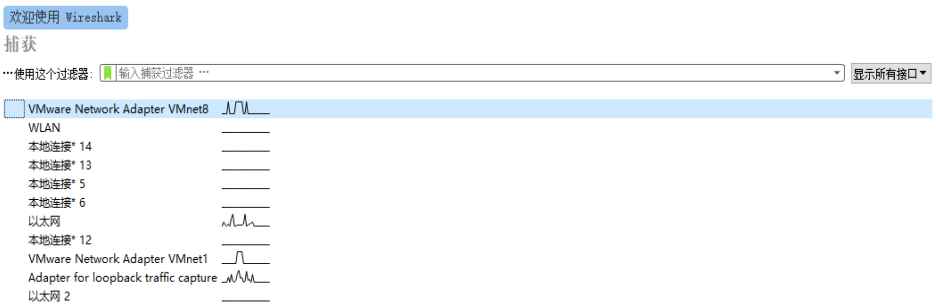


图 2. Wireshark 显示本地接口

然后我们需要选择一个接口进入。由于当前电脑连接的是网线，所以根据实验二的知识，我们需要进入以太网端口。双击“以太网”选项，进入抓包的界面，如图 3 所示。

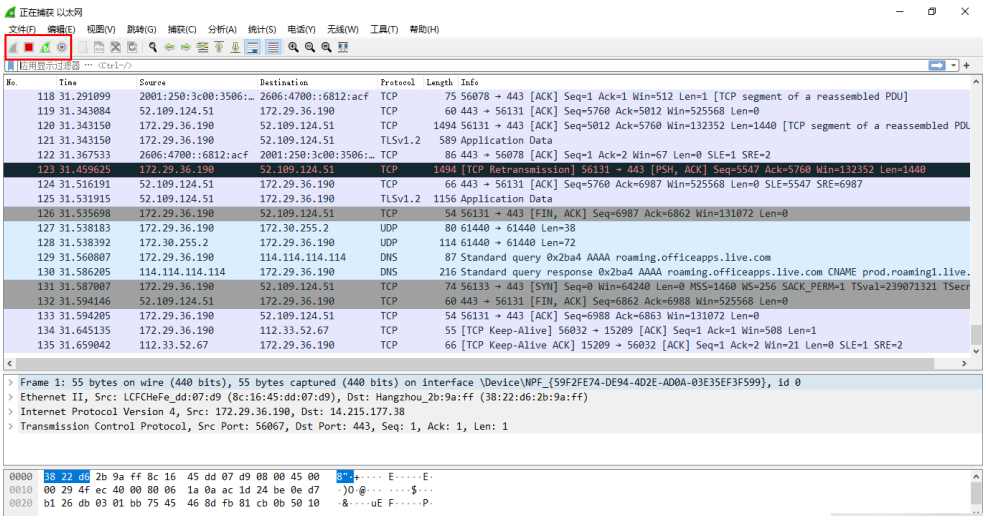


图 3. Wireshark 进入以太网接口

下面阐述嗅探器命令菜单的功能。

a） 首先我们看到有很多分栏，代表着数据包的编号、时间、Source（发送方 IP 地址）、Destination（接收方 IP 地址）、协议、长度以及信息；

b） 在图中红色标记区，我们看到工具栏分别为启动按钮、停止按钮、重新启动按钮、捕获选项按钮；

c） 捕获的数据包的文件操作、数据包的组操作、捕获的数据包的滚动并将不通的协议进行颜色标注，调整文字大小、分组自动选择合适的距离完整显示封包列表的各项内容；

d） Filter:显示过滤器，通过特定过滤规则显示要过滤的数据包。如图 4 所示，我们看到 Wireshark 的两种过滤规则，**抓包过滤器**和**显示过滤器**。

抓包过滤器：设置抓包条件，即只抓取那些感兴趣的包。此过滤器用于抓包过程中；

显示过滤器：用来设置显示条件，即只显示那些感兴趣的包。此过滤器用于显示过程中。如果抓包时未设置抓包过滤器，将所有包都抓了回来，则显示时通常需要设置显示过滤器。否则抓回来的各种包混杂在一起，将会给协议包分析

造成很大的困难。

图 4(a) 显示过滤器

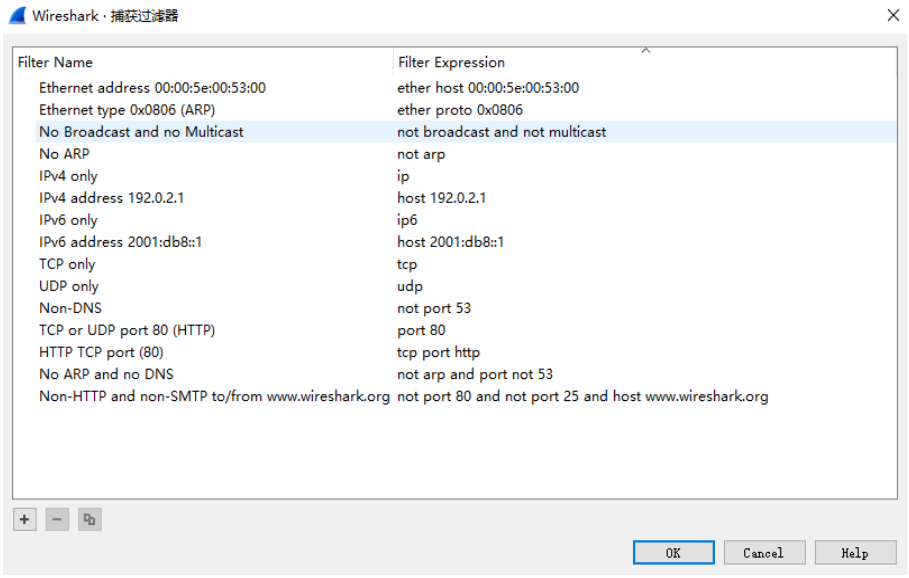


图 4(b) 捕获过滤器

图 4. Wireshark 的两种过滤规则

1.3 设置抓包过滤器和显示过滤器，抓取一个 HTTP 会话过程所传输的包。观察包结构和格式，分析各字段的语义，以图形化方式显示该 HTTP 会话过程。

我们访问 http://test.nago.club/这个网址，并且我们需要结合过滤器的语法规则，在显示过滤器中输入 http 表示过滤掉非 http 协议的数据包。接下来我们需要知道我们访问的主机名，我们通过实验二的经验，采用 ping 指令，如图 5 所示，主机名为 121.36.34.25.

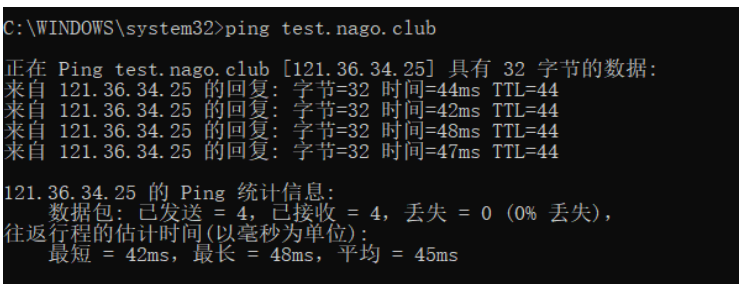


图 5. 测试主机名

在 IP 层要捕获所有 IP 地址为 121.36.34.25 的数据包，则可设置过滤规则为：ip.dst_host==121.36.34.25，注意这个 IP 可以作为接收方，也可以作为发送方，所以我们的过滤规则为

$$http \ \&\& \ ip.dst\_host == 121.36.34.25 \ || \ ip.src\_host == 121.36.34.25$$

如图 6 所示是我们过滤得到的结果。

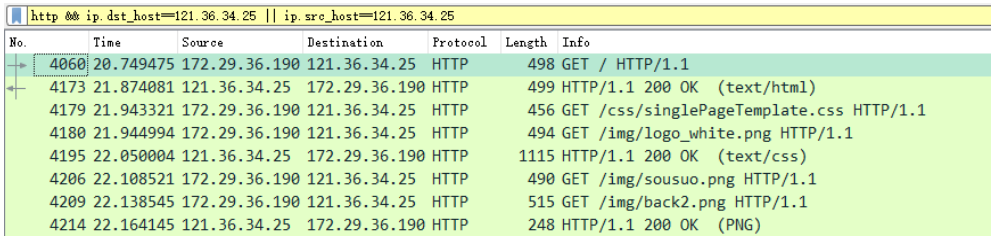


图 6. 抓取一个 HTTP 会话过程所传输的包

接下来我们查看一下封包的信息。我们点击其中一个数据包，如图 7 所示。

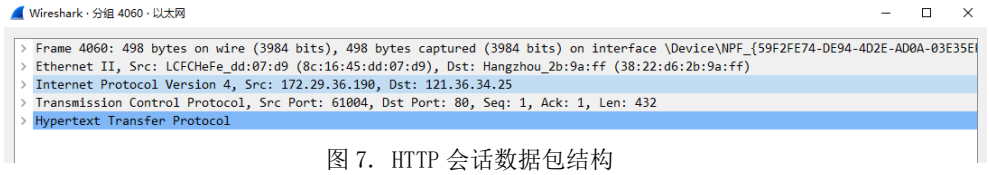


图 7. HTTP 会话数据包结构

我们看到封包有 Frame（物理层的数据帧概况），Ethernet II（数据链路层以太网帧头部信息），Internet Protocol Version 4（互联网层 IP 包头部信息），Transmission Control Protocol（传输层 T 的数据段头部信息），Hypertext Transfer Protocol（应用层的信息）。对应网络层次结构，我们看到各部分对应关系如图 8 所示。

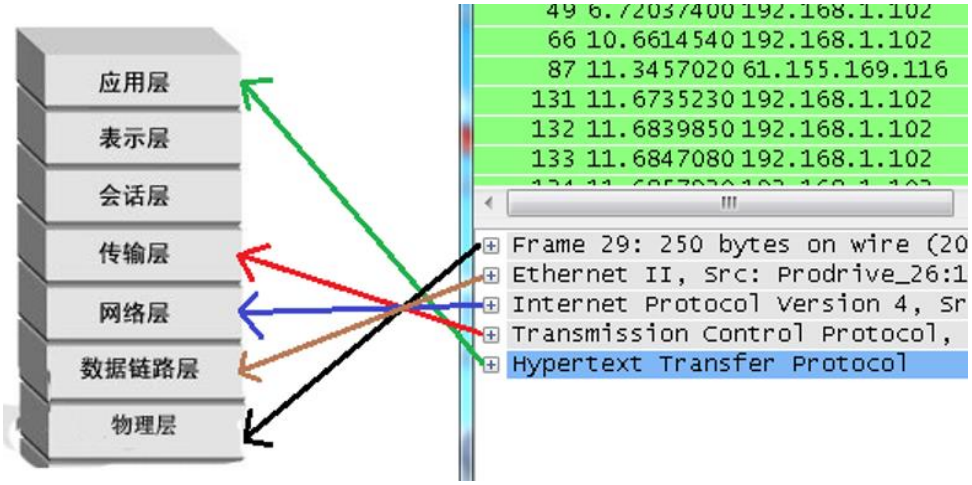


图 8. HTTP 会话数据包结构与网络层次之间的关系

然后我们以图形化方式显示该 HTTP 会话过程。

点击统计 ⟶ 流量图，我们可以看到如图 9 所示的图形。该图反映了请求-接收过程。我们在图中发现左边是我们的 IP 地址，右边是访问的网址的 IP 地址。首先我们向要访问的网址发送请求，然后接收允许访问的反馈。然后我们继续发送请求，然后我们的访问目标继续给我们反馈。这就体现了一个 Client-Server(C/S)架构。所以我们从图中可以看出 http 协议实际上是基于客户端/服务端（C/S）的架构模型，通过一个可靠的链接来交换信息。

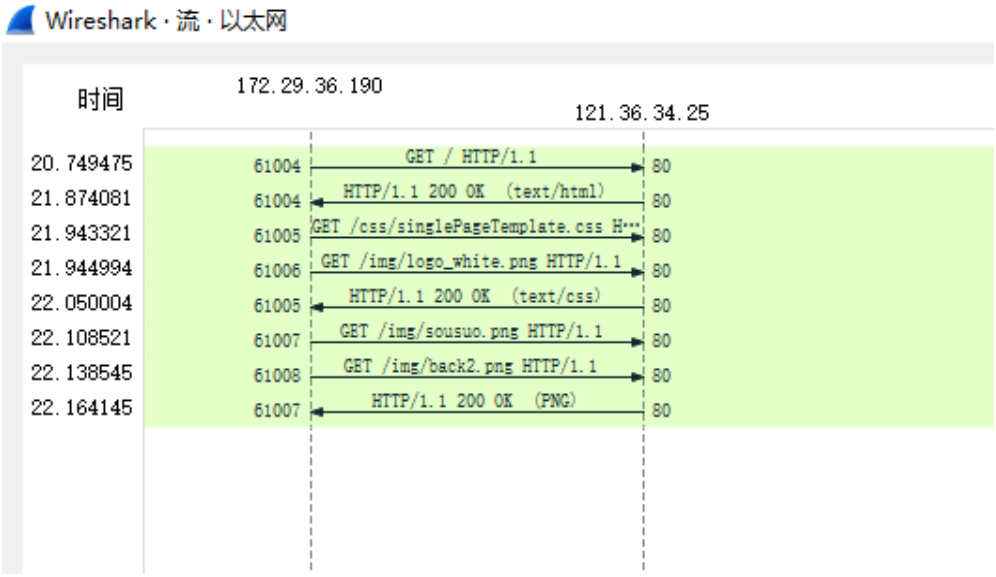


图 9. HTTP 会话流量图

1.4 利用过滤器抓取/显示一个完整的 POP3 会话(或显示样本文件中的一个完整的 POP3 会话)，将显示窗口进行截屏并粘贴到实验报告中，在实验报告中详细说明该会话中双方交换的各报文的含义。

