南開大學
Nankai University

南　　开　　大　　学

计　算　机　学　院

计算机网络实验报告

分析 WEB 服务器 HTTP 交互过程

姓名：姚知言

年级：2022 级

专业：计算机科学与技术

指导教师：张建忠 徐敬东

2024 年 11 月 1 日

摘要

本次实验通过制作 WEB 界面，搭建 WEB 服务器，并使用 Wireshark 捕获浏览器与 WEB 服务器的交互过程，更进一步的理解三次握手、四次挥手等概念。

关键词：WEB，服务器搭建，Wireshark，服务器交互

目录

一、 实验要求

1. 搭建 WEB 服务器（自由选择系统），并制作简单的 WEB 页面，包含简单文本信息（至少包含专业、学号、姓名）、自己的 LOGO、自我介绍的音频信息。
2. 通过浏览器获取自己编写的 WEB 页面，使用 Wireshark 捕获浏览器与 WEB 服务器的交互过程，使用 Wireshark 过滤器使其只显示 HTTP 协议。
3. 现场演示。
4. 提交 HTML 文档、Wireshark 捕获文件和实验报告，对 HTTP 交互过程进行详细说明。
5. 页面不要太复杂，包含所要求的基本信息即可。使用 HTTP，不要使用 HTTPS。

二、 实验环境

本次实验在 x86-64 架构物理机中进行。
使用 Internet Information Services（IIS）进行服务器搭建。
使用 Wireshark 工具，捕获交互过程。
使用 Google 浏览器连接服务器。
相关系统及应用程序版本如下：

Windows 及 IIS	Windows11,23H2
Wireshark	4.4.1
Google 浏览器	130.0.6723.92

表 1: 实验环境版本

三、 WEB 页面搭建

(一) 内容介绍

html 文件命名为”index.html”，以便后续搭建服务器识别为入口文件。并将后续要用到的图片、音频等都放在同一根目录下，以便后续间接寻址使用。

整个网页除标题外，主要分为三个部分：个人信息，LOGO，自我介绍，并使用少量 CSS 辅助排版。

标题为“姚知言的个人主页”。

1. 个人信息

个人信息部分包括姓名，学号，年级，学院，专业的基本信息，每一条一个 <p> 段落块。

2. LOGO

LOGO 部分使用本人的个人微信头像“logo.jpg”。

3. 自我介绍

自我介绍部分包括一段 6s 的 mp3 格式音频“intro.mp3”，内容为” 大家好，我是计算机学院的姚知言。”，使用 AI 朗读翻录生成。

(二) 源码总览

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>姚知言的个人主页</title>
        <style>
            *{
                padding:0;
                font-family:Microsoft YaHei;
            }
            h1{
                font-size:38px;
                text-align:center;
            }
            .p1{
                margin:20px 50px;
            }
            .p2{
                margin:15px 0px;
            }
            .b1{
                text-align:center;
                width:33%;
                margin:15px 0px;
                font-size:20px;
                color:aliceblue;
            }
            .a1{
                color:aliceblue;
                text-decoration: none;
            }
            .bt{
                margin:11.8px 20px;
                color:orchid;
                font-size:20px;
            }
        </style>
    </head>
    <body>
            <h1 id="top">姚知言的个人主页</h1>
        </div>
        <div style="display:flex;margin:0;height:50px;background-color:
            orchid">
```

