



南开大学
Nankai University

南 开 大 学

网 络 空 间 安 全 学 院

计算机网络实验报告

Web 服务器与 Wireshark 抓包

李潇逸 2111454

年级：2021 级

专业：信息安全、法学

指导教师：张建忠

2023 年 11 月 2 日

目录

一、 实验要求	1
二、 搭建 web 服务器	1
三、 web 页面展示	1
四、 通过 Wireshark 捕获与 Web 服务器的交互过程	2
(一) TCP 链接	2
1. 简述	2
2. TCP 头部	2
3. TCP 三次握手建立连接	3
4. TCP 四次挥手关闭连接	4
(二) 网页访问	5
(三) 页面链接与断开	5
(四) 捕捉	6
1. 捕捉文字	6
2. 捕捉图片	7
3. 捕捉音视频	7
五、 总结	7

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。页面不要太复杂，包含要求的基本信息即可。

(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，并进行简单的分析说明。

(3) 使用 HTTP，不要使用 HTTPS。

(4) 提交实验报告。

二、 搭建 web 服务器

- 系统：Windows11
- 软件：phpnow
- 语言：html、php
- IP：192.168.30.132
- 端口：80

安装软件后将书写的 web 文件存放在 C:/Desktop/PHP/htdocs/my_web

三、 web 页面展示

在物理机的浏览器中输入 `http://192.168.30.132/my_web/welcome.html` 后可以在浏览器中得到如下界面：

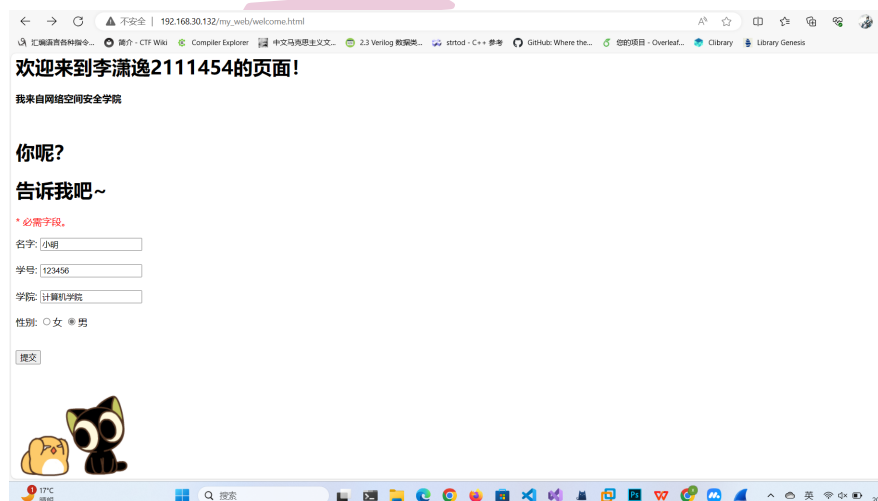


图 1: web 页面 1

本界面中有本人的姓名、学号、输入框、gif 图和外部链接，在输入相关内容后浏览器会使用 POST 方式将其传入 `my_web_main.php` 页面之中：



图 2: web 页面 2

四、通过 Wireshark 捕获与 Web 服务器的交互过程

(一) TCP 链接

1. 简述

TCP 提供面向有连接的通信传输，面向有连接是指在传送数据之前必须先建立连接，数据传送完成后要释放连接。

无论哪一方向另一方发送数据之前，都必须先在双方之间建立一条连接。在 TCP/IP 协议中，TCP 协议提供可靠的连接服务，连接是通过三次握手进行初始化的。同时由于 TCP 协议是一种面向连接的、可靠的、基于字节流的运输层通信协议，TCP 是全双工模式，所以需要四次挥手关闭连接。

2. TCP 头部

TCP 头部的定义如下：



图 3: TCP 头部

对这些部分的分析如下：

1. TCP 端口号。TCP 的连接是需要四个要素确定唯一一个连接：（源 IP，源端口号）+（目的地 IP，目的端口号）。所以 TCP 首部预留了两个 16 位作为端口号的存储，而 IP 地址由上一层 IP 协议负责传递。源端口号和目的地端口各占 16 位两个字节，也就是端口的范围是 $2^{16} = 65535$ 。另外 1024 以下是系统保留的，从 1024-65535 是用户使用的端口范围。
2. TCP 序号和确认号。

