

浙江大学

本科实验报告

课程名称： 计算机网络基础

实验名称： 实现一个轻量级的 WEB 服务器

姓 名： 李文博

学 院： 计算机学院

系： 计算机科学与技术

专 业： 计算机科学与技术

学 号： 3180104939

指导教师： 黄正谦

2021 年 1 月 13 日

浙江大学实验报告

实验名称: 实现一个轻量级的 WEB 服务器 实验类型: 编程实验

同组学生: _____ 实验地点: 计算机网络实验室

一、 实验目的

深入掌握 HTTP 协议规范, 学习如何编写标准的互联网应用服务器。

二、 实验内容

- 服务程序能够正确解析 HTTP 协议, 并传回所需的网页文件和图片文件
- 使用标准的浏览器, 如 IE、Chrome 或者 Safari, 输入服务程序的 URL 后, 能够正常显示服务器上的网页文件和图片
- 服务端程序界面不做要求, 使用命令行或最简单的窗体即可
- 功能要求如下:
 1. 服务程序运行后监听在 80 端口或者指定端口
 2. 接受浏览器的 TCP 连接 (支持多个浏览器同时连接)
 3. 读取浏览器发送的数据, 解析 HTTP 请求头部, 找到感兴趣的部分
 4. 根据 HTTP 头部请求的文件路径, 打开并读取服务器磁盘上的文件, 以 HTTP 响应格式传回浏览器。要求按照文本、图片文件传送不同的 Content-Type, 以便让浏览器能够正常显示。
 5. 分别使用单个纯文本、只包含文字的 HTML 文件、包含文字和图片的 HTML 文件进行测试, 浏览器均能正常显示。
- 本实验可以在前一个 Socket 编程实验的基础上继续, 也可以使用第三方封装好的 TCP 类进行网络数据的收发
- 本实验要求不使用任何封装 HTTP 接口的类库或组件, 也不使用任何服务端脚本程序如 JSP、ASPX、PHP 等

三、 主要仪器设备

联网的 PC 机、Wireshark 软件、Visual Studio、gcc 或 Java 集成开发环境。

四、 操作方法与实验步骤

- 阅读 HTTP 协议相关标准文档, 详细了解 HTTP 协议标准的细节, 有必要的话使用 Wireshark 抓包, 研究浏览器和 WEB 服务器之间的交互过程
- 创建一个文档目录, 与服务器程序运行路径分开
- 准备一个纯文本文件, 命名为 test.txt, 存放在 txt 子目录下
- 准备好一个图片文件, 命名为 logo.jpg, 放在 img 子目录下
- 写一个 HTML 文件, 命名为 test.html, 放在 html 子目录下, 主要内容为:

```

<html>
  <head><title>Test</title></head>
  <body>
    <h1>This is a test</h1>
    
    <form action="dopost" method="POST">
      Login:<input name="login">
      Pass:<input name="pass">
      <input type="submit" value="login">
    </form>
  </body>
</html>

```

- 将 test.html 复制为 noimg.html，并删除其中包含 img 的这一行。
- 服务端编写步骤（**需要采用多线程模式**）
 - a) 运行初始化，打开 Socket，监听在指定端口（**请使用学号的后 4 位作为服务器的监听端口**）
 - b) 主线程是一个循环，主要做的工作是等待客户端连接，如果有客户端连接成功，为该客户端创建处理子线程。该子线程的主要处理步骤是：
 1. 不断读取客户端发送过来的字节，并检查其中是否连续出现了 2 个回车换行符，如果未出现，继续接收；如果出现，按照 HTTP 格式解析第 1 行，分离出方法、文件和路径名，其他头部字段根据需要读取。

✧ 如果解析出来的方法是 GET

2. 根据解析出来的文件和路径名，读取响应的磁盘文件（该路径和服务端程序可能不在同一个目录下，需要转换成绝对路径）。如果文件不存在，第 3 步的响应消息的状态设置为 404，并且跳过第 5 步。
3. 准备好一个足够大的缓冲区，按照 HTTP 响应消息的格式先填入第 1 行（状态码=200），加上回车换行符。然后模仿 Wireshark 抓取的 HTTP 消息，填入必要的几行头部（需要哪些头部，请试验），其中不能缺少的 2 个头部是 Content-Type 和 Content-Length。Content-Type 的值要和文件类型相匹配（请通过抓包确定应该填什么），Content-Length 的值填写文件的字节大小。
4. 在头部行填完后，再填入 2 个回车换行
5. 将文件内容按顺序填入到缓冲区后面部分。

✧ 如果解析出来的方法是 POST

6. 检查解析出来的文件和路径名，如果不是 dopost，则设置响应消息的状态为 404，然后跳到第 9 步。如果是 dopost，则设置响应消息的状态为 200，并继续下一步。
7. 读取 2 个回车换行后面的体部内容（长度根据头部的 Content-Length 字段的指示），并提取出登录名（login）和密码（pass）的值。**如果登录名是你的学号，密码是学号的后 4 位，则将响应消息设置为登录成功，否则将响应消息设置为登录失败。**
8. 将响应消息封装成 html 格式，如

<html><body>响应消息内容</body></html>

9. 准备好一个足够大的缓冲区，按照 HTTP 响应消息的格式先填入第 1 行（根据前面的情况设置好状态码），加上回车换行符。然后填入必要的几行头部，其中不能缺少的 2 个头部是 Content-Type 和 Content-Length。Content-Type 的值设置为 text/html，如果状态码=200，则 Content-Length 的值填写响应消息的字节大小，并将响应消息填入缓冲区的后面部分，否则填写为 0。
 10. 最后一次性将缓冲区内的字节发送给客户端。
 11. 发送完毕后，关闭 socket，退出子线程。
- c) 主线程还负责检测退出指令（如用户按退出键或者收到退出信号），检测到后即通知并等待各子线程退出。最后关闭 Socket，主程序退出。
- 编程结束后，将服务器部署在一台机器上（本机也可以）。在服务器上分别放置纯文本文件（.txt）、只包含文字的测试 HTML 文件（[将测试 HTML 文件中的包含 img 那一行去掉](#)）、包含文字和图片的测试 HTML 文件（以及图片文件）各一个。
 - 确定好各个文件的 URL 地址，然后使用浏览器访问这些 URL 地址，如 <http://x.x.x.x:port/dir/a.html>，其中 port 是服务器的监听端口，dir 是提供给外部访问的路径，请设置为与文件实际存放路径不同，通过服务器内部映射转换。
 - 检查浏览器是否正常显示页面，如果有问题，查找原因，并修改，直至满足要求
 - 使用多个浏览器同时访问这些 URL 地址，检查并发性

五、 实验数据记录和处理

请将以下内容和本实验报告一起打包成一个压缩文件上传：

- 源代码：需要说明编译环境和编译方法，如果不能编译成功，将影响评分
- 可执行文件：可运行的.exe 文件或 Linux 可执行文件

以下实验记录均需结合屏幕截图（截取源代码或运行结果），进行文字标注（看完请删除本句）。

- 服务器的主线程循环关键代码截图（解释总体处理逻辑，省略细节部分）

初始化，获取 socket 句柄，使用 bind 绑定 ip 和端口，调用 listen 设置监听模式。

然后使用 accept 等待客户端连接，连接后创建子进程，急需处理。

```
int main(){
    init();
    while(1){
        struct sockaddr_in client_sockaddr;
        socklen_t socklen=sizeof(client_sockaddr);
        memset(&client_sockaddr, 0, sizeof(client_sockaddr));
        int client_sock=-1;
        while(client_sock==-1){
            cout << "Waiting for new clients....." << endl;
            client_sock=accept(server_sock, (struct sockaddr*)&client_sockaddr, &socklen);
        }
        cout << "Connection to " << inet_ntoa(client_sockaddr.sin_addr) << endl;
        pid_t pid=fork();
        if(pid==0){
            signal(SIGINT, child_sig_handle);
            child(client_sock);
            exit(0);
        }else if(pid>0){
            children.push_back(pid);
            sockets.push_back(client_sock);
        }else error_exit("Cannot create a child process.");
    }
}
```