```
            <p class="b1"><a href="#C1" class="a1">个人信息</a></p>
            <p class="b1"><a href="#C2" class="a1">LOGO</a></p>
            <p class="b1"><a href="#C3" class="a1">自我介绍</a></p>
        </div>
        <div style="display:flex;margin:20px 40px">
            <div>
                <div id="C1" class="p1">
                    <div style="display:flex">
                        <p class="bt">个人信息</p>
                    </div>
                    <p>姓名：姚知言</p>
                    <p>学号：2211290</p>
                    <p>年级：2022级</p>
                    <p>学院：计算机学院</p>
                    <p>专业：计算机科学与技术</p>
                </div>
                <div id="C2" class="p1">
                    <div style="display:flex">
                        <p class="bt">LOGO</p>
                    </div>
                    <div>
                        <img src="logo.jpg" width="20%" length="auto">
                    </div>
                </div>
                <div id="C3" class="p1">
                    <div style="display:flex">
                        <p class="bt">自我介绍</p>
                    </div>
                    <audio controls>
                        <source src="intro.mp3" type="audio/mpeg">
                    </audio>
                </div>
            </div>
        </div>
    </body>
</html>
```

(三) 网页展示

网页展示图如图1所示。

图 1: 网页展示图

四、 WEB 服务器搭建

(一) 搭建流程

打开 IIS，在左侧边栏中选择” 网站” 后如图2所示。

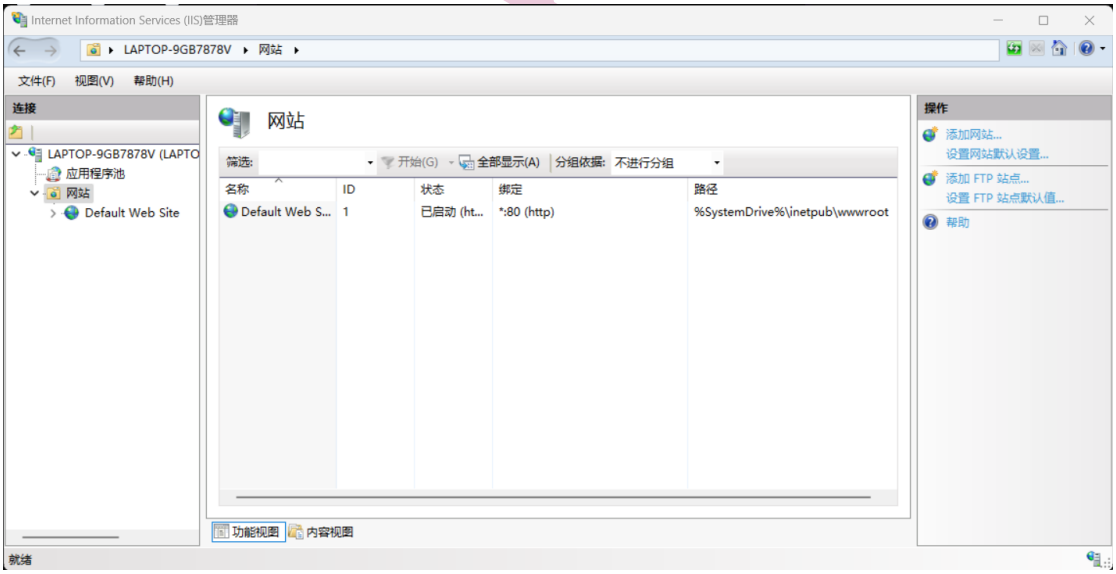

图 2: IIS 主界面

点击右边栏” 添加网站”，如图3所示。填写网站名称为”networklab2”，设置物理路径为 html 网页的根目录。设定 IP 地址为“全部未分配”（含义是本机全部可分配的地址均可打开该网站），端口设置为“1101”（80 以上未分配的端口均合法，这里使用检查实验的日期）。点击确定，完成服务器的搭建。

图 3: IIS 服务器搭建

(二) 使用方式

理论上来说，可以使用 localhost 或者任何本机可用的 IP 地址访问网页。为便于区分，使用校园网 WLAN 的 IP 地址进行实验，如图4所示。这一网址与先前网页展示（图1）相对应。

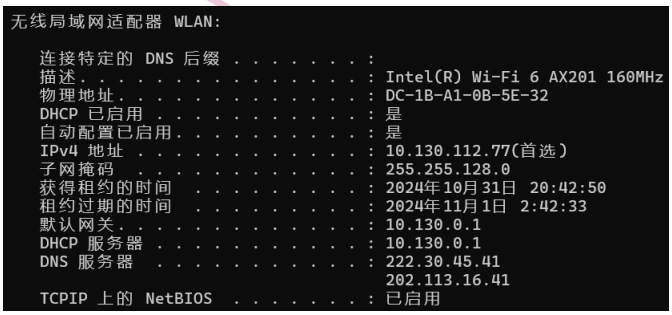

图 4: 无线局域网 IP 地址

五、 Wireshark 抓包

(一) 抓包流程