- 32 位序号 seq: Sequence number 缩写 seq, TCP 通信过程中某一个传输方向上的字节流的每个字节的序号, 通过这个来确认发送的数据有序, 比如现在序列号为 1000, 发送了 1000, 下一个序列号就是 2000。
 - 32 位确认号 ack: Acknowledge number 缩写 ack, TCP 对上一次 seq 序号做出的确认号, 用来响应 TCP 报文段, 给收到的 TCP 报文段的序号 seq 加 1。
3. TCP 标志位。每个 TCP 段都有一个目的, 这是借助于 TCP 标志位选项来确定的, 允许发送方或接收方指定哪些标志应该被使用, 以便段被另一端正确处理。运用最广泛的标志是 SYN, ACK 和 FIN, 用于建立连接, 确认成功的段传输, 最后终止连接。
- SYN: 同步标志位, 用于建立会话连接, 同步序列号
 - ACK: 确认标志位, 对已接收的数据包进行确认
 - FIN: 完成标志位, 表示我已经没有数据要发送了, 即将关闭连接
 - PSH: 推送标志位, 表示该数据包被对方接收后应立即交给上层应用, 而不在缓冲区排队
 - RST: 重置标志位, 用于连接复位、拒绝错误和非法的数据包
 - URG: 紧急标志位, 表示数据包的紧急指针域有效, 用来保证连接不被阻断, 并督促中间设备尽快处理

3. TCP 三次握手建立连接

所谓三次握手 (Three-way Handshake), 是指建立一个 TCP 连接时, 需要客户端和服务端总共发送 3 个报文。三次握手的目的是连接服务器指定端口, 建立 TCP 连接, 并同步连接双方的序列号和确认号, 交换 TCP 窗口大小信息。

三次握手过程的示意图如下:

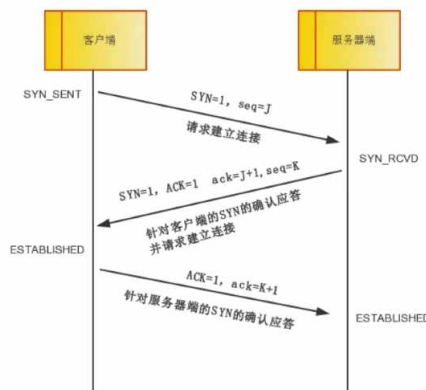


图 4: 三次握手

1. 第一次握手。客户端将 TCP 报文标志位 SYN 置为 1, 随机产生一个序号值 seq=J, 保存在 TCP 首部的序列号 (Sequence Number) 字段里, 指明客户端打算连接的服务器的端口, 并将该数据包发送给服务器端, 发送完毕后, 客户端进入 SYN_SENT 状态, 等待服务器端确认。
2. 第二次握手。服务器端收到数据包后由标志位 SYN=1 知道客户端请求建立连接, 服务器端将 TCP 报文标志位 SYN 和 ACK 都置为 1, ack=J+1, 随机产生一个序号值 seq=K, 并将该数据包发送给客户端以确认连接请求, 服务器端进入 SYN_RCVD 状态。

- 第三次握手。客户端收到确认后，检查 ack 是否为 $J+1$ ， ACK 是否为 1，如果正确则将标志位 ACK 置为 1， $ack=K+1$ ，并将该数据包发送给服务器端，服务器端检查 ack 是否为 $K+1$ ， ACK 是否为 1，如果正确则连接建立成功，客户端和服务端进入 $ESTABLISHED$ 状态，完成三次握手，随后客户端与服务端之间可以开始传输数据了。

注意: 我们上面写的 ack 和 ACK ，不是同一个概念：

- 小写的 ack 代表的是头部的确认号 Acknowledge number，缩写 ack ，是对上一个包的序号进行确认的号， $ack=seq+1$ 。
- 大写的 ACK ，则是我们上面说的 TCP 首部的标志位，用于标志的 TCP 包是否对上一个包进行了确认操作，如果确认了，则把 ACK 标志位设置成 1。