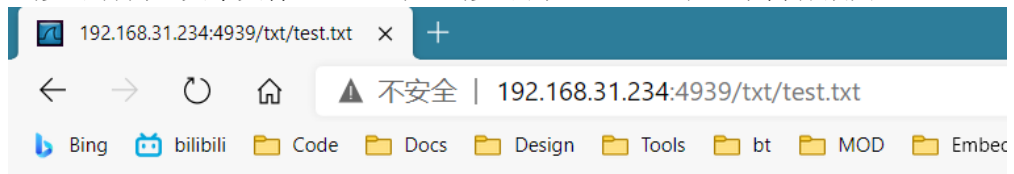
- 服务器的客户端处理子线程关键代码截图（解释总体处理逻辑，省略细节部分）

```
void child(int client_sock){
    static string dat;
    char buf[BUF_SIZE];
    int last_recv=-1;
    int http_head=-1;
    while(1){
        memset(buf, 0, sizeof(buf));
        ssize_t bytes;
        bytes=recv(client_sock, buf, sizeof(buf)-1, 0);
        if(bytes>0){
            last_recv=dat.length();
            buf[bytes]=0;
            dat+=buf;
            if(http_head==-1) http_head=find_http_head(dat, last_recv);
            if(http_head!=-1) try_read_http(client_sock, dat, http_head);
        }
    }
}
```

- 服务器运行后，用 netstat -an 显示服务器的监听端口

```
martinit@ubuntu-martinit:~/zju/cnlab/html/img$ netstat -an | grep 4939
tcp        0      0 192.168.31.234:4939  0.0.0.0:*        LISTEN
martinit@ubuntu-martinit:~/zju/cnlab/html/img$
```

- 浏览器访问纯文本文件（.txt）时，浏览器的 URL 地址和显示内容截图。



This is test.txt

服务器上文件实际存放的路径:

~/zju/cnlab/txt/test.txt

服务器的相关代码片段:

```
fclose(fp);
fseek(fp, 0, SEEK_END);
int flen=ftell(fp);
char *buf=(char*)malloc(sizeof(char)*flen);
rewind(fp);
int readlen=fread(buf, 1, flen, fp);
buf[readlen]=0;
ack.data=buf;
ack.datalen=readlen;
ack.code="200";
ack.code_description="OK";
if(pack.url.substr(0, 3)=="txt") ack.heads.push_back(pair<string, string>("Content-Type", "text/plain"));
else if(pack.url.substr(0, 8)=="html/img") ack.heads.push_back(pair<string, string>("Content-Type", "image/png"));
else if(pack.url.substr(0, 4)=="html") ack.heads.push_back(pair<string, string>("Content-Type", "text/html"));
char tmp[10];
sprintf(tmp, "%d", readlen);
ack.heads.push_back(pair<string, string>("Content-Length", tmp));
fclose(fp);
inst_close= pack.url.substr(0, 3)=="txt" ? 1 : 0;
```

Wireshark 抓取的数据包截图（通过跟踪 TCP 流，只截取 HTTP 协议部分）:

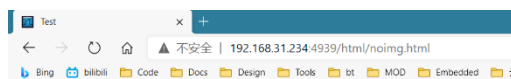
734	26.705826755	192.168.31.234	192.168.31.234	HTTP	554 GET /txt/test.txt HTTP/1.1
735	26.705848075	192.168.31.234	192.168.31.234	TCP	66 4939 → 2243 [ACK] Seq=1 Ack=489 Win
736	26.706004395	192.168.31.234	192.168.31.234	HTTP	179 HTTP/1.1 200 OK (text/plain)
737	26.747475021	192.168.31.234	192.168.31.234	TCP	66 2243 → 4939 [ACK] Seq=489 Ack=114 W
738	26.751894744	192.168.31.234	192.168.31.234	HTTP	488 GET /favicon.ico HTTP/1.1
739	26.751904302	192.168.31.234	192.168.31.234	TCP	66 4939 → 2243 [ACK] Seq=114 Ack=911 W

```
Frame 734: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits) on interface enp7s0, id 0
  Ethernet II, Src: IntelCor_34:98:9c (14:4f:8a:34:98:9c), Dst: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9)
  Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.234
  Transmission Control Protocol, Src Port: 2243, Dst Port: 4939, Seq: 1, Ack: 1, Len: 488
  Hypertext Transfer Protocol
    GET /txt/test.txt HTTP/1.1\r\n
    Host: 192.168.31.234:4939\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/si
    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
    \r\n
    [Full request URI: http://192.168.31.234:4939/txt/test.txt]
    [HTTP request 1/2]
    [Response in frame: 736]
    [Next request in frame: 738]
```

734	26.705826755	192.168.31.243	192.168.31.234	HTTP	554 GET /txt/test.txt HTTP/1.1
735	26.705848075	192.168.31.234	192.168.31.243	TCP	66 4939 → 2243 [ACK] Seq=1 Ack=489 Win=0 Len=0
736	26.706004395	192.168.31.234	192.168.31.243	HTTP	179 HTTP/1.1 200 OK (text/plain)
737	26.747475021	192.168.31.243	192.168.31.234	TCP	66 2243 → 4939 [ACK] Seq=489 Ack=114 Len=0
738	26.751894744	192.168.31.243	192.168.31.234	HTTP	488 GET /favicon.ico HTTP/1.1
739	26.751904302	192.168.31.234	192.168.31.243	TCP	66 4939 → 2243 [ACK] Seq=114 Ack=911 Len=0

<p>Frame 736: 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits) on interface enp7s0, id 0</p> <p>Ethernet II, Src: ASUSTekC_d0:a9 (24:4b:fe:cd:c0:a9), Dst: IntelCor_34:98:9c (14:4f:8a:34:98:9c)</p> <p>Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.243</p> <p>Transmission Control Protocol, Src Port: 4939, Dst Port: 2243, Seq: 1, Ack: 489, Len: 113</p> <p>Hypertext Transfer Protocol</p> <p>HTTP/1.1 200 OK\r\n</p> <p>Server: Martinit's http server\r\n</p> <p>Content-Type: text/plain\r\n</p> <p>Content-Length: 16\r\n</p> <p>[Content length: 16]</p> <p>\r\n</p> <p>[HTTP response 1/2]</p> <p>[Time since request: 0.000177640 seconds]</p> <p>[Request in frame: 734]</p> <p>[Next request in frame: 738]</p> <p>[Request URI: http://192.168.31.234:4939/txt/test.txt]</p> <p>File Data: 16 bytes</p> <p>Line-based text data: text/plain (1 lines)</p>	<pre> 0000 14 4f 8a 34 98 9c 24 4b fe cd c0 a9 08 00 45 00 .0.4.\$K 0010 00 a5 cb fa 40 00 40 06 ad 2a c0 a8 1f ea c0 a8 ...@... 0020 1f f3 13 4b 08 c3 1b 88 a4 96 d5 7d 52 22 00 18 ...K... 0030 01 fa c1 c5 00 00 01 01 08 0a 36 27 0b 10 b5 0040 1e 02 48 54 54 50 2f 31 2e 31 20 32 30 20 4f ..HTTP/1.1 200 0050 4b 0d 0a 53 65 72 76 65 72 3a 20 4d 61 72 74 69 K...Serve r... 0060 6e 69 74 27 73 20 68 74 74 70 20 73 65 72 76 65 nit's ht tp 0070 72 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a r...Conte nt- 0080 20 74 65 78 74 2f 70 6c 61 69 6e 0d 0a 43 6f 6e text/pl 0090 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 31 36 0d tent-Len gth: 00a0 0a 0d 0a 54 68 69 73 20 69 73 20 74 65 73 74 2e ...This is 00b0 74 78 74 +... </pre>
---	---