注：每次抓包前已经清理过浏览器缓存，以避免可能存在的由于存在缓存而不需要进行通信的情况。

打开 Wireshark，使用 Adapter for loopback traffic capture 抓包（用于捕获本机自己的网络）。

在浏览器中输入“10.130.112.77:1101”进入网站，播放音频后，退出网站，并关闭捕获查看结果。

将捕获结果保存为”捕获结果.pcapng”（在 26 秒内共 280 条结果），然后通过在过滤器中填入“ip.addr==10.130.112.77”，查找有关我们建立的 WEB 服务器的结果，保存为”捕获结果(筛选后).pcapng”（共 43 条结果）。

(二) 抓包结果解析

由于浏览器在加载页面的时候建立了多个连接以加快效率，产生的捕获结果较多。不过在下面分析中，我们在握手挥手部分以端口 50332 为例，在数据传输部分分开进行讨论。

TCP 报文的结构主要如图5所示。

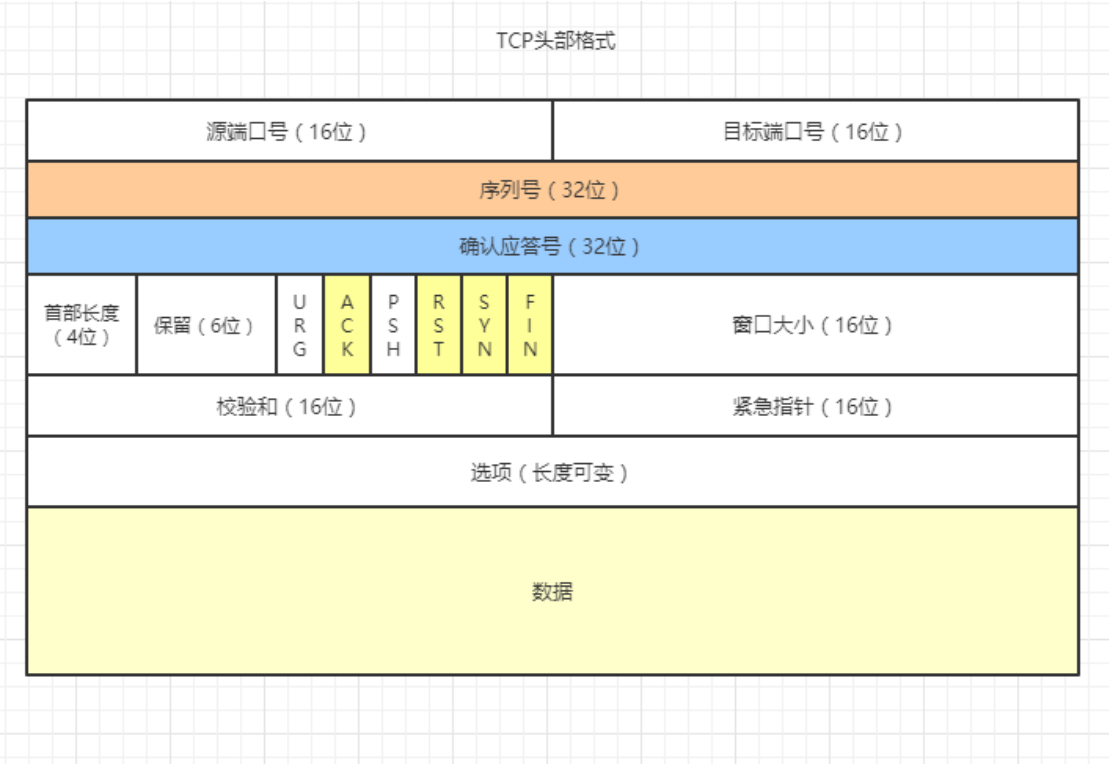

图 5: TCP 报文结构

其中:

- 序列号：在连接建立时由计算机计算出的初始值，通过 SYN 包传给对端主机，每发送一次新的数据包，就累加一次该序列号的大小。
- 确认应答号：下次期望收到的数据的序列号，发送端收到这个确认应答以后可以确认确认应答号-1 的数据包已经被正常接收。
- ACK：该字段为 1 用以指示确认字段中的值是有效的，即该报文段包括一个对已被成功接收的报文段的确认。
- RST：该字段为 1 用以指示收到错误连接后，连接的强制拆除。

- SYN：用以指示连接的建立，该位为 1 的报文表示希望建立连接。
- FIN：用以指示连接的终止，该位为 1 的报文表示希望终止连接。

1. 三次握手

浏览器与服务器建立连接的过程，即“三次握手”如图6所示（以。
三次握手的优点和必要性包括：

- 避免历史连接。
- 同步初始序列号。
- 减少资源开销。

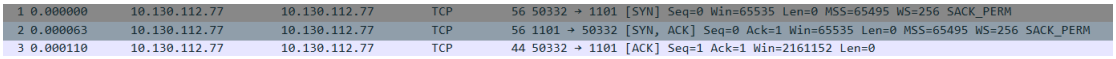

图 6: 三次握手捕获结果

在 Wireshark 中，我们可以看到相对序列号和确认应答号（这表示会话开始时的序列号偏移量）。同时也可以在详细信息中看到绝对序列号和确认应答号。例如，报文 1 的信息如图7所示。他的序列号为 3371096880，相对序列号由于是对这一条的偏移量，所以为 0。

在后续讨论中，均使用相对序列号（对应报文 Seq）和确认应答号（对应报文 Ack）。

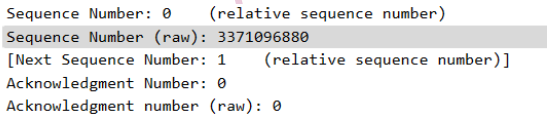

图 7: 报文 1 的序列号

1. 第一次握手

   客户端随机初始化序列号（此时相对序列号为 0），填入序列号字段中，并设置 SYN=1，发送给服务器端，表示发起连接。

2. 第二次握手

   服务器端收到客户端的报文后，同样初始化序列号（此时序列号为 1710365515，相对序列号为 0），填入序列号字段中，并在确认应答号字段中填入收到的序列号 +1（此时确认应答号为 3371096881，相对确认应答号为 1），并设置 SYN=1，ACK=1，发送给客户端。

3. 第三次握手

   客户端收到服务器端的报文后，并在确认应答号字段中填入收到的序列号 +1（此时确认应答号为 1710365516，相对确认应答号为 1），序列号为 3371096881（相对序列号为 1）。此时客户端已经建立连接成功，服务器端收到报文后，同样建立连接成功。

2. 数据传输

在数据传输部分，针对我们设计的网页，客户端先发起了主页请求，接受完成后，依次发起对图片“/logo.jpg”，音频“/intro.mp3”，以及”favicon.ico”图标。

其中图标就是显示在浏览器标签页、书签栏或历史记录旁边的一个小图标，这一请求并不需要网页的显式指定就会自动发起，由于我们并没有对图片进行设置，会返回一个 404。

通过对 http 协议的筛选可以清楚看到请求流程，如图8所示。其中由于多端口的连接音频请求在图片收到之前就已经发送，音频的发送结束报文为 MPEG-1 格式。

4	0.000325	10.130.112.77	10.130.112.77	HTTP	477	GET / HTTP/1.1