解决这个问题之前，我们来看一下邮件报文交付的三个阶段，如图 10 所示。

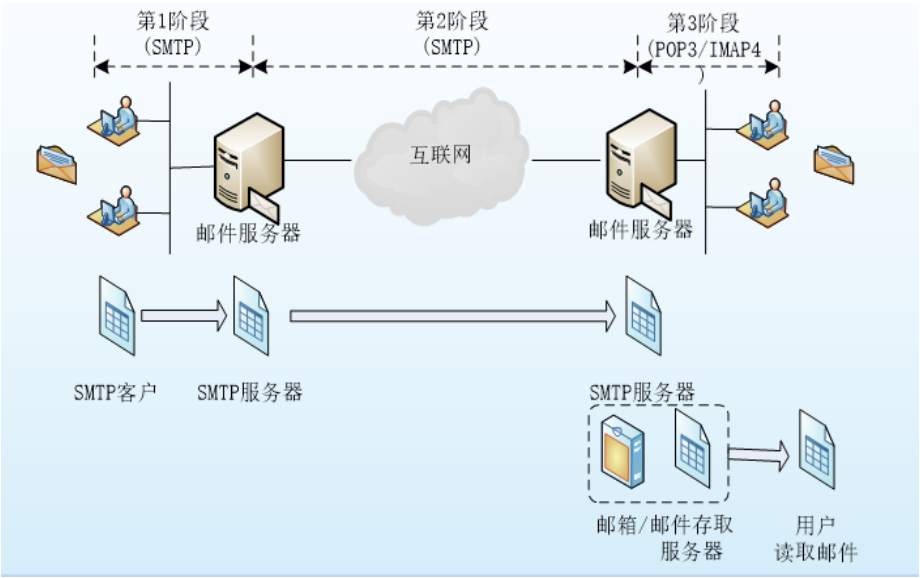


图 10. 邮件报文交付的三个阶段

我们从图中可以看出，POP3 协议实际上是用来接收电子邮件的协议。我们知道网易邮箱实际上是可以开启 POP3 服务的，所以我们给网易邮箱 harry000403@163.com 来进行 POP 账户设置，如图 11 所示。

邮箱设置好之后，我们打开 outlook，以网易邮箱来登录，此时我们需要在 Wireshark 中设置过滤条件为 pop，也就是过滤掉非 pop 协议的数据包。

我们可以看到结果如图 12 所示。

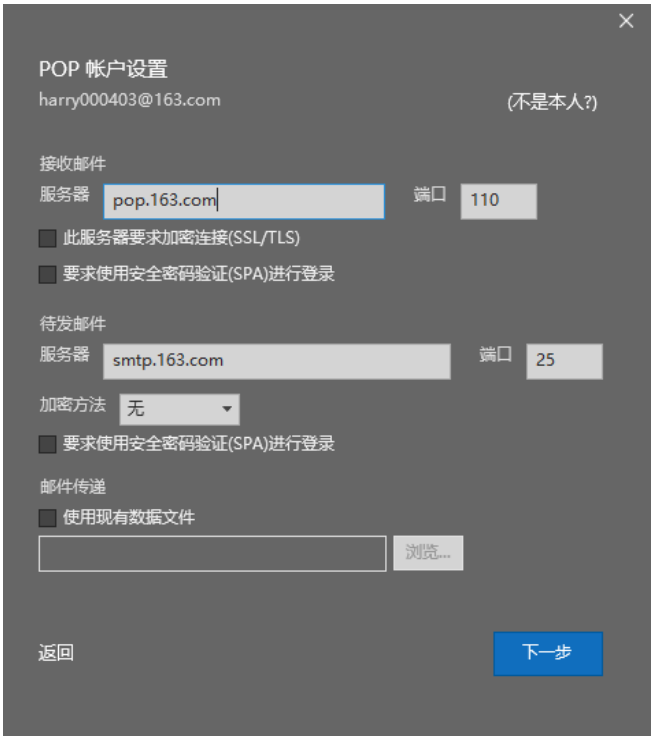
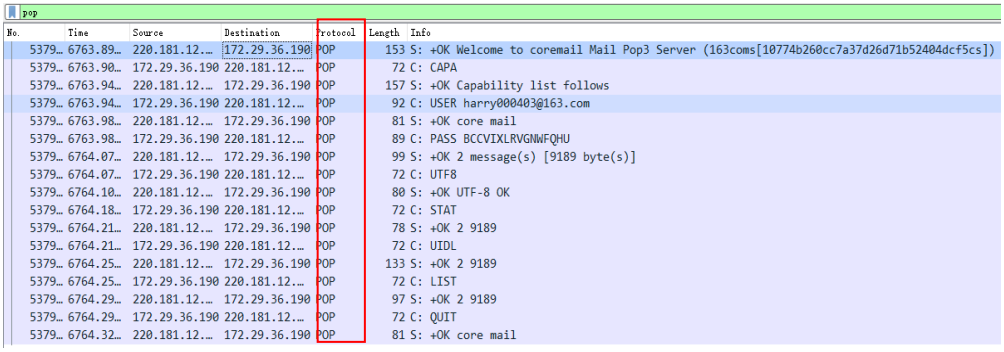


图 11. 网易邮箱进行 POP 账户设置



pop

No.	Time	Source	Destination	Protocol	Length	Info
5379…	6763.89…	220.181.12.…	172.29.36.190	POP	153	S: +OK Welcome to coremail Mail Pop3 Server (163coms[10774b260cc7a37d26d71b52404dcf5cs])
5379…	6763.90…	172.29.36.190	220.181.12.…	POP	72	C: CAPA
5379…	6763.94…	220.181.12.…	172.29.36.190	POP	157	S: +OK Capability list follows
5379…	6763.94…	172.29.36.190	220.181.12.…	POP	92	C: USER harry000403@163.com
5379…	6763.98…	220.181.12.…	172.29.36.190	POP	81	S: +OK core mail
5379…	6763.98…	172.29.36.190	220.181.12.…	POP	89	C: PASS BCCVIXLRVGNWFQHU
5379…	6764.07…	220.181.12.…	172.29.36.190	POP	99	S: +OK 2 message(s) [9189 byte(s)]
5379…	6764.07…	172.29.36.190	220.181.12.…	POP	72	C: UTF8
5379…	6764.10…	220.181.12.…	172.29.36.190	POP	80	S: +OK UTF-8 OK
5379…	6764.18…	172.29.36.190	220.181.12.…	POP	72	C: STAT
5379…	6764.21…	220.181.12.…	172.29.36.190	POP	78	S: +OK 2 9189
5379…	6764.21…	172.29.36.190	220.181.12.…	POP	72	C: UIDL
5379…	6764.25…	220.181.12.…	172.29.36.190	POP	133	S: +OK 2 9189
5379…	6764.25…	172.29.36.190	220.181.12.…	POP	72	C: LIST
5379…	6764.29…	220.181.12.…	172.29.36.190	POP	97	S: +OK 2 9189
5379…	6764.29…	172.29.36.190	220.181.12.…	POP	72	C: QUIT
5379…	6764.32…	220.181.12.…	172.29.36.190	POP	81	S: +OK core mail

图 12. 接收 pop 协议数据包

下面我们观察一下邮件传输的全过程。我们用网易邮箱向另一个邮箱发送信息，然后观察 Wireshark 里面的变化，如图 13 所示。

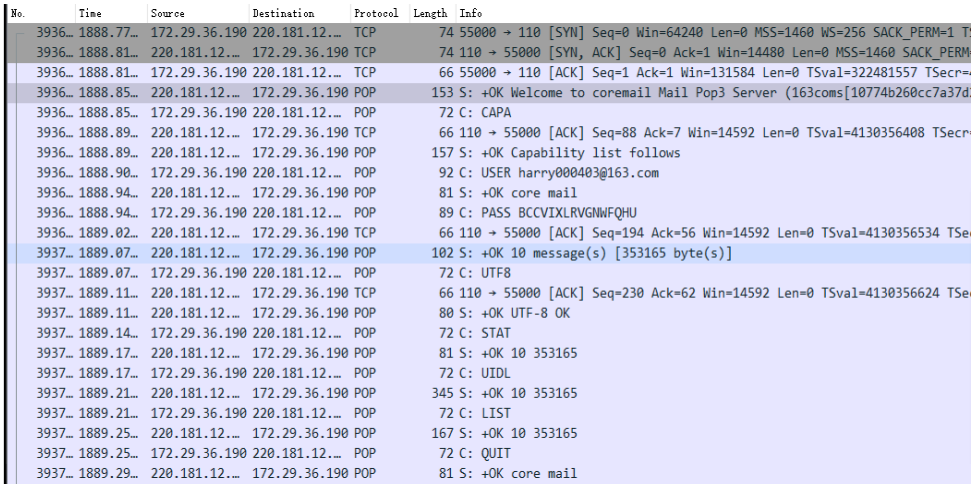
```
142 49.195636 172.29.36.190 220.181.12.11 SMTP      100 C: MAIL FROM: <harry000403@163.com>
144 49.236235 220.181.12.11 172.29.36.190 SMTP       79 S: 250 Mail OK
145 49.237448 172.29.36.190 220.181.12.11 SMTP       96 C: RCPT TO: <1820426105@qq.com>
146 49.276005 220.181.12.11 172.29.36.190 SMTP       79 S: 250 Mail OK
147 49.277778 172.29.36.190 220.181.12.11 SMTP       98 C: RCPT TO: <harry000403@163.com>
148 49.318797 220.181.12.11 172.29.36.190 SMTP       79 S: 250 Mail OK
149 49.319926 172.29.36.190 220.181.12.11 SMTP       72 C: DATA
151 49.358323 220.181.12.11 172.29.36.190 SMTP      103 S: 354 End data with <CR><LF>.<CR><LF>
152 49.379513 172.29.36.190 220.181.12.11 SMTP     1090 C: DATA fragment, 1024 bytes
153 49.379601 172.29.36.190 220.181.12.11 SMTP     1514 C: DATA fragment, 1448 bytes
154 49.379601 172.29.36.190 220.181.12.11 SMTP       67 C: DATA fragment, 1 byte
157 49.426879 220.181.12.11 172.29.36.190 SMTP      139 S: 250 Mail OK queued as smtp7,C8CowADHH1MZeGVgKB6UVA--.27028S2 1617262617
168 51.931316 172.29.36.190 220.181.12.11 SMTP       72 C: DATA fragment, 6 bytes
170 51.971563 220.181.12.11 172.29.36.190 SMTP       75 S: 221 Bye
```

图 13. 发送邮件以及邮件传输过程

我们查阅 SMTP 的响应码表，250 码代表要求的邮件操作完成。SMTP 的关键词还有 RCPT TO（预期的收信人）、MAIL FROM（发信人）、DATA（邮件主体）。由这个过程我们看到网易邮箱 harry000403@163.com 给 QQ 邮箱 1820426105@qq.com 发送了信息的过程。最后的响应码 221 代表服务关闭传输通道，代表传输结束。

