南開大學
Nankai University

计算机学院

计算机网络报告

Web 服务器的搭建与 wireshark 抓包

姓名：岳一名

学号：2212472

专业：计算机科学与技术

2024 年 11 月 2 日

目录

1 实验内容

实验要求我们使用任意系统，任意方式，搭建一个 Web 服务器，之后制作简单的 Web 页面，包括文本，图片和音频，之后通过网页访问编写的 HTML 文档，现实 Web 页面，之后显示过程中通过 Wireshark 进行抓包，通过过滤仅显示 HTTP 的报文，并且详细说明 HTTP 交互过程。

实验流程 下面主要讲解整个实验流程，从 Web 服务器的搭建到网页代码的编写，最后就是抓包的详细过程以及抓包内的解析，下面是具体内容。Web 服务器的搭建

```
import http.server
import socketserver

class CustomHandler(http.server.SimpleHTTPRequestHandler):
    def end_headers(self):
        self.send_header('Connection', 'keep-alive')
        super().end_headers()

PORT = 5600

with socketserver.TCPServer(("", PORT), CustomHandler) as httpd:
    print("Serving at port", PORT)
    httpd.serve_forever()
```

上面就是使用 python 代码进行的 Web 服务器搭建，运行的程序比较简单，但是该服务器的搭建重写了类 SimpleHTTPRequestHandler，SimpleHTTPRequestHandler 这个类就是用来处理 HTTP 的一些简单请求，但是由于 python 版本的问题，如果重写这个类的话，我们只能使用 HTTP1.0，不能进行持久链接，所以在该类重写中我们能够发现其添加了‘keep-alive’这一关键字，能够使得 HTTP1.0 和 HTTP1.1 一样进行持久链接，不需要每次进行传输后就进行一次握手和挥手，大大提升效率。

1.1 网页的搭建

```
<body>
    <div class="container">
        <h1>欢迎来到我的个人主页</h1>
        <h2>个人信息</h2>
        <p>专业：计算机科学与技术</p>
        <p>学号：2212472</p>
        <p>姓名：岳一名</p>
        <img src="logo.jpg" alt="我的LOGO" style="width:100px;height:100px;">
        <h2>自我介绍</h2>
        <audio controls>
            <source src="p5.mp3" type="audio/mpeg">
        </audio>
    </div>
</body>
```

上面是网页 body 部分的代码，为了演示的直观和页面的简洁，我添加了一些 CSS 进行美化，详细请查看源代码，其中的文本信息包含标题，段落。使用 img 标签添加了图片，并对其进行了大小的设置，最后使用 audio 标签插入了音频，音频格式采用 mp3。最后网页的效果大致如下。

图 1.1: 网页效果展示

1.2 wireshark 抓包

在使用 python 进行服务器启动之后，我们打开 Wireshark 进行抓包，进入本地网络监测的选项，同时进入网页。

首先过滤我们使用的端口号 (使用的是 3636)，得到三次握手建立链接的信息。

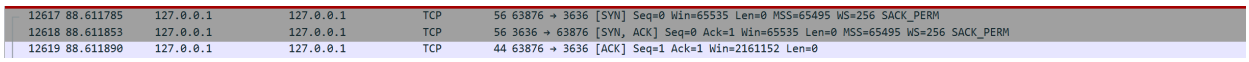

图 1.2: 三次握手

之后就是加入 http 的筛选条件，获取我们 http 的报文

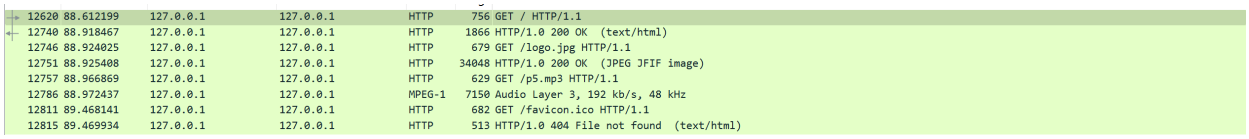

图 1.3: http 报文展示

在这段 HTTP 报文记录中，客户端首先发起了一个 GET 请求，请求根路径 /，成功获得了 200 OK 的响应，返回了 HTML 内容。随后，客户端请求了 logo.jpg 和 p5.mp3 这两个资源，均获得了成功的 200 OK 响应，分别返回了 JPEG 图像和 MP3 音频文件。最后，客户端尝试获取网站图标 favicon.ico，虽然发送了请求，但是没有被接受到，这在很多网页上都有，这是网页用来进行区分的图标，能够给用户比较好的体验，但是这在我们本地是没有的，所以这个没有被访问到是很正常的。

之后就是四次挥手

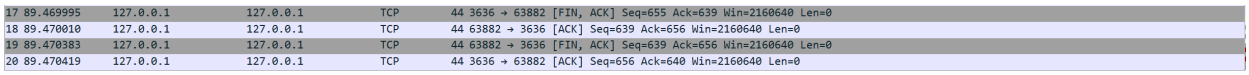

图 1.4: 四次挥手

第一个包是「第一次挥手」：我向服务器发送一个「FIN+ACK」，表示这是一个「释放连接」的请求；第二个包是「第二次挥手」：服务器向我响应一个「ACK」，表示这是一个「确认请求」，我收到后，就会释放我到百度的单向连接；

第三个包是「第三次挥手」：服务器向我发送一个「FIN+ACK」，表示这是一个「释放连接」的请求；

第四个包是「第四次挥手」：我向服务器响应一个「ACK」，表示这是一个「确认请求」，百度收到后，就会释放到我这边的单向连接。