9	0.553655	10.130.112.77	10.130.112.77	HTTP	1323	HTTP/1.1 200 OK (text/html)
11	0.570478	10.130.112.77	10.130.112.77	HTTP	421	GET /logo.jpg HTTP/1.1
13	0.571240	10.130.112.77	10.130.112.77	HTTP	384	GET /intro.mp3 HTTP/1.1
15	0.581535	10.130.112.77	10.130.112.77	HTTP	39921	HTTP/1.1 200 OK (JPEG JFIF image)
25	0.582257	10.130.112.77	10.130.112.77	MPEG-1	6252	Audio Layer 3, 320 kb/s, 48 kHz
28	0.606307	10.130.112.77	10.130.112.77	HTTP	424	GET /favicon.ico HTTP/1.1
30	0.607589	10.130.112.77	10.130.112.77	HTTP	5007	HTTP/1.1 404 Not Found (text/html)
32	5.397128	10.130.112.77	10.130.112.77	HTTP	424	GET /favicon.ico HTTP/1.1
34	5.397661	10.130.112.77	10.130.112.77	HTTP	5007	HTTP/1.1 404 Not Found (text/html)

图 8: 请求流程

其中，用户端端口 50332 负责了主页和 logo 的传输，端口 50333 负责了音频和图标的传输。使用“tcp.port == 50332”进行筛选，结果如图9所示。

4	0.000325	10.130.112.77	10.130.112.77	HTTP	477	GET / HTTP/1.1
5	0.000366	10.130.112.77	10.130.112.77	TCP	44	1101 → 50332 [ACK] Seq=1 Ack=434 Win=2160896 Len=0
9	0.553655	10.130.112.77	10.130.112.77	HTTP	1323	HTTP/1.1 200 OK (text/html)
10	0.553716	10.130.112.77	10.130.112.77	TCP	44	50332 → 1101 [ACK] Seq=434 Ack=1280 Win=2159872 Len=0
11	0.570478	10.130.112.77	10.130.112.77	HTTP	421	GET /logo.jpg HTTP/1.1
12	0.570520	10.130.112.77	10.130.112.77	TCP	44	1101 → 50332 [ACK] Seq=1280 Ack=811 Win=2160384 Len=0
15	0.581535	10.130.112.77	10.130.112.77	HTTP	39921	HTTP/1.1 200 OK (JPEG JFIF image)
19	0.581596	10.130.112.77	10.130.112.77	TCP	44	50332 → 1101 [ACK] Seq=811 Ack=41157 Win=2119936 Len=0

图 9: 端口 50332 的传输

除去之前介绍过的 Seq 和 Ack 之外，新增了两个参数 Win 和 Len，在此进行介绍。

Win：窗口大小，表示接收方当前的缓冲区还可以容纳的字节数量。

Len：数据包中的数据长度，单位是字节，表示有效载荷大小，不包括 TCP 头部。

两次传输结构类似，步骤如下：

1. 客户端发送 GET 请求。（主页第 4 条，图片第 11 条）
2. 服务器返回 ACK 报文，确认接收。（主页第 5 条，图片第 12 条）
3. 服务器发送内容，HTTP/1.1 200 OK 响应。（主页第 9 条，图片第 15 条）
4. 客户端发送 ACK 确认。（主页第 10 条，图片第 19 条）

以对主页的传输为例，分析 Seq，Ack，Len 参数如表2所示：

报文编号	Seq	Ack	Len
4	1	1	433