以上部分是邮件发送和传输的过程。下面我们展示接收邮件的过程。

打开 Outlook 后登录我们的网易邮箱。然后在 Wireshark 中设置过滤规则为 POP，可以看到如图 14 所示的界面。



No.	Time	Source	Destination	Protocol	Length	Info
3936…	1888.77…	172.29.36.190	220.181.12.…	TCP	74	55000 → 110 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1 T
3936…	1888.81…	220.181.12.…	172.29.36.190	TCP	74	110 → 55000 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM
3936…	1888.81…	172.29.36.190	220.181.12.…	TCP	66	55000 → 110 [ACK] Seq=1 Ack=1 Win=131584 Len=0 TSval=322481557 TSecr=
3936…	1888.85…	220.181.12.…	172.29.36.190	POP	153	S: +OK Welcome to coremail Mail Pop3 Server (163coms[10774b260cc7a37d
3936…	1888.85…	172.29.36.190	220.181.12.…	POP	72	C: CAPA
3936…	1888.89…	220.181.12.…	172.29.36.190	TCP	66	110 → 55000 [ACK] Seq=88 Ack=7 Win=14592 Len=0 TSval=4130356408 TSecr
3936…	1888.89…	220.181.12.…	172.29.36.190	POP	157	S: +OK Capability list follows
3936…	1888.90…	172.29.36.190	220.181.12.…	POP	92	C: USER harry000403@163.com
3936…	1888.94…	220.181.12.…	172.29.36.190	POP	81	S: +OK core mail
3936…	1888.94…	172.29.36.190	220.181.12.…	POP	89	C: PASS BCCVIXLRVGNWFQHU
3936…	1889.02…	220.181.12.…	172.29.36.190	TCP	66	110 → 55000 [ACK] Seq=194 Ack=56 Win=14592 Len=0 TSval=4130356534 TSe
3937…	1889.07…	220.181.12.…	172.29.36.190	POP	102	S: +OK 10 message(s) [353165 byte(s)]
3937…	1889.07…	172.29.36.190	220.181.12.…	POP	72	C: UTF8
3937…	1889.11…	220.181.12.…	172.29.36.190	TCP	66	110 → 55000 [ACK] Seq=230 Ack=62 Win=14592 Len=0 TSval=4130356624 TSe
3937…	1889.11…	220.181.12.…	172.29.36.190	POP	80	S: +OK UTF-8 OK
3937…	1889.14…	172.29.36.190	220.181.12.…	POP	72	C: STAT
3937…	1889.17…	220.181.12.…	172.29.36.190	POP	81	S: +OK 10 353165
3937…	1889.17…	172.29.36.190	220.181.12.…	POP	72	C: UIDL
3937…	1889.21…	220.181.12.…	172.29.36.190	POP	345	S: +OK 10 353165
3937…	1889.21…	172.29.36.190	220.181.12.…	POP	72	C: LIST
3937…	1889.25…	220.181.12.…	172.29.36.190	POP	167	S: +OK 10 353165
3937…	1889.25…	172.29.36.190	220.181.12.…	POP	72	C: QUIT
3937…	1889.29…	220.181.12.…	172.29.36.190	POP	81	S: +OK core mail

图 14. 登录 outlook 过程

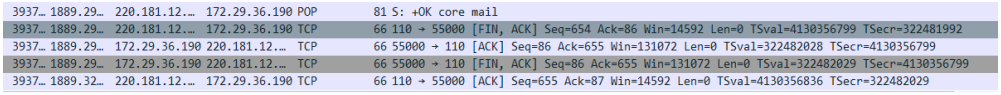
我们可以看到，可以看到，先进行了 TCP 三次握手建立起连接。之后本地分别输入命令 USER+邮箱，服务器先返回一个 ACK 确认包，然后再返回一个 POP3 报文，本地收到后也返回一个 ACK 确认包给服务器。

接着本地输入命令 PASS+密码，服务器认证成功后，开始进入处理阶段。

STAT 命令用于查询邮箱中的统计信息，当本地输入命令 STAT 时，服务器给出 ACK 确认包和 POP3 报文，本地再用 ACK 确认包响应。

LIST 命令用于列出邮箱中的邮件信息，当本地输入命令 LIST 时，服务器直接用 POP3 报文进行响应，少了 ACK 确认包，本地收到 POP3 报文后再发送 ACK 确认包给服务器。当用 LIST 命令后带数字参数时，是用于列出邮箱中指定邮件序号的邮件信息。

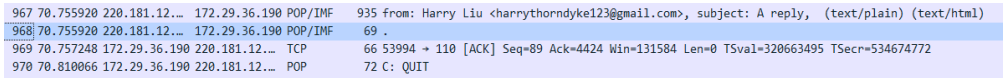
最后，QUIT 命令表示要结束邮件接收过程。本地发送 QUIT 命令后，服务器先发送 ACK 确认包，然后发送 POP3 报文。最后本地发起 TCP 断开连接，经过四次挥手后结束整个过程，如图 15 所示。



3937…	1889.29…	220.181.12.…	172.29.36.190	POP	81	S: +OK core mail
3937…	1889.29…	220.181.12.…	172.29.36.190	TCP	66	110 → 55000 [FIN, ACK] Seq=654 Ack=86 Win=14592 Len=0 TSval=4130356799 TSecr=322481992
3937…	1889.29…	172.29.36.190	220.181.12.…	TCP	66	55000 → 110 [ACK] Seq=86 Ack=655 Win=131072 Len=0 TSval=322482028 TSecr=4130356799
3937…	1889.29…	172.29.36.190	220.181.12.…	TCP	66	55000 → 110 [FIN, ACK] Seq=86 Ack=655 Win=131072 Len=0 TSval=322482029 TSecr=4130356799
3937…	1889.32…	220.181.12.…	172.29.36.190	TCP	66	110 → 55000 [ACK] Seq=655 Ack=87 Win=14592 Len=0 TSval=4130356836 TSecr=322482029

图 15. 四次挥手结束过程

以图 16 为例，我们接收到了一封邮件，来自 harrythorndyke123@gmail.com.



967	70.755920	220.181.12.…	172.29.36.190	POP/IMF	935	from: Harry Liu <harrythorndyke123@gmail.com>, subject: A reply, (text/plain) (text/html)
968	70.755920	220.181.12.…	172.29.36.190	POP/IMF	69	.
969	70.757248	172.29.36.190	220.181.12.…	TCP	66	53994 → 110 [ACK] Seq=89 Ack=4424 Win=131584 Len=0 TSval=320663495 TSecr=534674772
970	70.810066	172.29.36.190	220.181.12.…	POP	72	C: QUIT

图 16. 四次挥手结束过程

1.5 总结网络嗅探器都能做些什么

网络嗅探器不仅可以用过滤器来筛选我们想要分析的数据包，还可以查看报文格式以及将绘画过程进行可视化，方便用户进行理解。

2. HTTP 协议的认知与分析

2.1 抓取一个 HTTP 会话所传输的报文，并分析报文的结构、格式及其内容.

1） 开启 Wireshark 抓包，在过滤器中输入 http，即过滤 http 协议的分组。

2） 打开浏览器，输入一个网址，这里以 test.nago.club 为例。（注意：为了避免浏览器缓存起 作用，最好使用 chrome 浏览器的 incognito 隐身模式）。

3） 观察到 Wireshark 分组列表栏中出现了 HTTP 协议分组，如图 17 所示。

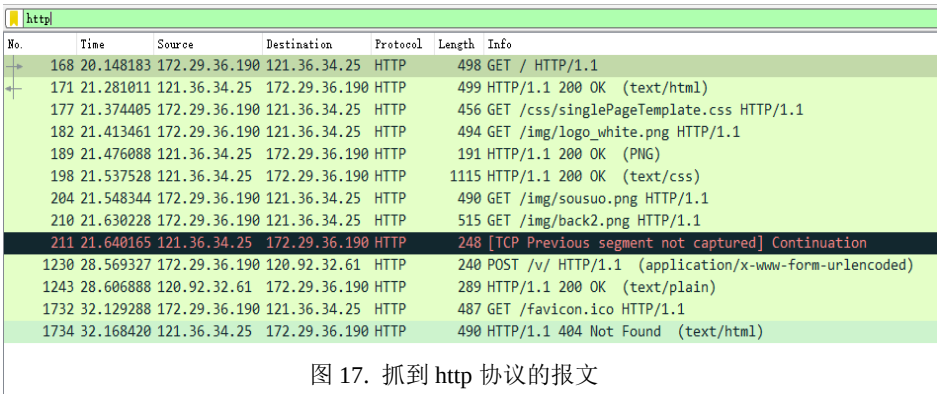


图 17. 抓到 http 协议的报文

4） 在前面抓 http 协议的包的时候我们通过 ping 指令已经得知该网站的 IP 地址为 121.36.34.25，再用上述规则

$$http \ \&\& \ ip.dst\_host == 121.36.34.25 \ || \ ip.src\_host == 121.36.34.25$$

进行过滤，可以得到新的结果如图 18 所示。

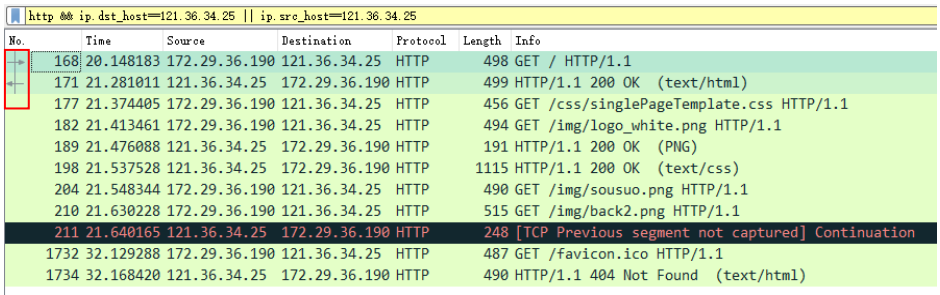


图 18. 过滤 http 协议报文

5） 从图 18 中可以看出两个 IP 地址，分别是我们的访问地址 121.36.34.25，另一个就是我们的 IP 地址。在途中我们还发现了标记位置的箭头是往复的，这个代表我们用 GET 方法进行 HTTP 请求，然后收到答复为 OK，代表可以进行访问。

6） 下面我们分析箭头右指“ ⟶ ”的 HTTP 请求。我们点击其中一个报文，然后观察结果如图 19 所示。

图 19. http 结构

HTTP 请求包括三部分，分别是请求行（请求方法）、请求头（消息报头）和请求正文。

请求行：对应于图 19 中的第一行，由三部分组成，第一部分说明了该请求是 GET 请求，第二部分是一个斜杠（/css/singlePageTemplate.css），用来说明请求

是该域名根目录下的/css/singlePageTemplate.css，第三部分说明使用的是HTTP1.1 版本。

请求头：对应于图中第二行至第四行，也被称为消息头。其中，HOST 代表请求主机地址，这里就是 test.nago.club，User-Agent 代表浏览器的标识，请求头由客户端自行设定。

请求正文：请求正文是可选的，它最常出现在 POST 请求方式中。

7）上述 HTTP 请求方法为 GET。事实上，HTTP 请求方法有很多，HTTP1.0 定义了三种请求方法： GET, POST 和 HEAD 方法；HTTP1.1 新增了五种请求方法：OPTIONS、PUT、PATCH、DELETE、TRACE 和 CONNECT 方法，具体应用如表 1 所示。

序号	方法	描述
1	GET	请求指定的页面信息，并返回实体主体。
2	HEAD	类似于 GET 请求，只不过返回的响应中没有具体的内容，用于获取报头
3	POST	向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST 请求可能会导致新的资源的建立和/或已有资源的修改。
4	PUT	从客户端向服务器传送的数据取代指定的文档的内容。
5	DELETE	请求服务器删除指定的页面。
6	CONNECT	HTTP/1.1 协议中预留给能够将连接改为管道方式的代理服务器。
7	OPTIONS	允许客户端查看服务器的性能。
8	TRACE	回显服务器收到的请求，主要用于测试或诊断。
9	PATCH	是对 PUT 方法的补充，用来对已知资源进行局部更新 。

表 1. HTTP 请求方法

8）下面我们分析箭头左指“ ← ”的 HTTP 响应。打开其中一个响应报文，如图 20 所示。

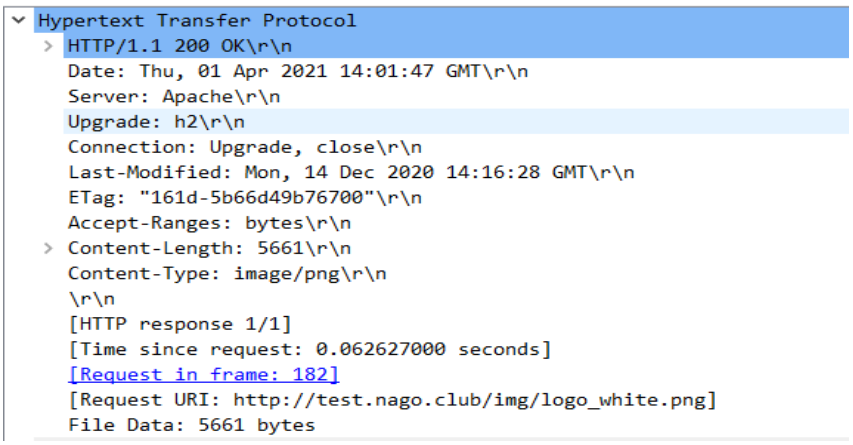


图 20. HTTP 响应报文

响应行：对应图 20 中第一行，其中有 HTTP 版本（HTTP/1.1）、状态码（200）以及消息“OK”。常见的状态码如表 2 所示。

序号	状态码	描述
1	200	客户端请求成功，是最常见的状态。
2	302	重定向。
3	404	请求资源不存在，是最常见的状态。
4	400	客户端请求有语法错误，不能被服务器所理解。
5	401	请求未经授权。
6	403	服务器收到请求，但是拒绝提供服务。
7	500	服务器内部错误，是最常见的状态。
8	503	服务器当前不能处理客户端的请求。

表 2.　HTTP 应答状态码

响应头：第二行以后。如，Date 代表系统的 GMT 时间，Server 代表服务器名字，Content-Length 代表表示内容长度等等。

响应正文：是服务器向客户端发送的 HTML 数据。

2.2　观察一个完整的 HTTP 会话，并分析会话各个阶段的请求/响应操作.