- 浏览器访问只包含文本的 HTML 文件时，浏览器的 URL 地址和显示内容截图。



This is a test

Login: Pass: login

服务器文件实际存放的路径：

~/zju/cnlab/html/noimg.html

Wireshark 抓取的数据包截图（只截取 HTTP 协议部分，包括 HTML 内容）：

788	80.289140507	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=1 Ack=492 Win=0 Len=0
789	80.289346910	192.168.31.234	192.168.31.243	HTTP	454 HTTP/1.1 200 OK (text/html)
790	80.295664799	192.168.31.243	192.168.31.234	TCP	66 2243 → 4939 [FIN, ACK] Seq=911 Ack=912 Len=0
791	80.337525271	192.168.31.243	192.168.31.234	TCP	66 2242 → 4939 [ACK] Seq=492 Ack=389 Len=0
792	80.339383287	192.168.31.234	192.168.31.243	TCP	66 4939 → 2243 [ACK] Seq=114 Ack=912 Len=0
793	80.346397709	192.168.31.243	192.168.31.234	HTTP	491 GET /favicon.ico HTTP/1.1
794	80.346416454	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=389 Ack=917 Len=0
795	80.346839048	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=389 Ack=917 Len=0
796	80.346847353	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=3285 Ack=917 Len=0

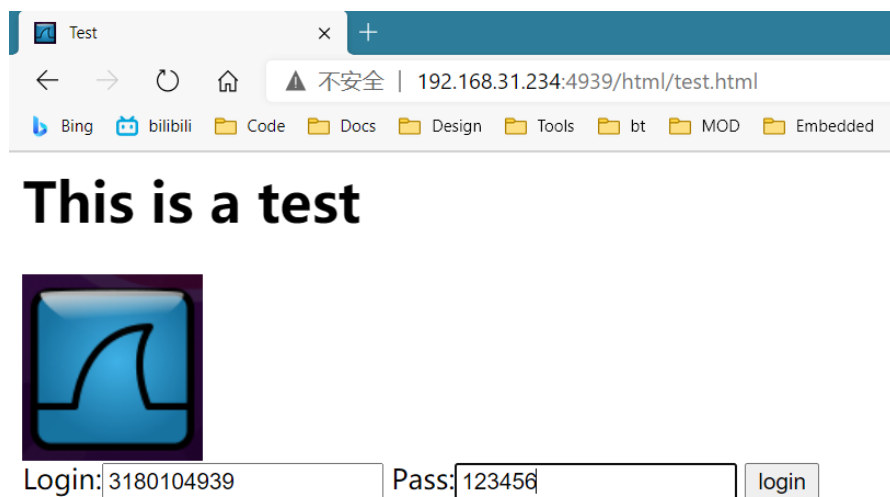
<p>Frame 787: 557 bytes on wire (4456 bits), 557 bytes captured (4456 bits) on interface enp7s0, id 0</p> <p>Ethernet II, Src: IntelCor_34:98:9c (14:4f:8a:34:98:9c), Dst: ASUSTekC_d0:a9 (24:4b:fe:cd:c0:a9)</p> <p>Internet Protocol Version 4, Src: 192.168.31.243, Dst: 192.168.31.234</p> <p>Transmission Control Protocol, Src Port: 2242, Dst Port: 4939, Seq: 1, Ack: 1, Len: 491</p> <p>Hypertext Transfer Protocol</p> <p>GET /html/noimg.html HTTP/1.1\r\n</p> <p>Host: 192.168.31.234:4939\r\n</p> <p>Connection: keep-alive\r\n</p> <p>Upgrade-Insecure-Requests: 1\r\n</p> <p>User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4382.108 Safari/537.36\r\n</p> <p>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n</p> <p>Accept-Encoding: gzip, deflate\r\n</p> <p>Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6\r\n</p> <p>\r\n</p> <p>[Full request URI: http://192.168.31.234:4939/html/noimg.html]</p> <p>[HTTP request 1/2]</p> <p>[Response in frame: 789]</p> <p>[Next request in frame: 793]</p>	<pre> 0000 24 4b fe cd c0 a9 14 4f 8a 34 98 9c 08 00 45 00 \$K....0 0010 02 1f 14 b4 40 00 80 06 22 f7 c0 a8 1f f3 c0 a8 ...@... 0020 1f ea 08 c2 13 4b ea 6a c5 f7 0f 33 f8 85 80 18 ...K.j 0030 02 02 e5 a8 00 00 01 01 08 0a 10 b5 ef 51 36 28 0040 1b 53 47 45 54 20 2f 68 74 6d 6c 2f 6e 6f 69 6d .SGET /h tml/ 0050 67 2e 68 74 6d 6c 20 48 54 54 50 2f 31 2e 31 0d g.html H TTP/ 0060 0a 48 6f 73 74 3a 20 31 39 32 2e 31 36 38 2e 33 .Host: 1 0070 31 2e 32 33 34 3a 34 39 33 39 0d 0a 43 6f 6e 6e 1.234:49 0080 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 ection: keep- 0090 76 65 0d 0a 55 70 67 72 61 64 65 2d 49 6e 73 65 ve..Upgr ade- 00a0 63 75 72 65 2d 52 65 71 75 65 73 74 73 3a 20 31 cure-Req uests: 00b0 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f ..User-A gents: 00c0 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f zilla/5. 0 </pre>
---	--

788	80.289140507	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=1 Ack=492
789	80.289346010	192.168.31.234	192.168.31.243	HTTP	454 HTTP/1.1 200 OK (text/html)
790	80.295664799	192.168.31.243	192.168.31.234	TCP	66 2243 → 4939 [FIN, ACK] Seq=911
791	80.337525271	192.168.31.243	192.168.31.234	TCP	66 2242 → 4939 [ACK] Seq=492 Ack=3
792	80.339383287	192.168.31.234	192.168.31.243	TCP	66 4939 → 2243 [ACK] Seq=114 Ack=9
793	80.346397709	192.168.31.243	192.168.31.234	HTTP	491 GET /favicon.ico HTTP/1.1
794	80.346416454	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=389 Ack=9
795	80.346839048	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=389
796	80.346847353	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=3285
797	80.346876107	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=3285

Frame 789: 454 bytes on wire (3632 bits), 454 bytes captured (3632 bits) on interface enp7s0, id 0
Ethernet II, Src: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9), Dst: IntelCor_34:98:9c (14:4f:8a:34:98:9c)
Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.243
Transmission Control Protocol, Src Port: 4939, Dst Port: 2242, Seq: 1, Ack: 492, Len: 388
Hypertext Transfer Protocol
HTTP/1.1 200 OK\r\n
Server: Martin's http server\r\n
Content-Type: text/html\r\n
Content-Length: 291\r\n
[Content length: 291]
\r\n
[HTTP response 1/2]
[Time since request: 0.000235707 seconds]
[Request in frame: 787]
[Next request in frame: 793]
[Next response in frame: 801]
[Request URI: http://192.168.31.234:4939/html/noimg.html]
File Data: 291 bytes
Line-based text data: text/html (11 lines)