5	1	434	0
9	1	434	1279
10	434	1280	0

表 2: 对主页传输 Seq/Ack/Len 参数

之前服务器端 Seq=0，Ack=1（第二个报文），客户端 Seq=1，Ack=1（第三个报文）。

服务器在报文 5 中 Seq 为 1（由于之前握手占据了 0，更新 Ack 为（之前的 Ack）1+（收到的长度）433=434。

同理，客户端在报文 10 中，由于之前发送的总长度为 434，所以更新了 Seq=434，Ack 也更新为 1+1279=1280。

此外，由于在握手阶段指定 WS=256，每次 win 的减少都需要以 256 为单位向上取整。

端口 50333 负责了 mp3 和图标的传输，其中对图标的传输进行了两次请求，不过因为并未设置该文件，所以并不会成功。

13 0.571240	10.130.112.77	10.130.112.77	HTTP	384 GET /intro.mp3 HTTP/1.1
14 0.571310	10.130.112.77	10.130.112.77	TCP	44 1101 → 50333 [ACK] Seq=1 Ack=341 Win=2160896 Len=0
16 0.581544	10.130.112.77	10.130.112.77	TCP	65539 1101 → 50333 [ACK] Seq=1 Ack=341 Win=2160896 Len=65495 [TCP PDU reassembled in 25]
17 0.581557	10.130.112.77	10.130.112.77	TCP	65539 1101 → 50333 [ACK] Seq=65496 Ack=341 Win=2160896 Len=65495 [TCP PDU reassembled in 25]
18 0.581579	10.130.112.77	10.130.112.77	TCP	429 1101 → 50333 [PSH, ACK] Seq=130991 Ack=341 Win=2160896 Len=385 [TCP PDU reassembled in 25]
20 0.581639	10.130.112.77	10.130.112.77	TCP	44 50333 → 1101 [ACK] Seq=341 Ack=131376 Win=2161152 Len=0
21 0.581975	10.130.112.77	10.130.112.77	TCP	65539 1101 → 50333 [ACK] Seq=131376 Ack=341 Win=2160896 Len=65495 [TCP PDU reassembled in 25]
22 0.581990	10.130.112.77	10.130.112.77	TCP	65539 1101 → 50333 [ACK] Seq=196871 Ack=341 Win=2160896 Len=65495 [TCP PDU reassembled in 25]
23 0.582003	10.130.112.77	10.130.112.77	TCP	126 1101 → 50333 [PSH, ACK] Seq=262366 Ack=341 Win=2160896 Len=82 [TCP PDU reassembled in 25]
24 0.582107	10.130.112.77	10.130.112.77	TCP	44 50333 → 1101 [ACK] Seq=341 Ack=262448 Win=2030080 Len=0
25 0.582257	10.130.112.77	10.130.112.77	MPEG-1	6252 Audio Layer 3, 320 kb/s, 48 kHz
26 0.582282	10.130.112.77	10.130.112.77	TCP	44 50333 → 1101 [ACK] Seq=341 Ack=268656 Win=2023936 Len=0
27 0.582377	10.130.112.77	10.130.112.77	TCP	44 [TCP Window Update] 50333 → 1101 [ACK] Seq=341 Ack=268656 Win=2155008 Len=0