首先我们先查看网页前端的构成，如图 21 所示。

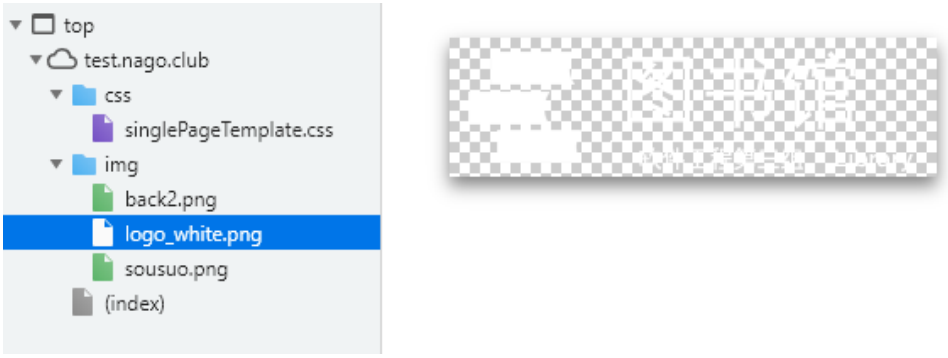


图 21.　网页前端构成

然后我们访问该网页，并设置过滤条件，得到如图 22 所示的结果。

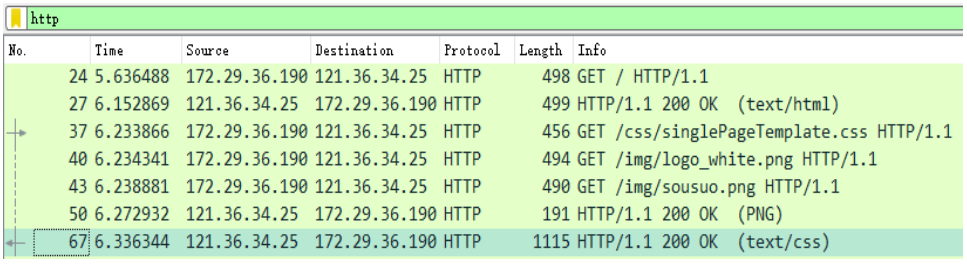


图 22.　http 访问过程

我们右键点击第一行数据包，得到 TCP 三次握手过程，如图 23 所示；

然后再点击 HTTP 响应，发现返回了 html；

观察返回结果，我们得到了应答包括了我们请求的资源，这些资源和图 21 中的资源都是对应的；

关闭请求方，四次挥手。

789	6.778325	121.36.34.25	172.29.36.190	HTTP
790	6.778325	121.36.34.25	172.29.36.190	TCP
791	6.779181	172.29.36.190	121.36.34.25	TCP
792	6.779978	172.29.36.190	121.36.34.25	TCP
793	6.816433	121.36.34.25	172.29.36.190	TCP

图 23. 四次挥手

```
Connection: Upgrade, close\r\n
Vary: Accept-Encoding\r\n
Content-Encoding: gzip\r\n
Content-Length: 1491\r\n
Content-Type: text/html; charset=UTF-8\r\n
\r\n
```

图 24. 返回 html

2.3 重建上述 HTTP 会话，并将截图粘贴到实验报告中，说明会话过程中各请求/响应报文的含义。

首先，发送访问请求，如图 25 所示，服务器响应成功，并返回 html。

No.	Time	Source	Destination	Protocol	Length	Info
24	5.636488	172.29.36.190	121.36.34.25	HTTP	498	GET / HTTP/1.1
27	6.152869	121.36.34.25	172.29.36.190	HTTP	499	HTTP/1.1 200 OK (text/html)
37	6.233866	172.29.36.190	121.36.34.25	HTTP	456	GET /css/singlePageTemplate.css HTTP/1.1
40	6.234341	172.29.36.190	121.36.34.25	HTTP	494	GET /img/logo_white.png HTTP/1.1
43	6.238881	172.29.36.190	121.36.34.25	HTTP	490	GET /img/sousuo.png HTTP/1.1
50	6.272932	121.36.34.25	172.29.36.190	HTTP	191	HTTP/1.1 200 OK (PNG)
67	6.336344	121.36.34.25	172.29.36.190	HTTP	1115	HTTP/1.1 200 OK (text/css)
75	6.389930	172.29.36.190	121.36.34.25	HTTP	515	GET /img/back2.png HTTP/1.1

图 25. 第一个分组

其次，我们请求 css 文件，如图 26 所示，请求成功，返回 css 文件。

No.	Time	Source	Destination	Protocol	Length	Info
24	5.636488	172.29.36.190	121.36.34.25	HTTP	498	GET / HTTP/1.1
27	6.152869	121.36.34.25	172.29.36.190	HTTP	499	HTTP/1.1 200 OK (text/html)
37	6.233866	172.29.36.190	121.36.34.25	HTTP	456	GET /css/singlePageTemplate.css HTTP/1.1
40	6.234341	172.29.36.190	121.36.34.25	HTTP	494	GET /img/logo_white.png HTTP/1.1
43	6.238881	172.29.36.190	121.36.34.25	HTTP	490	GET /img/sousuo.png HTTP/1.1
50	6.272932	121.36.34.25	172.29.36.190	HTTP	191	HTTP/1.1 200 OK (PNG)
67	6.336344	121.36.34.25	172.29.36.190	HTTP	1115	HTTP/1.1 200 OK (text/css)
75	6.389930	172.29.36.190	121.36.34.25	HTTP	515	GET /img/back2.png HTTP/1.1

图 26. 第二个分组，请求 css

第三个分组时请求图片，如图 27 所示，请求成功，返回 png 文件。

其余两个都是请求图片，这里不做赘述。

这就是访问这个网页整个的 http 会话过程。会话过程中的请求、响应报文的意义都在上进行了说明。

我们以流量图来展示整个绘画过程，如图 28 所示。

No.	Time	Source	Destination	Protocol	Length	Info
55	39.020562	172.29.36.190	121.36.34.25	HTTP	498	GET / HTTP/1.1
58	39.414655	121.36.34.25	172.29.36.190	HTTP	498	HTTP/1.1 200 OK (text/html)
68	39.597351	172.29.36.190	121.36.34.25	HTTP	456	GET /css/singlePageTemplate.css HTTP/1.1
71	39.601947	172.29.36.190	121.36.34.25	HTTP	494	GET /img/logo_white.png HTTP/1.1
74	39.611578	172.29.36.190	121.36.34.25	HTTP	490	GET /img/sousuo.png HTTP/1.1
82	39.642246	121.36.34.25	172.29.36.190	HTTP	191	HTTP/1.1 200 OK (PNG)
87	39.644918	121.36.34.25	172.29.36.190	HTTP	1115	HTTP/1.1 200 OK (text/css)
95	39.650685	121.36.34.25	172.29.36.190	HTTP	248	HTTP/1.1 200 OK (PNG)
105	39.747856	172.29.36.190	121.36.34.25	HTTP	515	GET /img/back2.png HTTP/1.1

图 27. 第三次分组，请求图片

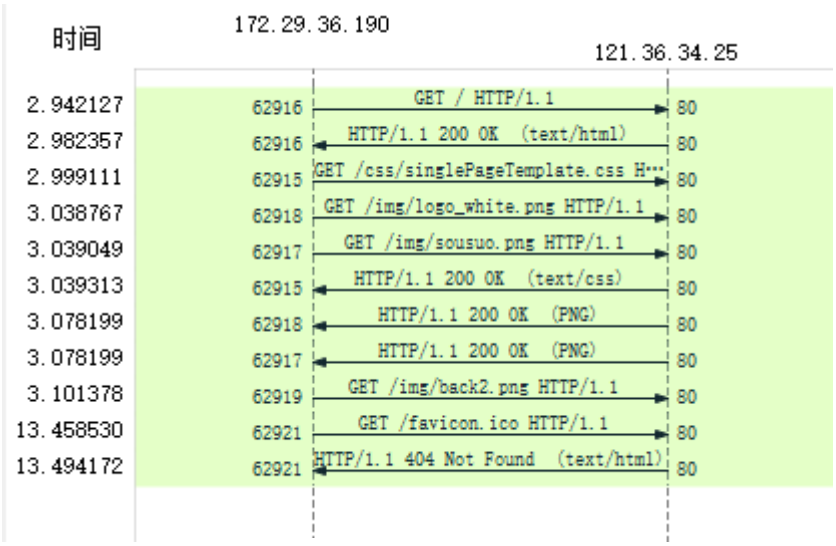


图 28. HTTP 会话过程流量图

2.4 根据上述实验内容回答以下问题

2.4.1 你的浏览器运行的 HTTP 版本是 1.0 还是 1.1？服务器端的 HTTP 版本是什么？

答：根据我们之前的解图，我们浏览器是 HTTP1.1 版本。如图 29 所示，服务器端的 HTTP 版本是 HTTP1.1。

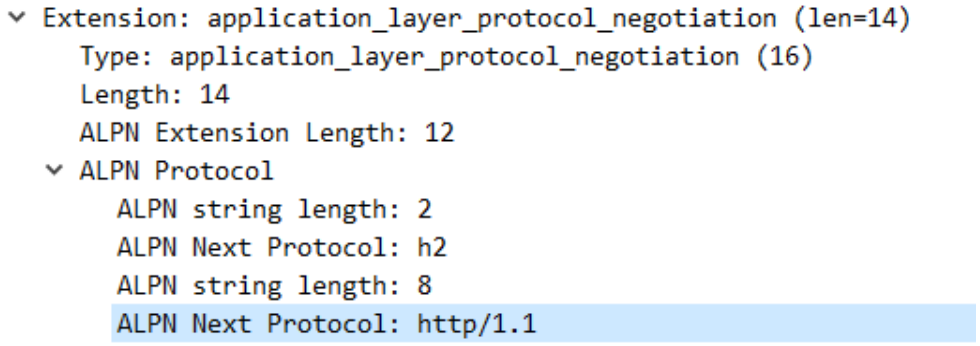


图 29. 服务器端的 HTTP 版本

2.4.2 你的浏览器指出它所能够接受的语言是什么？

答：如图 30 所示，网页语言为简体中文。

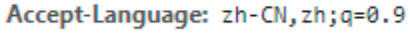


图 30. 网页接收语言

2.4.3 你的计算机的 IP 地址是什么？你所浏览的网站服务器的 IP 地址是什么？
答：我们通过 ping 指令得知在网线情况下，以太网 IP 地址为 172.29.36.190。网站服务器 IP 地址为 121.36.34.25。

2.4.4 服务器返回给浏览器的状态码（Status code）是什么？代表了什么含义？
答：根据表 2，状态码有 200、302 等等，其中 200 代表客户端请求成功，302 代表重定向等等。

2.4.5 你所访问的 HTML 文件在服务器上最后的修改时间是什么？
答：Fn+F12 进入网页配置，在 Network 下面点击文件即可观察日期。如图 31 所示，在选定区域，我们的最后修改时间为 2021.4.1 17:30:48.