0060	6e 69 74 27 73 20 68 74 74 70 20 73 65 72 76 65	nit's ht tp
0070	72 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a	r..Conte nt-
0080	20 74 65 78 74 2f 68 74 6d 6c 0d 0a 43 6f 6e 74	text/ht
0090	65 6e 74 2d 4c 65 6e 67 74 68 3a 20 32 39 31 0d	ent-Leng th:
00a0	0a 0d 0a 3c 68 74 6d 6c 3e 0a 20 20 20 20 3c 68	...<html>
00b0	65 61 64 3e 3c 74 69 74 6c 65 3e 54 65 73 74 3c	ead><tit
00c0	2f 74 69 74 6c 65 3e 3c 2f 68 65 61 64 3e 0a 20	/title>< /head>
00d0	20 20 20 3c 62 6f 64 79 3e 0a 20 20 20 20 20 20	<body>
00e0	20 20 3c 68 31 3e 54 68 69 73 20 69 73 20 61 20	<h1>Th is is a
00f0	74 65 73 74 3c 2f 68 31 3e 0a 20 20 20 20 20 20	test</h1>
0100	20 20 3c 66 6f 72 6d 20 61 63 74 69 6f 6e 3d 22	<form
0110	64 6f 70 6f 73 74 22 20 6d 65 74 68 6f 64 3d 22	dopost"
0120	50 4f 53 54 22 3e 0a 20 20 20 20 20 20 20 20 20	POST">
0130	20 20 20 4c 6f 67 69 6e 3a 3c 69 6e 70 75 74 20	Login :<input
0140	6e 64 65 2d 73 6e 6f 67 60 6a 72 2a 0a 7a 20	name="lo gin">

- 浏览器访问包含文本、图片的 HTML 文件时，浏览器的 URL 地址和显示内容截图。



服务器上文件实际存放的路径：

~/zju/cnlab/html/test.html

Wireshark 抓取的数据包截图（只截取 HTTP 协议部分，包括 HTML、图片文件的部分内容）：

833	154.562338333	192.168.31.243	192.168.31.234	TCP	66	22/2 → 4939	[ACK] Seq=1 Ack=1 Win=1315
834	154.565825278	192.168.31.243	192.168.31.234	HTTP	556	GET /html/test.html	HTTP/1.1
835	154.565843021	192.168.31.234	192.168.31.243	TCP	66	4939 → 2242	[ACK] Seq=17446 Ack=1407 W
836	154.565932326	192.168.31.234	192.168.31.243	HTTP	487	HTTP/1.1 200 OK	(text/html)
837	154.601996480	192.168.31.243	192.168.31.234	HTTP	496	GET /html/img/logo.png	HTTP/1.1
838	154.602006218	192.168.31.234	192.168.31.243	TCP	66	4939 → 2242	[ACK] Seq=17867 Ack=1837 W
839	154.602237057	192.168.31.234	192.168.31.243	TCP	2962	4939 → 2242	[PSH, ACK] Seq=17867 Ack=1
840	154.602244080	192.168.31.234	192.168.31.243	TCP	2962	4939 → 2242	[PSH, ACK] Seq=20763 Ack=1
841	154.602308259	192.168.31.234	192.168.31.243	TCP	2962	4939 → 2242	[PSH, ACK] Seq=23659 Ack=1
842	154.602316895	192.168.31.234	192.168.31.243	HTTP	1727	HTTP/1.1 200 OK	(PNG)
Frame 834: 556 bytes on wire (4448 bits), 556 bytes captured (4448 bits) on interface enp7s0, id 0							
Ethernet II, Src: IntelCor_34:98:9c (14:4f:8a:34:98:9c), Dst: ASUSTekC_d:c0:a9 (24:4b:fe:cd:c0:a9)							
Internet Protocol Version 4, Src: 192.168.31.243, Dst: 192.168.31.234							
Transmission Control Protocol, Src Port: 2242, Dst Port: 4939, Seq: 917, Ack: 17446, Len: 490							
Hypertext Transfer Protocol							
GET /html/test.html HTTP/1.1\r\n							
Host: 192.168.31.234:4939\r\n							
Connection: keep-alive\r\n							
Upgrade-Insecure-Requests: 1\r\n							
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.14							
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed							
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							
\r\n							
[Full request URI: http://192.168.31.234:4939/html/test.html]							
[HTTP request 3/4]							
[Prev request in frame: 793]							
[Response in frame: 836]							
[Next request in frame: 837]							
0040	ec 47 45 54 20 2f 68 74 6d 6c 2f 74 65 73 74	..GET /h tml/					
0050	2e 68 74 6d 6c 20 48 54 54 50 2f 31 2e 31 0d 0a	.html HT TP/					
0060	48 6f 73 74 3a 20 31 39 32 2e 31 36 38 2e 33 31	Host: 19					
0070	2e 32 33 34 3a 34 39 33 39 0d 0a 43 6f 6e 6e 65	.234:493					
0080	63 74 69 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76	ction: k eep-					
0090	65 0d 0a 55 70 67 72 61 64 65 2d 49 6e 73 65 63	e -Upgra de-					
00a0	75 72 65 2d 52 65 71 75 65 73 74 73 3a 20 31 0d	ure-Requ ests:					
00b0	0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a	-User-Ag ent:					
00c0	69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77	illa/5.0					
00d0	73 20 4e 54 20 31 30 2e 30 3b 20 57 69 6e 36 34	s NT 10. 0;					
00e0	3b 20 78 36 34 29 20 41 70 70 6c 65 57 65 62 4b	; x64) A					
00f0	69 74 2f 35 33 37 2e 33 36 20 28 4b 48 54 4d 4c	it/537.3 6					
0100	2c 20 6c 69 6b 65 20 47 65 63 6b 6f 29 20 43 68	, like G ecko)					
0110	72 6f 6d 65 2f 38 37 2e 30 2e 34 32 38 30 2e 31	rome/87.					
0120	34 31 20 53 61 66 61 72 69 2f 35 33 37 2e 33 36	41 Safar i/					
Frame 836: 487 bytes on wire (3896 bits), 487 bytes captured (3896 bits) on interface enp7s0, id 0							
Ethernet II, Src: ASUSTekC_d:c0:a9 (24:4b:fe:cd:c0:a9), Dst: IntelCor_34:98:9c (14:4f:8a:34:98:9c)							
Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.243							
Transmission Control Protocol, Src Port: 4939, Dst Port: 2242, Seq: 17446, Ack: 1407, Len: 421							
Hypertext Transfer Protocol							
HTTP/1.1 200 OK\r\n							
Server: Martin's http server\r\n							
Content-Type: text/html\r\n							
Content-Length: 324\r\n							
[Content length: 324]							
\r\n							
[HTTP response 3/4]							
[Time since request: 0.000197048 seconds]							
[Prev request in frame: 793]							
[Prev response in frame: 801]							
[Request in frame: 834]							
[Next request in frame: 837]							
[Next response in frame: 842]							
[Request URI: http://192.168.31.234:4939/html/test.html]							
0000	14 4f 8a 34 98 9c 24 4b fe cd c0 a9 08 00 45 00	0 4..\$K					
0010	01 d9 2d ed 40 00 40 06 4a 04 c0 a8 1f ea c0 a8	...0 0.					
0020	1f f3 13 4b 08 c2 0f 34 3c aa ea 6a cb 75 80 18	...K...4					
0030	01 f5 c2 f9 00 00 01 01 08 0a 36 29 5e fe 10 b7					
0040	11 76 48 54 54 50 2f 31 2e 31 20 32 30 30 20 4f	vHTTP/1 .1 200					
0050	4d 0d 0a 53 65 72 76 65 72 3a 20 4d 61 72 74 69	K -Serve r:					
0060	6e 69 74 27 73 20 68 74 74 70 20 73 65 72 76 65	nit's ht tp					
0070	72 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a	r -Conte nt-					
0080	20 74 65 78 74 2f 68 74 6d 6c 0d 0a 43 6f 6e 74	text/ht					
0090	65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 32 34 0d	ent-Leng th:					
00a0	0a 0d 0a 3c 68 74 6d 6c 3e 0a 20 20 20 20 3c 68	...<html >					
00b0	65 61 64 3e 3c 74 69 74 6c 65 3e 54 65 73 74 3c	ead><tit					
00c0	2f 74 69 74 6c 65 3e 3c 2f 68 65 61 64 3e 0a 20	/title>< /head>					
00d0	20 20 20 3c 62 6f 64 79 3e 0a 20 20 20 20 20 20	<body >					
00e0	20 20 3c 68 31 3e 54 68 69 73 20 69 73 20 61 20	<h1>Th is is a					