28 0.606307	10.130.112.77	10.130.112.77	HTTP	424 GET /favicon.ico HTTP/1.1
29 0.606344	10.130.112.77	10.130.112.77	TCP	44 1101 → 50333 [ACK] Seq=268656 Ack=721 Win=2160384 Len=0
30 0.607589	10.130.112.77	10.130.112.77	HTTP	5007 HTTP/1.1 404 Not Found (text/html)
31 0.607638	10.130.112.77	10.130.112.77	TCP	44 50333 → 1101 [ACK] Seq=721 Ack=273619 Win=2150144 Len=0
32 5.397128	10.130.112.77	10.130.112.77	HTTP	424 GET /favicon.ico HTTP/1.1
33 5.397169	10.130.112.77	10.130.112.77	TCP	44 1101 → 50333 [ACK] Seq=273619 Ack=1101 Win=2160128 Len=0
34 5.397661	10.130.112.77	10.130.112.77	HTTP	5007 HTTP/1.1 404 Not Found (text/html)
35 5.397701	10.130.112.77	10.130.112.77	TCP	44 50333 → 1101 [ACK] Seq=1101 Ack=278582 Win=2145024 Len=0

图 10: 端口 50333 的传输

传输过程类似，不过因为音频文件过大，需要分段传输。

对应 16-18，20-25 报文，18 和 23 报文的 PSH 表示不等待缓冲区填满先进行传输，对应的客户端接收为 20 和 24，25 报文为对音频包的解码，最终的确认接收 ACK 为 26。

27 的含义是调整缓冲区窗口大小，以便接收更多数据。

3. 四次挥手

四次挥手的流程如图11所示。

36 16.938937	10.130.112.77	10.130.112.77	TCP	44 50332 → 1101 [FIN, ACK] Seq=811 Ack=41157 Win=2119936 Len=0
37 16.938980	10.130.112.77	10.130.112.77	TCP	44 1101 → 50332 [ACK] Seq=41157 Ack=812 Win=2160384 Len=0
40 16.939072	10.130.112.77	10.130.112.77	TCP	44 1101 → 50332 [FIN, ACK] Seq=41157 Ack=812 Win=2160384 Len=0
41 16.939115	10.130.112.77	10.130.112.77	TCP	44 50332 → 1101 [ACK] Seq=812 Ack=41158 Win=2119936 Len=0

图 11: 四次挥手捕获结果

1. 第一次挥手

   客户端向服务器端发送一个 FIN 报文，用来关闭客户端到服务端的数据传送，表示不再向服务器端发数据。

2. 第二次挥手

   服务器端接收后，Ack 修改为收到的 Seq+1，发回一个 ACK 报文。

3. 第三次挥手

   服务器端向客户端发送一个 FIN 报文，用来关闭被动关闭方到主动关闭方的数据传送，即服务器端不再向数据端发数据。

4. 第四次挥手

   客户端收到后，Ack 修改为收到的 Seq+1，发挥一个 ACK 报文。服务器端在接收到这个报文后断开连接，客户端在发送后等待 2MSL，断开连接。

六、 总结

在这次实验中，我通过设计 WEB 网页，搭建 WEB 服务器，通过 Wireshark 抓包的方式对服务器与浏览器之间的通信有了更加深入的理解。

本次实验我遇到的最大困难是在 Wireshark 捕获的过程中，起初并不能筛选出我需要的网页对应的报文。通过反复查找网上资料解决了这一问题。

总的来说，本次实验综合性较强，与理论结合紧密，使我深入理解了 TCP 协议报文传输的流程。