2.4.6 浏览器发出了多少个 HTTP GET 请求？
答：如图 28 所示，我们的浏览器发出了 6 个 GET 请求，分别是请求访问、请求 css、请求三个图片以及网站图标。

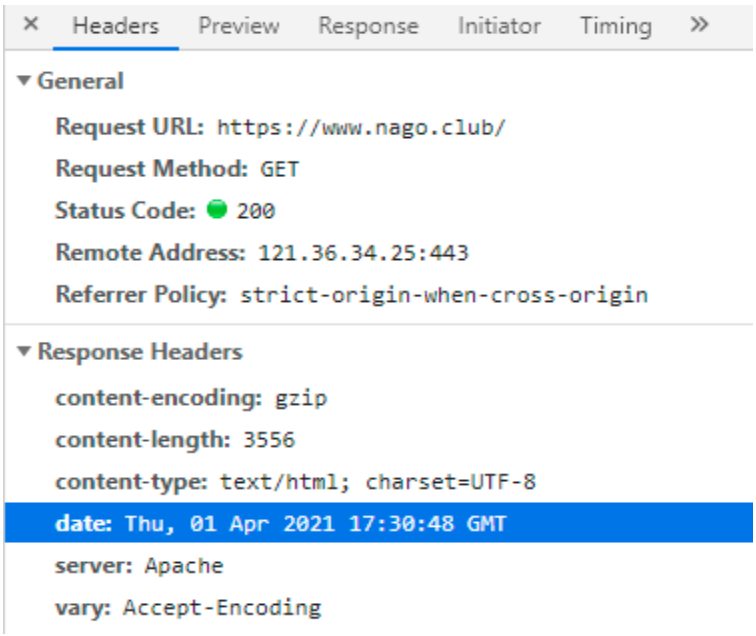


图 31. 获取网页修改时间

2.4.7 观察浏览器中发出的第 1 个 HTTP GET 请求，其中是否含有“IF-MODIFIED-SINCE”这一行？
答：Last-Modified 与 If-Modified-Since 都是用来记录页面的最后修改时间。
当客户端访问页面时，服务器会将页面最后修改时间通过 Last-Modified 标识由服务器发往客户端，客户端记录修改时间，再次请求本地存在的 cache 页面时，客户端会通过 If-Modified-Since 头将先前服务器端发过来的最后修改时间戳发送回去，服务器端通过这个时间戳判断客户端的页面是否是最新的，如果不是最新的，则返回新的内容，如果是最新的，则 返回 304 告诉客户端其本地 cache 的页面是最新的，于是客户端就可以直接从本地加载页面了，这样在网络上传输的数据就会大大减少，同时也减轻了服务器的负担。
事实上，大部分的网站都是流动的数据，这样势必导致网页一直在做修改，所以不会有 If-Modified-Since，或者说不必设置这样一个量。极个别的网站，数据流动量很小，设

置这样一个量。实验过程中的网站没有 IF-MODIFIED-SINCE，但是我们可以试着访问一些小型网站，得到了如图 32 所示的结果。

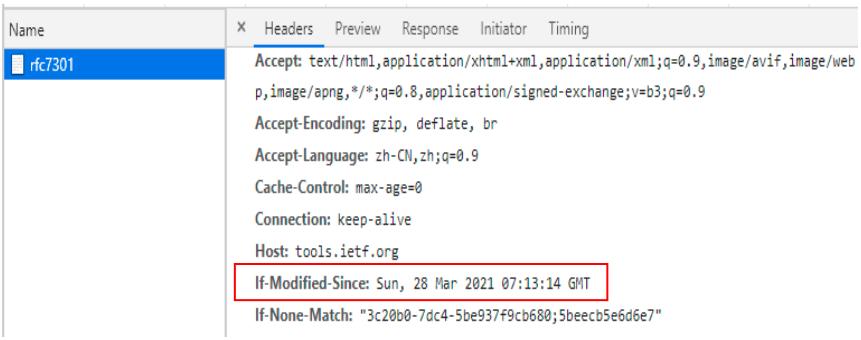


图 32. IF-MODIFIED-SINCE 显示值

2.4.8 观察服务器响应报文的内容，服务器是否显示地返回了所请求的文件内容？你如何解释这种现象？

答：没有全部返回。如图 33 所示我们发现了 404 错误代码，出现这个原因是网站没有设置这个名为 facicon 的文件，所以我们没有得到。

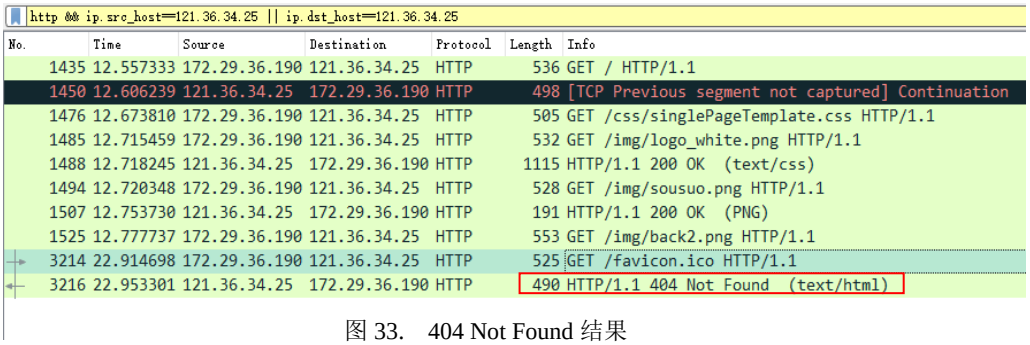


图 33.　404 Not Found 结果

2.4.9 HTTP 客户端从服务器端获取网页对象使用的命令是什么？该命令每次能获取一个完整的网页吗？

答：使用 GET 指令。如图 34 所示，我们切换到 login.php 界面之后，显示的指令是 GET。同时，由于后面又有很多 GET 指令在申请资源，所以该命令不能一次性获取一个完整的网页。

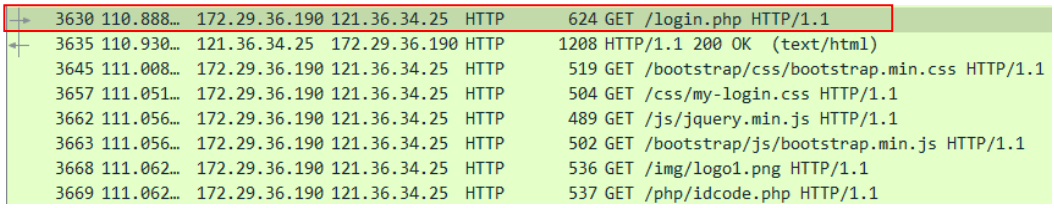


图 34. GET 指令获取网页

2.4.10 浏览一个网页需要进行几次 TCP 连接？连接次数与什么有关？

三次，俗称“三次握手”。两次握手建立连接，发送第一个连接时，因为网络或其他原因导致请求没有发送到服务端，从而客户端发送第二个连接请求，建立连接。并且成功与服务端通信，这时服务端收到了第一次发送的连接请求，并且又建立了连接，但是这次连接并不会发送数据而造成不必要的资源消耗。在 3.6 部分会详细说明。

3. TCP 协议的认知与分析

3.1 抓取一个 TCP 会话所传输的报文，并分析报文的结构、格式及其内容。

我们依然访问之前的 http 协议网站。选择其中一个 http 报文，右键然后点击追踪流，然后点击 TCP 流，可以看到 TCP 三次握手过程。点击其中一个 TCP 报文，下面我们详细叙述报文结构。

1） Source Port：源端口号，如图 35 所示，源端口号为 c9 dc，2 字节 16bit；

2） Destination port：目的端口号，如图 36 所示，目的端口号为 00 50，2 字节 16bit；

3） Sequence number：序列号，如图 37 所示，序列号为 2a ef c9 0d，4 字节 16bit；

4） Acknowledgement number：确认序号，如图 38 所示，序列号为 00 00 00 00，4 字节 16bit。由于该报文为 SYN 报文，ACK 标志为 0，所以序列号为 0，在后面会有详细一些的说明。