835	154.565843021	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=17446 Ac
836	154.565932326	192.168.31.234	192.168.31.243	HTTP	487 HTTP/1.1 200 OK (text/html)
837	154.601996480	192.168.31.243	192.168.31.234	HTTP	496 GET /html/img/logo.png HTTP/1.
838	154.602006218	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=17867 Ac
839	154.602237057	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=178
840	154.602244080	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=207
841	154.602308259	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=236
842	154.602316895	192.168.31.234	192.168.31.243	HTTP	1727 HTTP/1.1 200 OK (PNG)
843	154.602320004	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=1837 Ac

Frame 837: 496 bytes on wire (3968 bits), 496 bytes captured (3968 bits) on interface enp7s0, id 0
 Ethernet II, Src: IntelCor_34:98:9c (14:4f:8a:34:98:9c), Dst: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9)
 Internet Protocol Version 4, Src: 192.168.31.243, Dst: 192.168.31.234
 Transmission Control Protocol, Src Port: 2242, Dst Port: 4939, Seq: 1407, Ack: 17867, Len: 430

Hypertext Transfer Protocol

GET /html/img/logo.png HTTP/1.1\r\n

Host: 192.168.31.234:4939\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/87.

Accept: image/webp,image/apng,image/*,*/*;q=0.8\r\n

Referer: http://192.168.31.234:4939/html/test.html\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

\r\n

[Full request URI: http://192.168.31.234:4939/html/img/logo.png]

[HTTP request 4/4]

[Prev request in frame: 834]

[Response in frame: 842]

```

040 5e fe 47 45 54 20 2f 68 74 6d 6c 2f 69 6d 67 2f ^..GET /h tml/
050 6c 6f 67 6f 2e 70 6e 67 20 48 54 54 50 2f 31 2e logo.png HTTP/
060 31 0d 0a 48 6f 73 74 3a 20 31 39 32 2e 31 36 38 1..Host:
070 2e 33 31 2e 32 33 34 3a 34 39 33 39 0d 0a 43 6f .31.234:
080 6e 6e 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d 61 nnection : keep-
090 6c 69 76 65 0d 0a 55 73 65 72 2d 41 67 65 6e 74 live..Us er-
0a0 3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 : Mozill a/5.0
0b0 69 6e 64 6f 77 73 20 4e 54 20 31 30 2e 30 3b 20 indows N T 10.0;
0c0 57 69 6e 36 34 3b 20 78 36 34 29 20 41 70 70 6c Win64; x 64)
0d0 65 57 65 62 4b 69 74 2f 35 33 37 2e 33 36 20 28 eWebKit/ 537.36
0e0 4b 48 54 4d 4c 2c 20 6c 69 6b 65 20 47 65 63 6b KHTML, l ike
0f0 6f 29 20 43 68 72 6f 6d 65 2f 38 37 2e 30 2e 34 o) Chrom e/
100 32 38 30 2e 31 34 31 20 53 61 66 61 72 69 2f 35 280.141 Safari/
110 33 37 2e 33 36 20 45 64 67 2f 38 37 2e 30 2e 36 37.36 Fd n/

```

841	154.602308259	192.168.31.234	192.168.31.243	TCP	2962 4939 → 2242 [PSH, ACK] Seq=
842	154.602316895	192.168.31.234	192.168.31.243	HTTP	1727 HTTP/1.1 200 OK (PNG)
843	154.602320004	192.168.31.234	192.168.31.243	TCP	66 4939 → 2242 [ACK] Seq=1837

Frame 842: 1727 bytes on wire (13816 bits), 1727 bytes captured (13816 bits) on interface enp7s0, id 0
 Ethernet II, Src: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9), Dst: IntelCor_34:98:9c (14:4f:8a:34:98:9c)
 Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.243
 Transmission Control Protocol, Src Port: 4939, Dst Port: 2242, Seq: 26555, Ack: 1837, Len: 1661
 [4 Reassembled TCP Segments (10349 bytes): #839(2896), #840(2896), #841(2896), #842(1661)]

Hypertext Transfer Protocol

HTTP/1.1 200 OK\r\n

Server: Martin's http server\r\n

Content-Type: image/png\r\n

Content-Length: 10250\r\n

[Content length: 10250]

\r\n

[HTTP response 4/4]

[Time since request: 0.000320415 seconds]

[Prev request in frame: 834]

[Prev response in frame: 836]

[Request in frame: 837]

[Request URI: http://192.168.31.234:4939/html/img/logo.png]