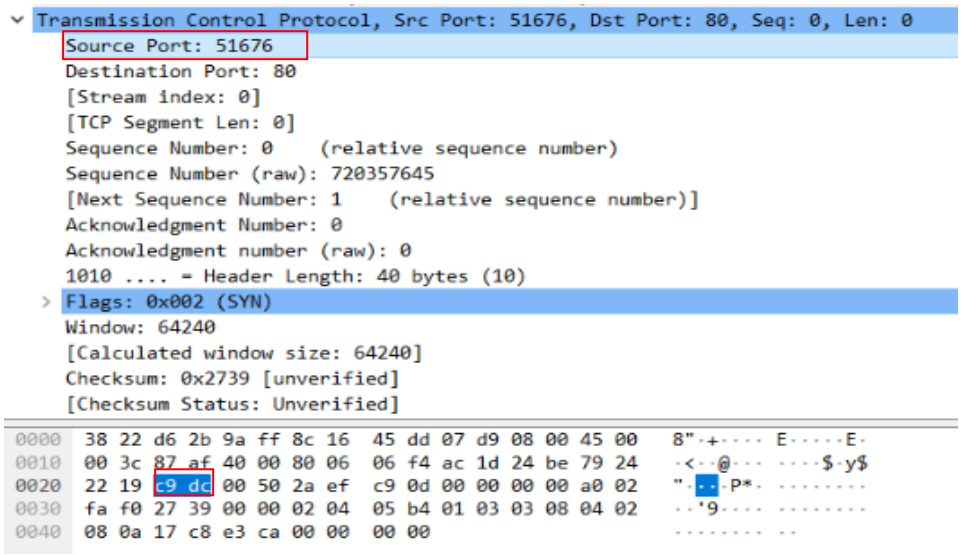


图 35. TCP 报文源端口号

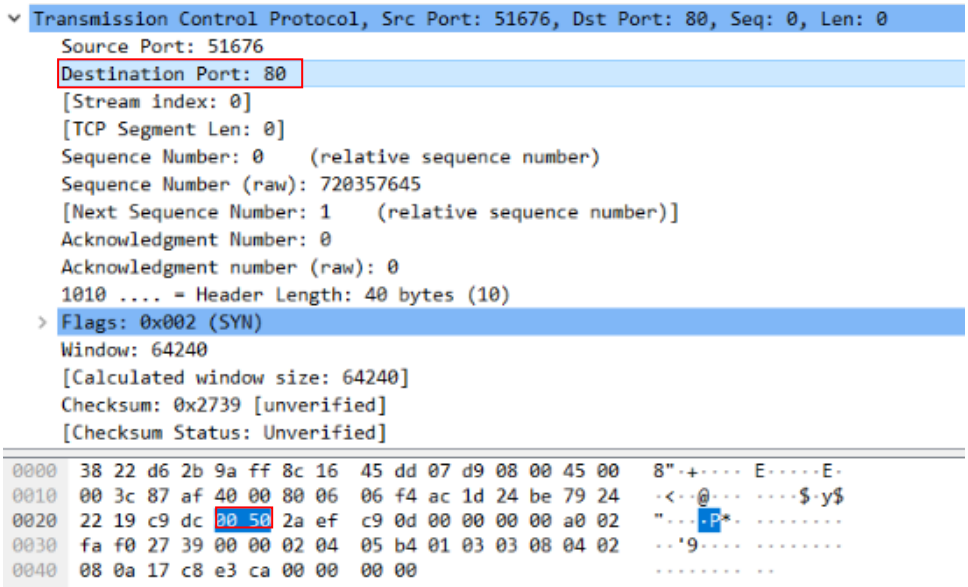


图 36. TCP 报文目的端口号

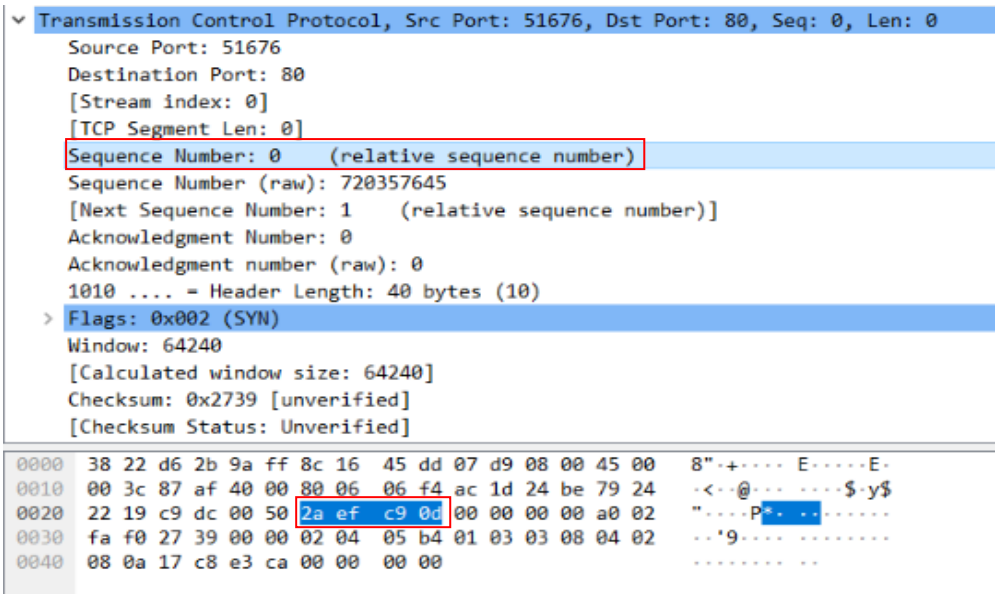


图 37. TCP 报文序列号

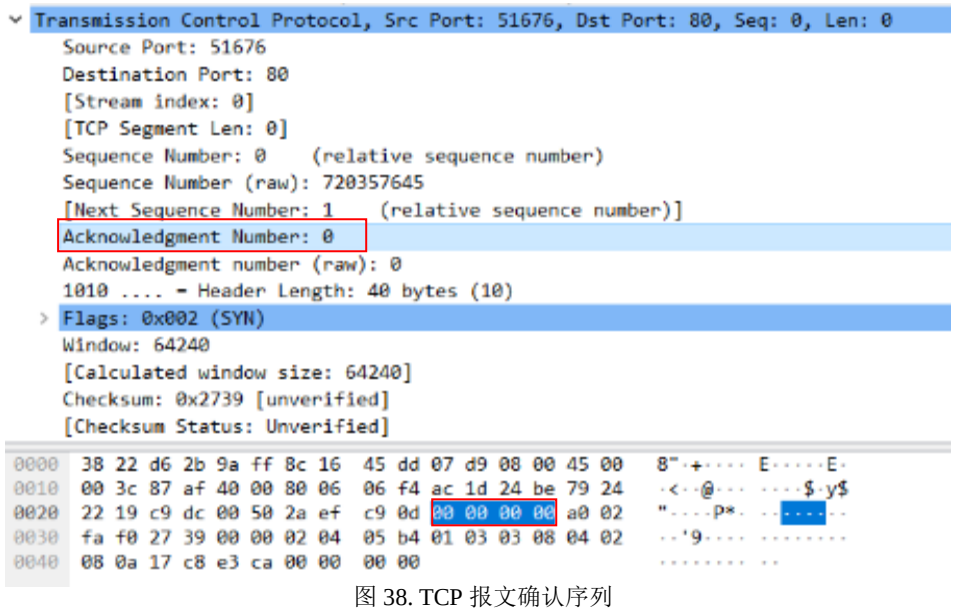


图 38. TCP 报文确认序列

5） Header length：报头长度，这里面显示为 40bytes；

6） Flag：标志位，如图 39 所示。

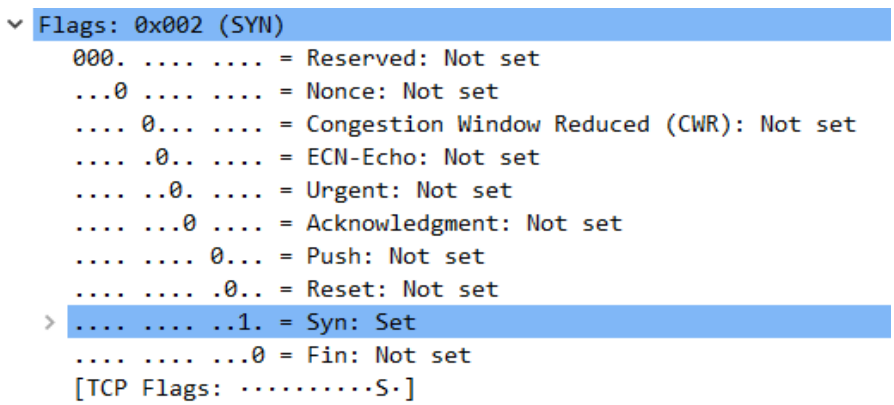


图 39. 标志位构成

标志位：

U / Urgent：紧急指针有效性标志；

A / Acknowledgment：确认序号有效性标志，一旦一个连接建立起来，该标志总被置为 1，即除了请求建立连接报文（仅设置 SYN 标志位为 1），其它所有报文的该标志总为 1；

P / Push：Push 标志（接收方应尽快将报文段提交至应用层）；

R / Reset：重置连接标志；

S / Syn：同步序号标志；

F / Fin：传输数据结束标志。

由于我们这个报文属于请求建立连接报文，所以仅仅设置 SYN 标志位为 1。

7） Windows：窗口大小（2 字节）：TCP 流量控制通过连接的每一端声明窗口大小进行控制（接收缓冲区大小）。两字节所能表示最大整数为$2^{16}-1=65535$，这里窗口大小为 64240.

8） Checksum and urgent pointer：校验和与紧急指针，如图 40 和图 41 所示，二者均为两字节。当 Urgent 标志置 1 时，紧急指针才有效。

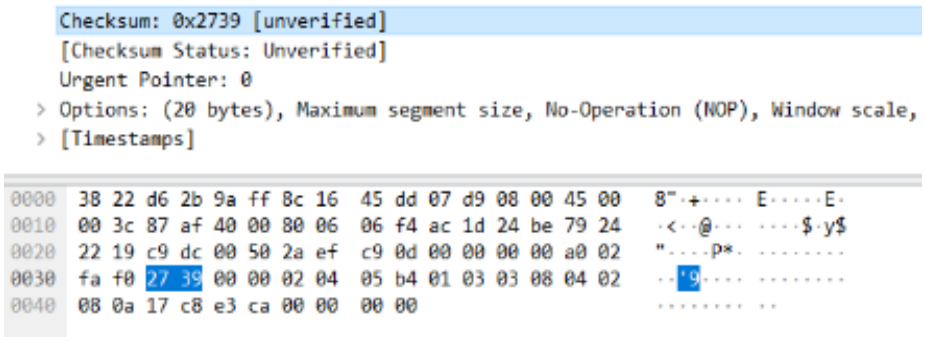


图 40. TCP 报文校验和

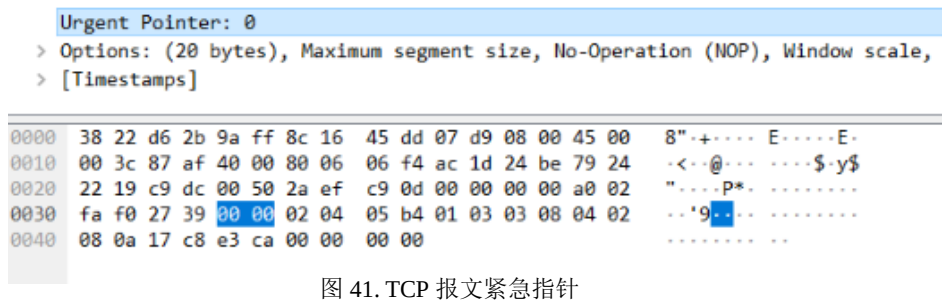


图 41. TCP 报文紧急指针

9） Options：任选字段。

TCP 报文格式总体上来看，如图 42 所示。

3.2 捕获 TCP 协议通过三次握手建立连接过程完整数据包，截图粘贴到报告中，并分析该 TCP 连接的时序关系。（注意：源 IP 地址与目的 IP 地址的匹配，特别是源 PORT 和目的 PORT 匹配）

我们分析一下之前提及的三次握手原则。三次握手发生于客户端请求服务器资源的时候，如图 43 所示。

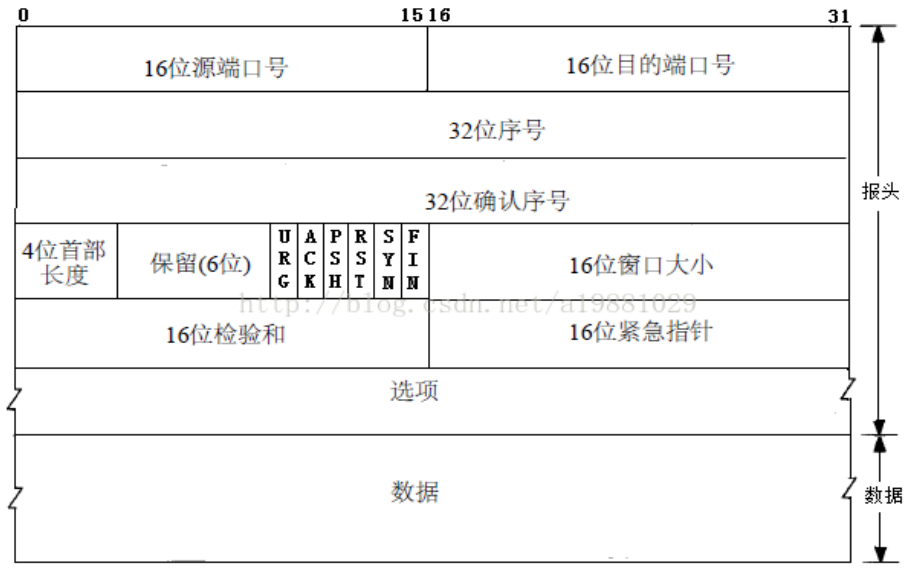


图 42. TCP 报文格式

No.	Time	Source	Destination	Protocol	Length	Info
26	4.314218	172.29.36.190	121.36.34.25	TCP	74	51679 → 80 [SYN] Seq=0 Win=64240
31	4.352683	121.36.34.25	172.29.36.190	TCP	74	80 → 51679 [SYN, ACK] Seq=0 Ack=
32	4.352761	172.29.36.190	121.36.34.25	TCP	66	51679 → 80 [ACK] Seq=1 Ack=1 Win
33	4.353082	172.29.36.190	121.36.34.25	HTTP	533	GET /img/sousuo.png HTTP/1.1

图 43. TCP 三次握手

点击三次 TCP 报文，我们观察到图 44(a)(b)(c)所示。

tcp.stream eq 3

No.	Time	Source	Destination	Protocol	Length
26	4.314218	172.29.36.190	121.36.34.25	TCP	74
31	4.352683	121.36.34.25	172.29.36.190	TCP	74
32	4.352761	172.29.36.190	121.36.34.25	TCP	66
33	4.353082	172.29.36.190	121.36.34.25	HTTP	533

```
> Frame 26: 74 bytes on wire (592 bits), 74 bytes captured (
> Ethernet II, Src: LCFCHeFe_dd:07:d9 (8c:16:45:dd:07:d9), D
> Internet Protocol Version 4, Src: 172.29.36.190, Dst: 121.
v Transmission Control Protocol, Src Port: 51679, Dst Port:
    Source Port: 51679
    Destination Port: 80
```

图 44(a). TCP 三次握手（第一次）

tcp.stream eq 3

No.	Time	Source	Destination	Protocol	Length
26	4.314218	172.29.36.190	121.36.34.25	TCP	74
31	4.352683	121.36.34.25	172.29.36.190	TCP	74
32	4.352761	172.29.36.190	121.36.34.25	TCP	66
33	4.353082	172.29.36.190	121.36.34.25	HTTP	533

```
> Frame 31: 74 bytes on wire (592 bits), 74 bytes captured (
> Ethernet II, Src: Hangzhou_2b:9a:ff (38:22:d6:2b:9a:ff), D
> Internet Protocol Version 4, Src: 121.36.34.25, Dst: 172.2
v Transmission Control Protocol, Src Port: 80, Dst Port: 516
    Source Port: 80
    Destination Port: 51679
```

图 44(b). TCP 三次握手（第二次）

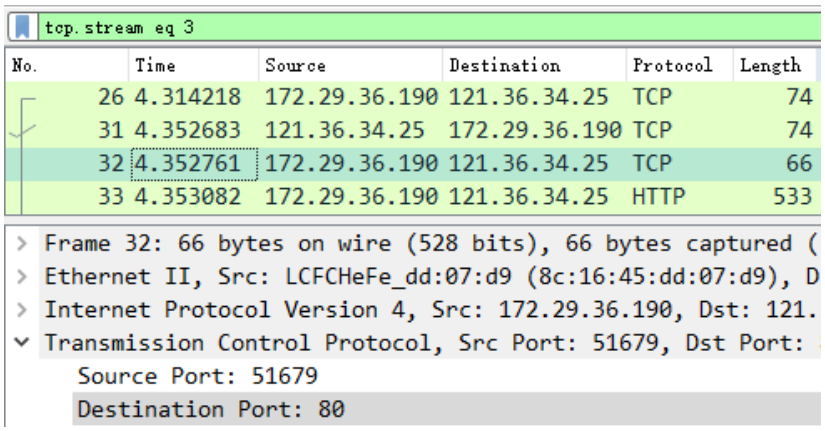


图 44(c). TCP 三次握手（第三次）

图 44. TCP 三次握手

1） 如图 44 所示，在 Time 一栏可以观察握手发生的时间。第一次握手发生于 4.314218，第二次握手发生于 4.352683，第三次握手发生于 4.352761，体现了三次握手有一个时序的变化。

2） 在 Source 和 Destination 两栏，可以看到握手两侧 IP 地址的变化。

第一次握手，源 IP 地址为 172.29.36.190，也就是本机 IP 地址，目的 IP 地址为 121.36.34.25，也就是网站的 IP 地址，体现了过程 1。且来源端口 source port 为 51679，目的端口为 80；

第二次握手，源 IP 地址为 121.36.34.25，也就是本机 IP 地址，目的 IP 地址为 172.29.36.190，也就是网站的 IP 地址，体现了过程 2。且来源端口 source port 为 80，目的端口为 51679；

第三次握手，源 IP 地址为 172.29.36.190，也就是本机 IP 地址，目的 IP 地址为 121.36.34.25，也就是网站的 IP 地址，体现了过程 3。且来源端口 source port 为 51679，目的端口为 80；