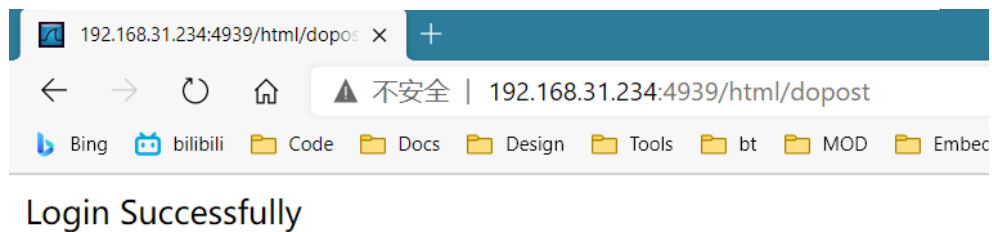
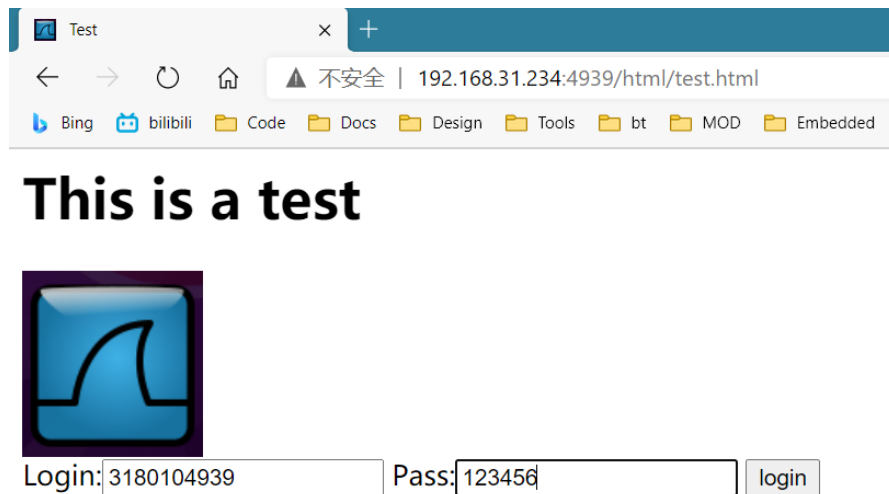
File Data: 10250 bytes

```

0000 14 4f 8a 34 98 9c 24 4b fe cd c0 a9 08 00 45 00 0..4..$K
0010 06 b1 2d f5 40 00 40 06 45 24 c0 a8 1f ea c0 a8 ...0..@
0020 1f f3 13 4b 08 c2 0f 34 60 3f ea 6a cd 23 80 18 ...K...4 `?
0030 01 f5 c7 d1 00 00 01 01 08 0a 36 29 5f 23 10 b7 .....
0040 11 9a f2 b6 30 91 57 40 53 7f c4 12 10 02 53 db ...0..W0
0050 c2 9a 55 eb 93 6a 5b e6 4f be b2 64 30 ac 41 07 ...U..j[
0060 50 d3 03 8c ab 4d 64 56 cb c5 5f d5 7a 22 35 83 P...Mdv
0070 5a 7d f1 6a 6a e8 38 c7 39 1c 70 80 38 06 e4 c8 Z}..jj-8
0080 4d df b7 21 79 82 c9 4b 9c 7f 08 40 fd 92 da bb M..!y..K
0090 fc ee 86 69 af 04 67 fb 61 bf 58 df 44 85 af 2a ...i..g.
00a0 00 b2 e0 a0 0d 8a 0f 38 ae bd eb 0b 48 62 fe 94 .....8
00b0 51 c9 ea 5f 7b 7c 3e f3 3e 4e 16 ca b3 a8 64 97 Q.._{|>
00c0 cd af 33 0c 0c 17 4c f8 ca b2 4d b5 5b 9d dd d3 ...3...L

```

- 浏览器输入正确的登录名或密码，点击登录按钮（login）后的显示截图。



服务器相关处理代码片段：

```
}else if(pack.req_type==REQ_TYPE_POST){
    if(pack.url=="html/dopost"){
        ack.code="200";
        ack.code_description="OK";
        ack.heads.push_back(pair<string, string>("Content-Type", "text/html"));
        if(check_login(pack.data)) ack.data=(char *)"<html><body>Login Successfully</body></html>";
        else ack.data=(char *)"<html><body>Login Failed</body></html>";
        ack.datalen=strlen(ack.data);
        char tmp[10];
        sprintf(tmp, "%d", ack.datalen);
        ack.heads.push_back(pair<string, string>("Content-Length", tmp));
        inst_close=1;
    }else{
        ack.code="404";
        ack.code_description="Resource not found";
        ack.heads.clear();
        ack.data=NULL;
        ack.datalen=0;
        inst_close=1;
    }
}
```

Wireshark 抓取的数据包截图（HTTP 协议部分）

865	239.484986983	192.168.31.243	192.168.31.234	TCP	66	2281 → 4939 [ACK] Seq=1 Ack=1 Win=13158
866	239.488639157	192.168.31.243	192.168.31.234	HTTP	765	POST /html/dopost HTTP/1.1 (applicatio
867	239.488653835	192.168.31.234	192.168.31.243	TCP	66	4939 → 2242 [ACK] Seq=28216 Ack=2536 Wi
868	239.488713666	192.168.31.234	192.168.31.243	HTTP	206	HTTP/1.1 200 OK (text/html)
869	239.523385878	192.168.31.234	192.168.31.243	TCP	66	4939 → 2272 [ACK] Seq=1 Ack=2 Win=65280
870	239.531024920	192.168.31.243	192.168.31.234	TCP	66	2242 → 4939 [ACK] Seq=2536 Ack=28356 Wi
871	260.369313331	192.168.31.243	192.168.31.234	TCP	60	[TCP Keep-Alive] 2243 → 4939 [ACK] Seq=
872	260.369328930	192.168.31.234	192.168.31.243	TCP	78	[TCP Keep-Alive ACK] 4939 → 2243 [ACK] :

Frame 866: 765 bytes on wire (6120 bits), 765 bytes captured (6120 bits) on interface enp7s0, id 0
Ethernet II, Src: IntelCor_34:98:9c (14:4f:8a:34:98:9c), Dst: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9)
Internet Protocol Version 4, Src: 192.168.31.243, Dst: 192.168.31.234
Transmission Control Protocol, Src Port: 2242, Dst Port: 4939, Seq: 1837, Ack: 28216, Len: 699

Hypertext Transfer Protocol

POST /html/dopost HTTP/1.1\r\n

Host: 192.168.31.234:4939\r\n

Connection: keep-alive\r\n

Content-Length: 28\r\n

[Content length: 28]

Cache-Control: max-age=0\r\n

Upgrade-Insecure-Requests: 1\r\n

Origin: http://192.168.31.234:4939\r\n

Content-Type: application/x-www-form-urlencoded\r\n

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.143

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-

Referer: http://192.168.31.234:4939/html/test.html\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

2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 73 69 67 , applica tion/

6e 65 64 2d 65 78 63 68 61 6e 67 65 3b 76 3d 62 ned-exch

33 3b 71 3d 30 2e 39 0d 0a 52 65 66 65 72 65 72 3;q=0.9

2550 3a 30 68 74 74 70 3a 2f 2f 31 39 32 2e 31 36 38 : http:/ /

2e 3a 31 2e 32 33 34 3a 34 39 33 39 2f 68 74 6d ;31.234: 4939/

2770 6c 2f 74 65 73 74 2e 68 74 6d 6c 0d 0a 41 63 63 l/test.h

2880 65 70 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a ept-Enco ding:

2990 69 70 2c 20 64 65 66 6c 61 74 65 0d 0a 41 63 63 ip, defl

2a00 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a 20 7a 68 ept-Lang uage:

2b00 2d 43 4e 2c 7a 68 3b 71 3d 30 2e 39 2c 65 6e 3b -CN,zh;q

2c00 71 3d 30 2e 38 2c 65 6e 2d 47 42 3b 71 3d 30 2e q=0.8,en -

2d00 37 2c 65 6e 2d 55 53 3b 71 3d 30 2e 36 0d 0a 0d 7,en-US;

2e00 0a 6c 6f 67 69 6e 3d 33 31 38 30 31 30 34 39 33 .login=3

2f00 39 26 70 61 73 73 3d 31 32 33 34 35 36 9&pass=1 23456

867	239.488653835	192.168.31.234	192.168.31.243	TCP	66	4939 → 2242 [ACK] Seq=28216 Ack=25
868	239.488713666	192.168.31.234	192.168.31.243	HTTP	206	HTTP/1.1 200 OK (text/html)
869	239.523385878	192.168.31.234	192.168.31.243	TCP	66	4939 → 2272 [ACK] Seq=1 Ack=2 Win=
870	239.531024920	192.168.31.243	192.168.31.234	TCP	66	2242 → 4939 [ACK] Seq=2536 Ack=283