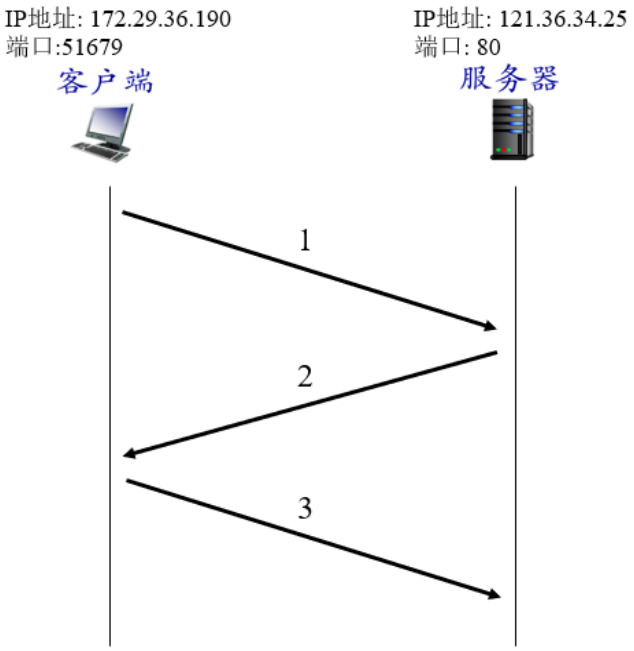


图 45. 三次握手图解

用图形化的方法说明三次握手如图 45 所示。

3.3 在 TCP 的三次握手过程中，试分析 SYN、ACK、Sequence number 之间的关系。

我们再深入看一下三次握手的数据包。

第一次握手：如图 46 所示，Sequence number 为 2830103598，SYN 位置 1；

第二次握手：如图 47 所示，Sequence number 为 390012788，SYN 位置 1，ACK 位置 1，Acknowledgement number 为 2830103599，等于 Sequence number 加一；

第三次握手：如图 48 所示，Acknowledgement number 为 390012789，等于 Sequence number 加一；ACK 位置 1；Sequence number 为初始的 Sequence number 加一。

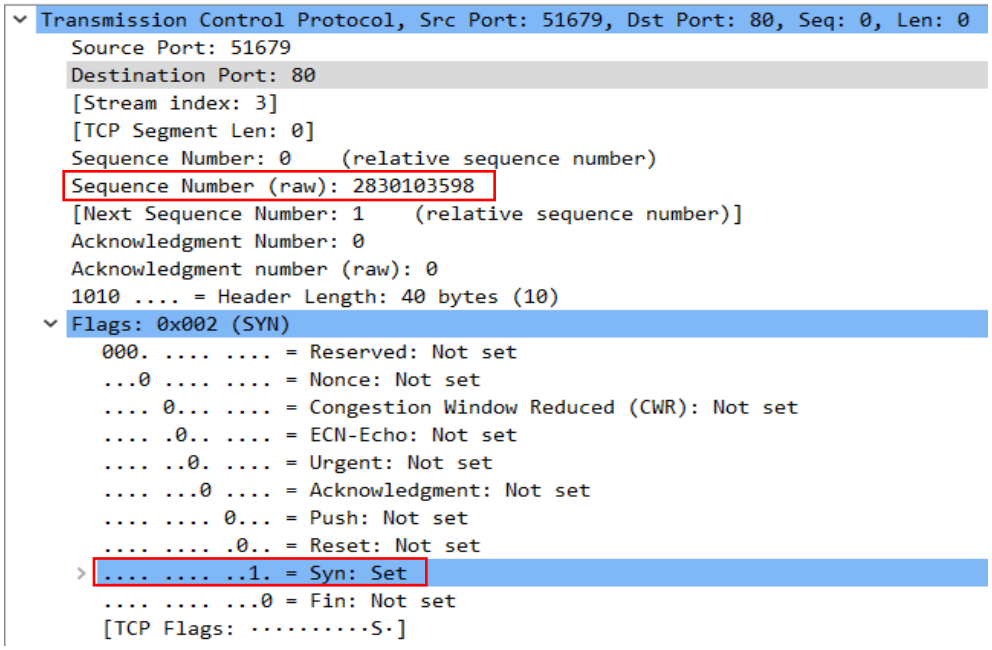


图 46. 第一次握手

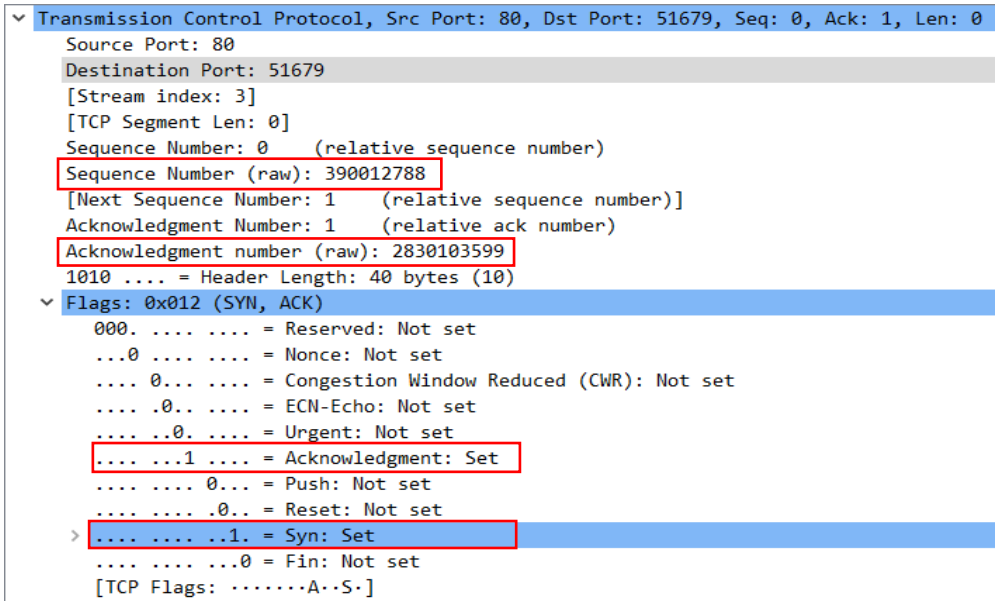


图 47. 第二次握手

```
Transmission Control Protocol, Src Port: 51679, Dst Port: 80, Seq: 1, Ack: 1, Len: 0
    Source Port: 51679
    Destination Port: 80
    [Stream index: 3]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 2830103599
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 390012789
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x010 (ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
      .... .... ...0 = Fin: Not set
      [TCP Flags: ·····A····]
```

图 48. 第三次握手

综上，我们解释一下三次握手的过程：

第一次握手：客户端给服务端发一个 SYN 报文，同步位 SYN 置 1，初始序号 seq=x，SYN=1 的报文段不能携带数据，但要消耗掉一个序号。

第二次握手：服务器收到客户端的 SYN 报文之后，会以自己的 SYN 报文作为应答，同时会把客户端的 SYN+1 作为 ACK 的值，表示自己已经收到了客户端的 SYN。在确认报文段中 SYN=1，ACK=1，确认号 ack=x+1，初始序号 seq=y；

第三次握手：客户端收到 SYN 报文之后，会发送一个 ACK 报文，当然，也是一样把服务器的 SYN+1 作为 ACK 的值，表示已经收到了服务端的 SYN 报文，此时客户端处于 ESTABLISHED 状态。服务器收到 ACK 报文之后，也处于 ESTABLISHED 状态，此时，双方已建立起了连接。

确认报文段 ACK=1，确认号 ack=y+1，序号 seq=x+1（初始为 seq=x，第二个报文段所以要加一），ACK 报文段可以携带数据。

整个过程如图 49 所示。

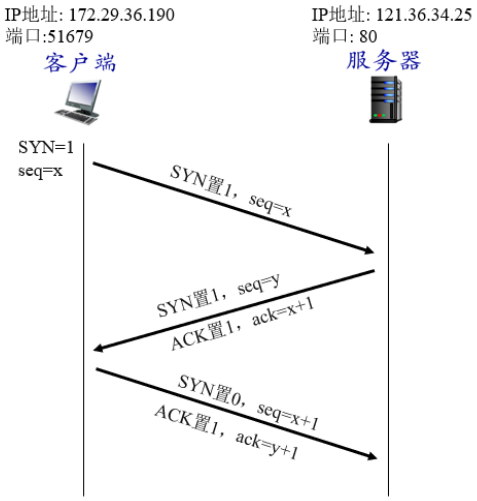


图 49. 三次握手全过程

3.4 报文的序号以什么为单位？它在数据传输过程中的作用是什么？不同会话中报文的起始序号都是一样的吗？

报文以 bit 为单位，例如在 TCP 协议中，TCP 报文为 32 位序号。序号用来标识 TCP 发端向 TCP 收端发送的数据字节流，不同的会话起始序号不同，如图 50 所示。

第一个会话的起始序号为 2404303133

第二个会话的起始序号为 227861847

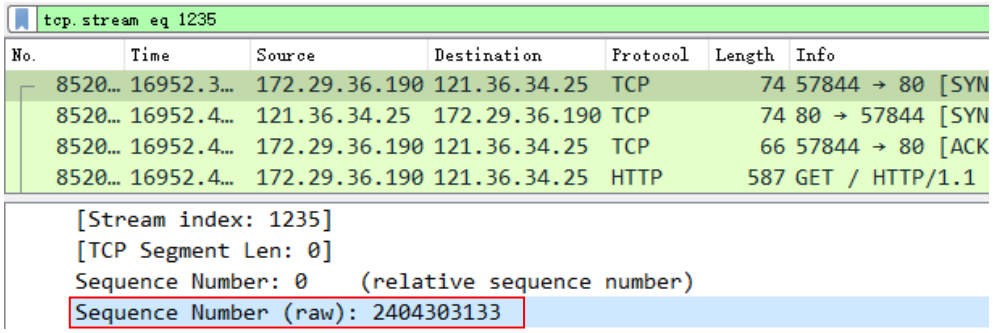


图 50(a).会话 1 的序列号

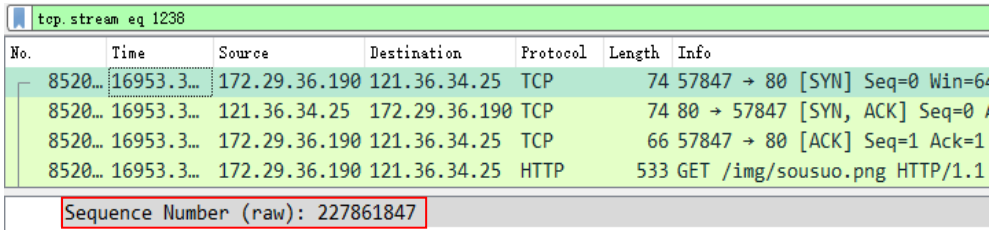


图 50(b).会话 2 的序列号

图 50. 不同的会话有不同的序列号

3.5 请根据以上捕获到的 TCP 数据报，分析 TCP 报文组成格式（即包含哪些部分）

根据 3.1 中得叙述，我们对抓取的报文格式进行了整体的分析，可以得知报文由源端口号、目的端口号、序号、确认序号、首部长度、窗口大小、检验和、紧急指针、选项以及数据构成，如图 51 所示。

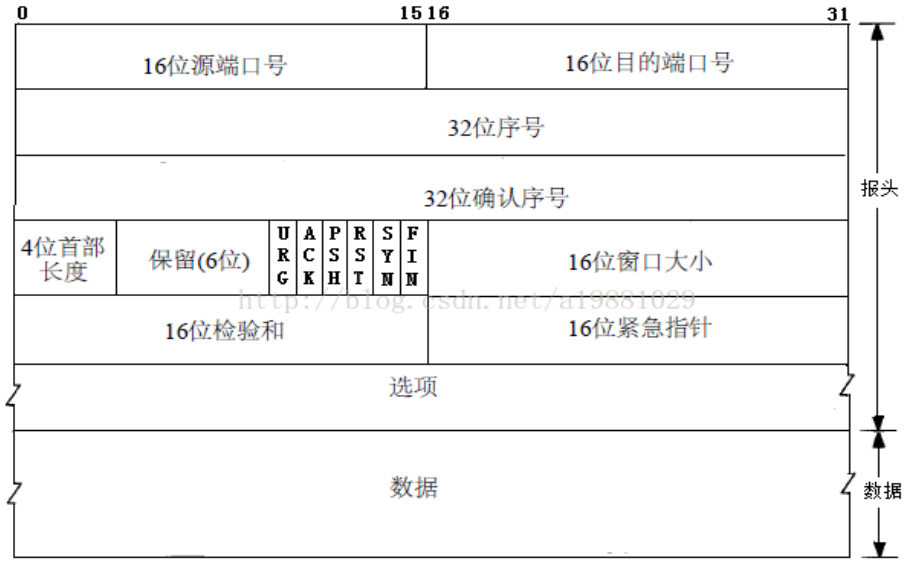


图 51. TCP 报文格式

3.6 关于三次握手和四次挥手的补充说明

我们都已经知道建立连接的时候需要三次握手，断开连接的时候需要四次挥手。但是这是什么原因呢？

首先来说握手的问题。

1） 当客户端要求建立连接的时候，我们需要告知服务器，告知方法就是 SYN=1，并且给一个序号 seq，这就是第一次握手；

2） 此时我们需要服务器给我们一个回信，这样我们才能确定服务器接收到了我们的请求，也就是确定我们能联系到服务器。所以这时候 ACK=1 并将 seq 加一作为确认序号 ack，然后用一个新的值来替代序号 seq，这就是第二次握手，为了确定我们能联系到服务器；