Frame 868: 206 bytes on wire (1648 bits), 206 bytes captured (1648 bits) on interface enp7s0, id 0
Ethernet II, Src: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9), Dst: IntelCor_34:98:9c (14:4f:8a:34:98:9c)
Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.243
Transmission Control Protocol, Src Port: 4939, Dst Port: 2242, Seq: 28216, Ack: 2536, Len: 140

Hypertext Transfer Protocol

HTTP/1.1 200 OK\r\n

Server: Martin's http server\r\n

Content-Type: text/html\r\n

Content-Length: 44\r\n

[Content length: 44]

\r\n

[HTTP response 5/5]

[Time since request: 0.000074509 seconds]

[Prev request in frame: 837]

[Prev response in frame: 842]

[Request in frame: 866]

[Request URI: http://192.168.31.234:4939/html/dopost]

File Data: 44 bytes

Line-based text data: text/html (1 lines)

000 14 4f 8a 34 98 9c 24 4b fe cd c0 a9 08 00 45 00 .0.4.SK

010 00 c0 2d f9 40 00 00 06 4b 11 c0 a8 1f ea c0 a8 ...@@

020 1f f3 13 4b 08 c2 0f 34 66 bc ea 6a cf de 80 18 ...K...4

030 01 f5 c1 e0 00 00 01 01 08 0a 36 2a aa b9 10 b8 ...

040 5d 31 48 54 54 50 2f 31 2e 31 20 32 30 30 20 4f]!HTTP/1 .1 200

050 4b 0d 0a 53 65 72 76 65 72 3a 20 4d 61 72 74 69 K- Serve r:

060 6e 69 74 27 73 20 68 74 74 70 20 73 65 72 76 65 nit's ht tp

070 72 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a r- Conte nt-

080 20 74 65 78 74 2f 68 74 6d 6c 0d 0a 43 6f 6e 74 text/ht

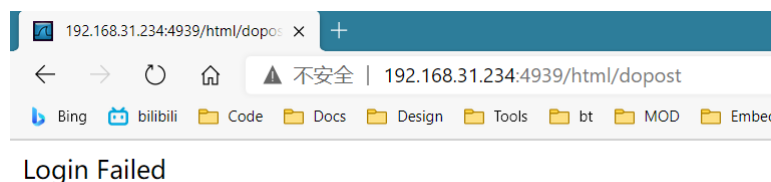
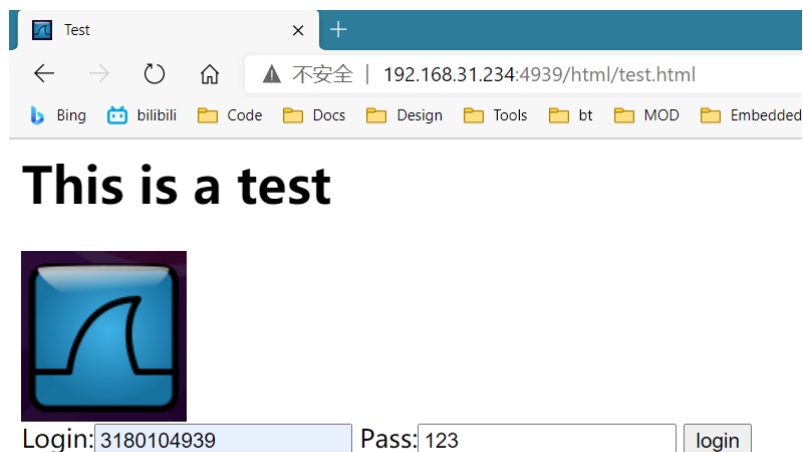
090 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 34 34 0d 0a ent-Leng th:

0a0 0d 0a 3c 68 74 6d 6c 3e 3c 62 6f 64 79 3e 4c 6f <<html>

0b0 67 69 6e 20 53 75 63 63 65 73 67 65 76 6c 6c 79 gin Succ

0c0 3c 2f 62 6f 64 79 3e 3c 2f 68 74 6d 6c 3e </body>< /html>

- 浏览器输入错误的登录名或密码，点击登录按钮（login）后的显示截图。



Wireshark 抓取的数据包截图（HTTP 协议部分）

961	329.541438539	192.168.31.234	192.168.31.243	TCP	78 [TCP Keep-Alive ACK] 4!
962	340.664436640	192.168.31.243	192.168.31.234	HTTP	762 POST /html/dopost HTTP
963	340.664452830	192.168.31.234	192.168.31.243	TCP	66 4939 → 12023 [ACK] Seq:
964	340.664571781	192.168.31.234	192.168.31.243	HTTP	200 HTTP/1.1 200 OK (text.
965	340.723188611	192.168.31.243	192.168.31.234	TCP	66 12023 → 4939 [ACK] Seq:
979	350.385592688	192.168.31.243	192.168.31.234	TCP	60 [TCP Keep-Alive] 2243 .
980	350.385600994	192.168.31.234	192.168.31.243	TCP	78 [TCP Keep-Alive ACK] 4!
982	357.074222345	192.168.31.243	192.168.31.234	TCP	60 [TCP Keep-Alive] 2243 .
▶ Ethernet II, Src: IntelCor_34:98:9c (14:4f:8a:34:98:9c), Dst: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9), Internet Protocol Version 4, Src: 192.168.31.243, Dst: 192.168.31.234					
▶ Transmission Control Protocol, Src Port: 12023, Dst Port: 4939, Seq: 921, Ack: 10771, Len: 696					
▼ Hypertext Transfer Protocol					
▶ POST /html/dopost HTTP/1.1\r\n					
Host: 192.168.31.234:4939\r\n					
Connection: keep-alive\r\n					
Content-Length: 25\r\n					
[Content length: 25]					
Cache-Control: max-age=0\r\n					
Upgrade-Insecure-Requests: 1\r\n					
Origin: http://192.168.31.234:4939\r\n					
Content-Type: application/x-www-form-urlencoded\r\n					
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chr					
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,af					
Referer: http://192.168.31.234:4939/html/test.html\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					
\r\n					
0220	2c 61 70 70 6c 69 63 61	74 69 6f 6e 2f 73 69 67	, applica tion/		
0230	6e 65 64 2d 65 78 63 68	61 6e 67 65 3b 76 3d 62	ned-exch		
0240	33 3b 71 3d 30 2e 39 0d	0a 52 65 66 65 72 65 72	3;q=0.9		
0250	3a 20 68 74 74 70 3a 2f	2f 31 39 32 2e 31 36 38	: http:/ /		
0260	2e 33 31 2e 32 33 34 3a	34 39 33 39 2f 68 74 6d	.31.234: 4939/		
0270	6c 2f 74 65 73 74 2e 68	74 6d 6c 0d 0a 41 63 63	l/test.h		
0280	65 70 74 2d 45 6e 63 6f	64 69 6e 67 3a 20 67 7a	ept-Enco ding:		
0290	69 70 2c 20 64 65 66 6c	61 74 65 0d 0a 41 63 63	ip, defl		
02a0	65 70 74 2d 4c 61 6e 67	75 61 67 65 3a 20 7a 68	ept-Lang uage:		
02b0	2d 43 4e 2c 7a 68 3b 71	3d 30 2e 39 2c 65 6e 3b	-CN,zh;q		
02c0	71 3d 30 2e 38 2c 65 6e	2d 47 42 3b 71 3d 30 2e	q=0.8,en		
02d0	37 2c 65 6e 2d 55 53 3b	71 3d 30 2e 36 0d 0a 0d	7,en-US;		
02e0	0a 6c 6f 67 69 6e 3d 33	31 38 30 31 30 34 39 33	-login=3		
02f0	39 26 70 61 73 73 3d 31	32 33	9&pass=1 23		

```

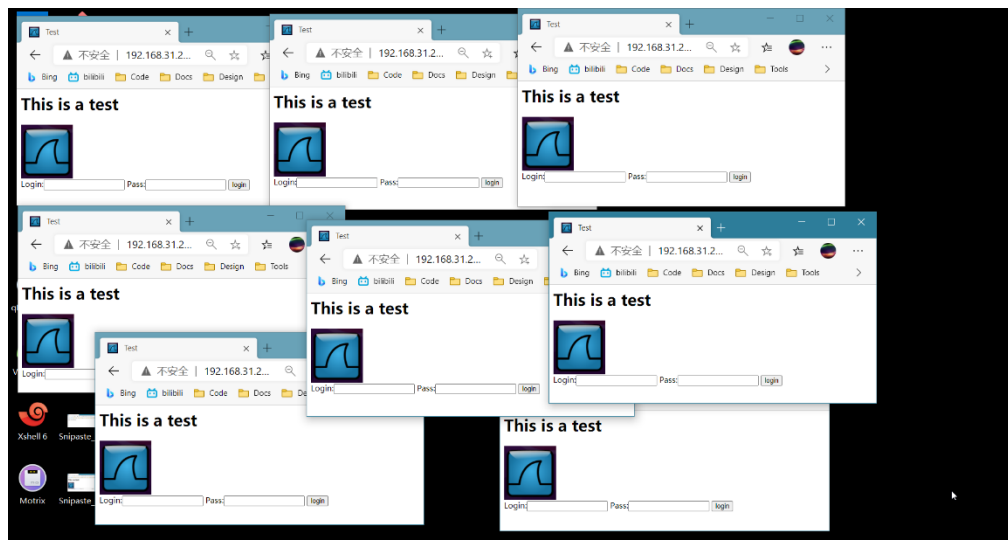
963 340.664452830 192.168.31.234 192.168.31.243 TCP 66 4939 → 12023 [ACK] Seq=
964 340.664571781 192.168.31.234 192.168.31.243 HTTP 200 HTTP/1.1 200 OK (text/html)
965 340.723188611 192.168.31.243 192.168.31.243 TCP 66 12023 → 4939 [ACK] Seq=
979 350.385592688 192.168.31.243 192.168.31.243 TCP 60 [TCP Keep-Alive] 2243
980 350.385600994 192.168.31.234 192.168.31.243 TCP 78 [TCP Keep-Alive ACK] 2204
982 357.074322345 192.168.31.234 192.168.31.243 TCP 60 [TCP Keep-Alive] 2204

> Ethernet II, Src: ASUSTekC_cd:c0:a9 (24:4b:fe:cd:c0:a9), Dst: IntelCor_34:98:9c (14:4f:8a:34:98:9c)
> Internet Protocol Version 4, Src: 192.168.31.234, Dst: 192.168.31.243
> Transmission Control Protocol, Src Port: 4939, Dst Port: 12023, Seq: 10771, Ack: 1617, Len: 134
> Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Server: Martinit's http server\r\n
    Content-Type: text/html\r\n
    Content-Length: 38\r\n
      [Content length: 38]
    \r\n
    [HTTP response 3/3]
    [Time since request: 0.000135141 seconds]
    [Prev request in frame: 933]
    [Prev response in frame: 937]
    [Request in frame: 962]
    [Request URI: http://192.168.31.234:4939/html/dopost]
    File Data: 38 bytes
> Line-based text data: text/html (1 lines)

0000 14 4f 8a 34 98 9c 24 4b fe cd c0 a9 08 00 45 00 04 .$.K
0010 00 ba 55 57 40 00 40 06 23 b9 c0 a8 1f ea c0 a8 ..Uw@.
0020 1f f3 13 4b 2e f7 0d 3b f4 ef 8d d6 99 c1 80 18 ...K...;
0030 01 f5 c1 da 00 00 01 01 08 0a 36 2c 35 f1 10 b9 .....
0040 e8 69 48 54 54 50 2f 31 2e 31 20 32 30 30 20 4f .iHTTP/1 .1 200
0050 4b 0d 0a 53 65 72 76 65 72 3a 20 4d 61 72 74 69 K..Serve r:
0060 6e 69 74 27 73 20 68 74 74 70 20 73 65 72 76 65 nit's ht tp
0070 72 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a r..Conte nt-
0080 20 74 65 78 74 2f 68 74 6d 6c 0d 0a 43 6f 6e 74 text/ht
0090 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 38 0d 0a ent-Leng th:
00a0 0d 0a 3c 68 74 6d 6c 3e 3c 62 6f 64 79 3e 4c 6f ..<html>
00b0 67 69 6e 20 46 61 69 6c 65 64 3c 2f 62 6f 64 79 gin Fail ed</
00c0 3e 3c 2f 68 74 6d 6c 3e ></html>

```

- 多个浏览器同时访问包含图片的 HTML 文件时，浏览器的显示内容截图（将浏览器窗口缩小并列）



- 多个浏览器同时访问包含图片的 HTML 文件时,使用 `netstat -an` 显示服务器的 TCP 连接（截取与服务器监听端口相关的）

```

tcp        0      0 192.168.31.234:4939 0.0.0.0: LISTEN
martinit@ubuntu-martinit:~/zju/cnlab/html/img$ netstat -an | grep 4939
tcp        0      0 192.168.31.234:4939 0.0.0.0:* LISTEN
tcp        0      0 192.168.31.234:4939 192.168.31.243:2272 CLOSE_WAIT
tcp        423      0 192.168.31.234:4939 192.168.31.243:2243 CLOSE_WAIT
tcp        491      0 192.168.31.234:4939 192.168.31.243:12023 CLOSE_WAIT
tcp        0      0 192.168.31.234:4939 192.168.31.243:12051 CLOSE_WAIT
tcp        491      0 192.168.31.234:4939 192.168.31.243:2242 CLOSE_WAIT
tcp        0      0 192.168.31.234:4939 192.168.31.243:12080 CLOSE_WAIT
tcp        0      0 192.168.31.234:4939 192.168.31.243:2281 CLOSE_WAIT
tcp        0      0 192.168.31.234:4939 192.168.31.243:12125 ESTABLISHED
tcp        0      0 192.168.31.234:4939 192.168.31.243:12022 CLOSE_WAIT
tcp        0      0 192.168.31.234:4939 192.168.31.243:12081 ESTABLISHED
tcp        0      0 192.168.31.234:4939 192.168.31.243:2287 CLOSE_WAIT
martinit@ubuntu-martinit:~/zju/cnlab/html/img$

```

六、 实验结果与分析

根据你编写的程序运行效果，分别解答以下问题（看完请删除本句）：

- HTTP 协议是怎样对头部和体部进行分隔的？
使用一个空行进行分隔，即\r\n
- 浏览器是根据文件的扩展名还是根据头部的哪个字段判断文件类型的？
根据 Content-Type 判断
- HTTP 协议的头部是不是一定是文本格式？体部呢？
一定是，体部不一定。
- POST 方法传递的数据是放在头部还是体部？两个字段是用什么符号连接起来的？
体部，使用&分隔

七、 讨论、心得

本实验在 socket 的基础上实现了对 http1.1 协议的解析,实现了一个建议的 http 服务器，这让我对计算机网络有了更深的理解。