3） 我们收到了服务器的答复之后我们就知道我们能联系到服务器，但我们也需要让服务器直到它能联系到我们，所以我们同样将 ACK 置 1，然后将 ack 设置为新的 seq 值加一，这样服务器也能确定它能联系到我们。这就是第三次握手，为了让服务器确定它能联系到我们。

然后我们再说一下挥手的问题。

1） 我们需要断开和服务器的连接，客户端进程发出连接释放报文，并且停止发送数据。如图 52 所示，释放数据报文首部，FIN=1，其序列号为 seq=74165064；源 IP 为网站 IP（121.36.34.25），也就是我们接收到了 FIN 报文通知，它表示网站方没有数据再发送给我们了。此时网站想要和我们断开连接。

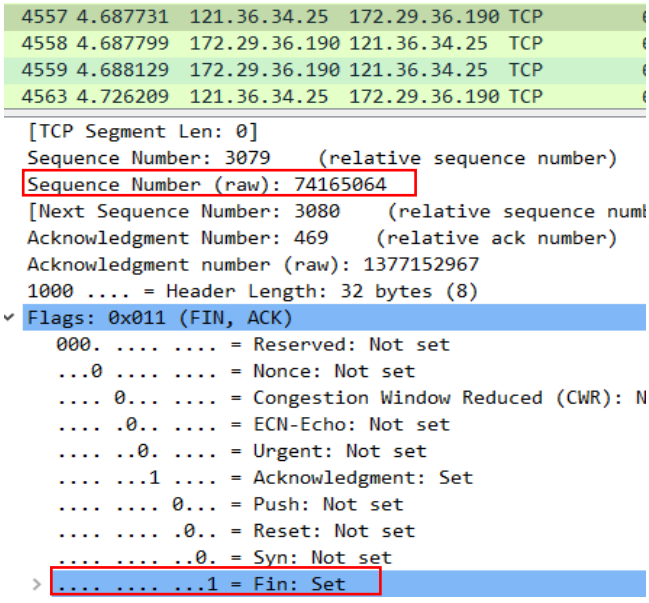


图 52. TCP 第一次挥手

2） 当我们客户端收到网站服务器端的 FIN 包后，就知道服务器想要断开连接了，于是返回一个 ACK 包，设置 seq 加一为 74165065。现在还不能断开连接，因为我们客户端还没有发完全部的数据。这就是第二次挥手，如图 53 所示。

3） 当我们客户端发送一个 FIN 报文，设置 seq 序列号为 1377152967，表示我们的数据都已经发送完了，服务器端可以断开连接了，如图 54 所示。

4） 最后，服务器端收到客户端的 FIN 包后，返回一个 ACK 包，设置 seq 序列号加一为 1377152968，相当于服务器端接收到了我们的确认信息，连接断开，如图 55 所示。

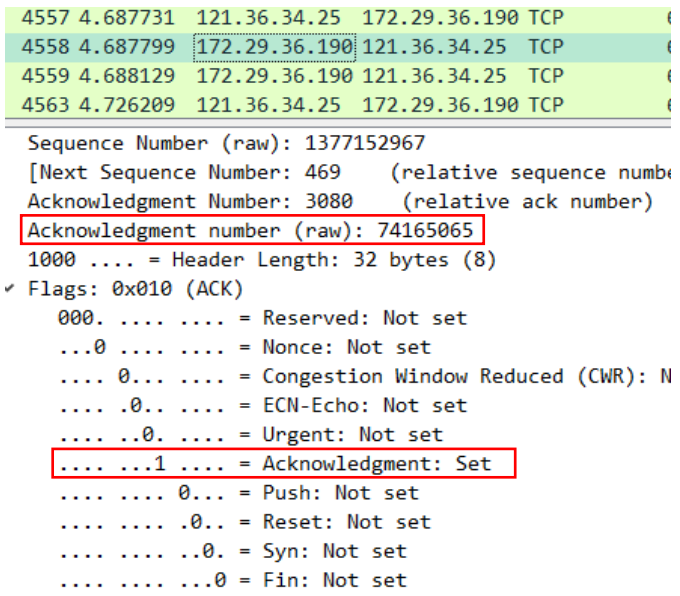


图 53. 第二次挥手

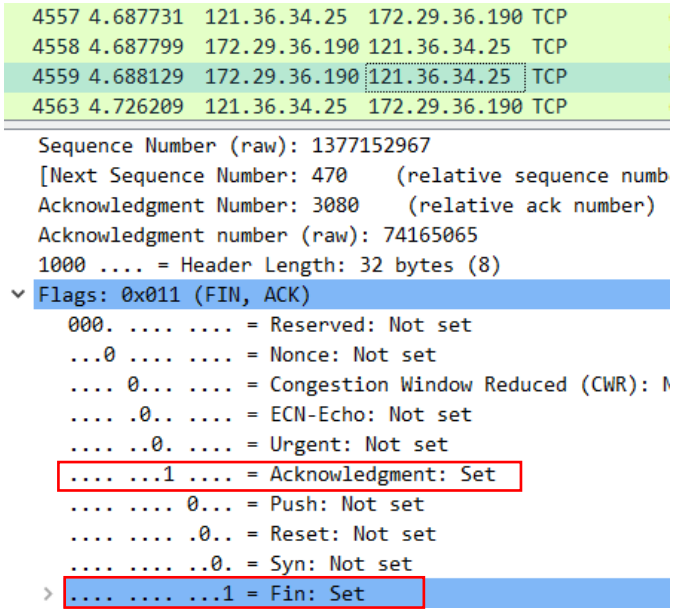


图 54. 第三次挥手

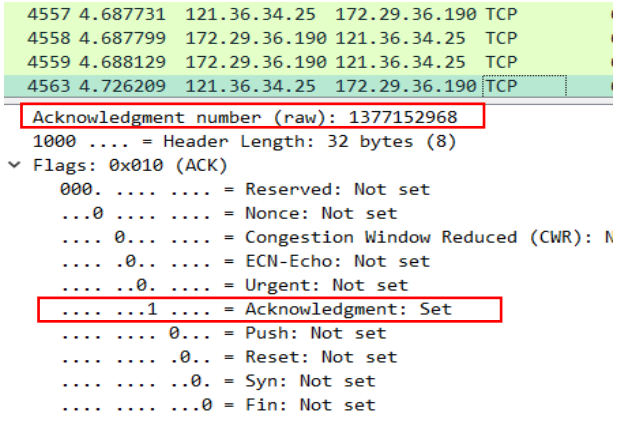


图 55. 第四次挥手

为什么我们需要四次挥手而不是三次挥手呢？因为第二次挥手和第三次挥手不能合并，我们只能先回复一个 ACK 报文，告诉服务器我们所有的报文还没发送完，等到发送完了才会返回一个 FIN，二者不能不能同时发送，所以分两步进行。

四次挥手过程如图 56 所示。值得一提的是，在 C/S 架构中，client 和 server 都可以中断连接，而请求连接只有 client 可以进行。

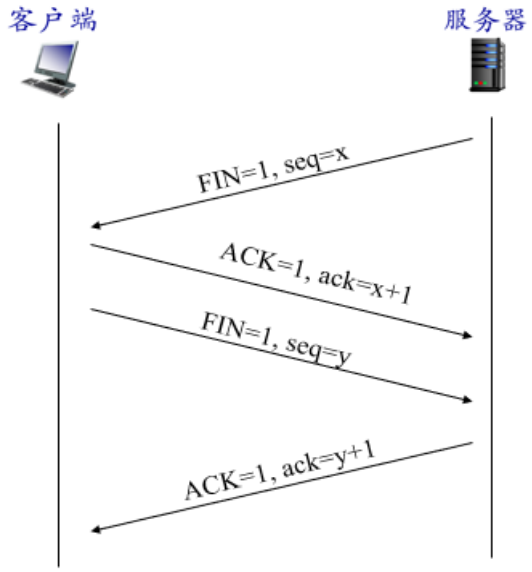


图 56. 四次挥手过程示意图

4. ICMP(Internet Control Message Protocol)协议的简单分析

4.1 ICMP 协议简介

ICMP 协议是一个网络层协议。一个新搭建好的网络，往往需要先进行一个简单的测试，来验证网络是否畅通；但是 IP 协议并不提供可靠传输。如果丢包了，IP 协议并不能通知传输层是否丢包以及丢包的原因。所以我们就需要一种协议来完成这样的功能–ICMP 协议。

4.2 ICMP 的报文格式

ICMP 报文格式如图 57 所示。

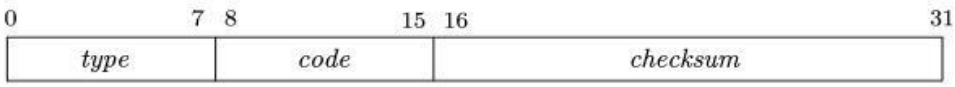


图 57. ICMP 报文格式

如图 56 所示，ICMP 报文由类型(type)、代码(code)、校验和(checksum)组成。其中，类型有 8bit 组成，代码由 8bit 组成，校验和由 16bit 组成。

ICMP 的类型有很多，它们都用数字表示。8bit 能储存的最大数为 255，1-127 一般为差错报文，128 以上为信息报文。

ICMP 的代码部分占 1 字节，标识对应 ICMP 报文的代码。它与类型字段一起共同标识了 ICMP 报文的详细类型。

ICMP 的校验和部分是对包括 ICMP 报文数据部分在内的整个 ICMP 数据报的校验和，以检验报文在传输过程中是否出现了差错。

4.3 ICMP 检验主机是否可达

为了检测 ICMP 的作用，我们现在 Wireshark 中设置过滤条件为 icmp，然后我们先

用 cmd 指令去 ping 一个网址，如图 58 所示。

```
C:\Users\未央>ping www.bing.com

正在 Ping china.bing123.com [202.89.233.100] 具有 32 字节的数据:
来自 202.89.233.100 的回复: 字节=32 时间=38ms TTL=117
来自 202.89.233.100 的回复: 字节=32 时间=38ms TTL=117
来自 202.89.233.100 的回复: 字节=32 时间=38ms TTL=117
来自 202.89.233.100 的回复: 字节=32 时间=38ms TTL=117

202.89.233.100 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 = 38ms，最长 = 38ms，平均 = 38ms
```

图 58. ping www.bing.com

然后我们观察 Wireshark 里面的变化，如图 59 所示。

icmp

No.	Time	Source	Destination	Protocol	Length	Info
35	8.439161	172.29.36.190	202.89.233.…	ICMP	74	Echo (ping) request id=0x0001, seq=141/36096, ttl=128 (reply in 36)
36	8.477188	202.89.233.…	172.29.36.190	ICMP	74	Echo (ping) reply id=0x0001, seq=141/36096, ttl=117 (request in 35)
40	9.446873	172.29.36.190	202.89.233.…	ICMP	74	Echo (ping) request id=0x0001, seq=142/36352, ttl=128 (reply in 41)
41	9.484801	202.89.233.…	172.29.36.190	ICMP	74	Echo (ping) reply id=0x0001, seq=142/36352, ttl=117 (request in 40)
42	10.456092	172.29.36.190	202.89.233.…	ICMP	74	Echo (ping) request id=0x0001, seq=143/36608, ttl=128 (reply in 43)
43	10.494093	202.89.233.…	172.29.36.190	ICMP	74	Echo (ping) reply id=0x0001, seq=143/36608, ttl=117 (request in 42)
46	11.465049	172.29.36.190	202.89.233.…	ICMP	74	Echo (ping) request id=0x0001, seq=144/36864, ttl=128 (reply in 47)
47	11.502955	202.89.233.…	172.29.36.190	ICMP	74	Echo (ping) reply id=0x0001, seq=144/36864, ttl=117 (request in 46)

图 59. icmp 过滤条件

可以看到，我们的 Wireshark 中也出现了四组 request-reply 的组，而且 Destination 中也可以看到 bing 的 IP 地址（202.89.233.100）。我们作为 request 方，每次都是我们发送请求，然后服务器给我们答复。而且，在图 59 中也可以看得到 reply 的过程中 TTL 的值（图中标记处）和图 58 中也是对应的。

实验结果：

本次实验通过 Wireshark 网络嗅探器分析了 HTTP 协议报文结构格式、TCP 协议报文格式、TCP 三次握手、TCP 四次挥手、ICMP 协议报文，加深了对于三种协议的理解。

实验小结：

1. 本次实验通过学习 HTTP 协议、TCP 协议、ICMP 协议，对网络协议有了一个相对深的认识。之前上网的时候并不知道这些协议的事情，再经过本次实验之后脑海中有了一个大致的印象，尤其关于 3 次握手 4 次挥手等抽象概念的理解；
2. 本次实验最开始的时候难点在于 http 协议网站的稀有性。目前主流网站都是 https 协议，都是被加密过了的。之后的难点在于三次握手四次挥手的理解；
3. 本次实验虽内容较多，但是实验之后获得了比较多的理解。

指导教师批阅意见：

成绩评定：

指导教师签字：邹永攀

年　　月　　日

备注：