4. TCP 四次挥手关闭连接

四次挥手即终止 TCP 连接，就是指断开一个 TCP 连接时，需要客户端和服务端总共发送 4 个包以确认连接的断开。在 socket 编程中，这一过程由客户端或服务端任一方执行 `close` 来触发。由于 TCP 连接是全双工的，因此，每个方向都必须单独进行关闭，这一原则是当一方完成数据发送任务后，发送一个 FIN 来终止这一方向的连接，收到一个 FIN 只是意味着这一方向上没有数据流动了，即不会再收到数据了，但是在这个 TCP 连接上仍然能够发送数据，直到这一方向也发送了 FIN 。首先进行关闭的一方将执行主动关闭，而另一方则执行被动关闭。

四次挥手过程的示意图如下：

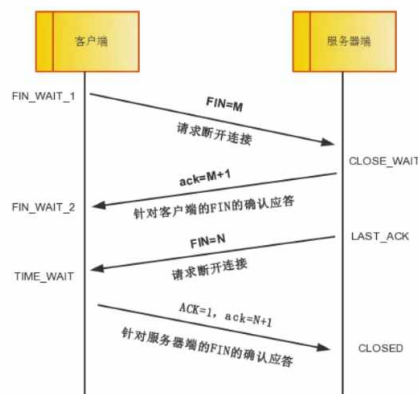


图 5: 四次挥手

- 第一次挥手：Client 端发起挥手请求，向 Server 端发送标志位是 FIN 报文段，设置序列号 seq ，此时，Client 端进入 FIN_WAIT_1 状态，这表示 Client 端没有数据要发送给 Server 端了。
- 第二次分手：Server 端收到了 Client 端发送的 FIN 报文段，向 Client 端返回一个标志位是 ACK 的报文段， ack 设为 seq 加 1，Client 端进入 FIN_WAIT_2 状态，Server 端告诉 Client 端，我确认并同意你的关闭请求。
- 第三次分手：Server 端向 Client 端发送标志位是 FIN 的报文段，请求关闭连接，同时 Client 端进入 $LAST_ACK$ 状态。

4. 第四次分手：Client 端收到 Server 端发送的 FIN 报文段，向 Server 端发送标志位是 ACK 的报文段，然后 Client 端进入 TIME_WAIT 状态。Server 端收到 Client 端的 ACK 报文段以后，就关闭连接。此时，Client 端等待 2MSL 的时间后依然没有收到回复，则证明 Server 端已正常关闭，那好，Client 端也可以关闭连接了。

(二) 网页访问

打开 Wireshark 界面。由于 Vmware 虚拟机使用 Net 连接，所使用的是 Vmnet8，因此使用 wireshark 监听 Vmnet8。

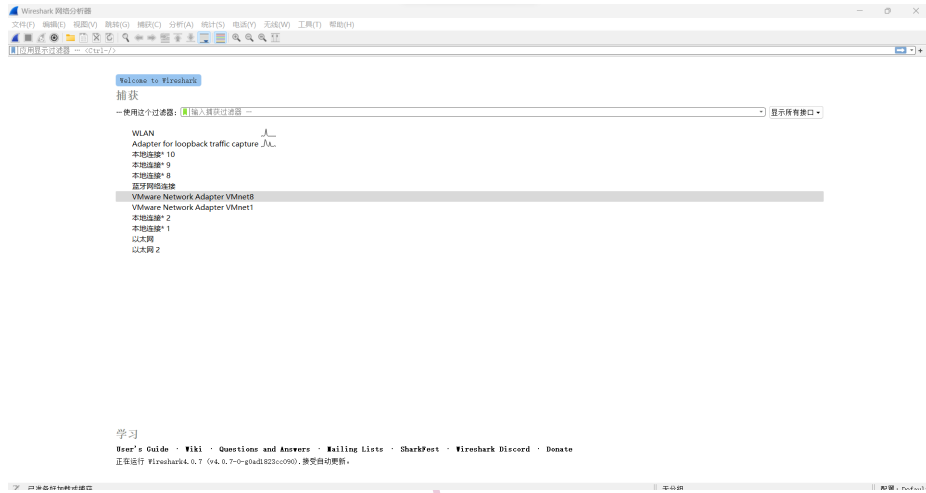


图 6: wireshark

开始进行捕捉后可以发现如下界面：

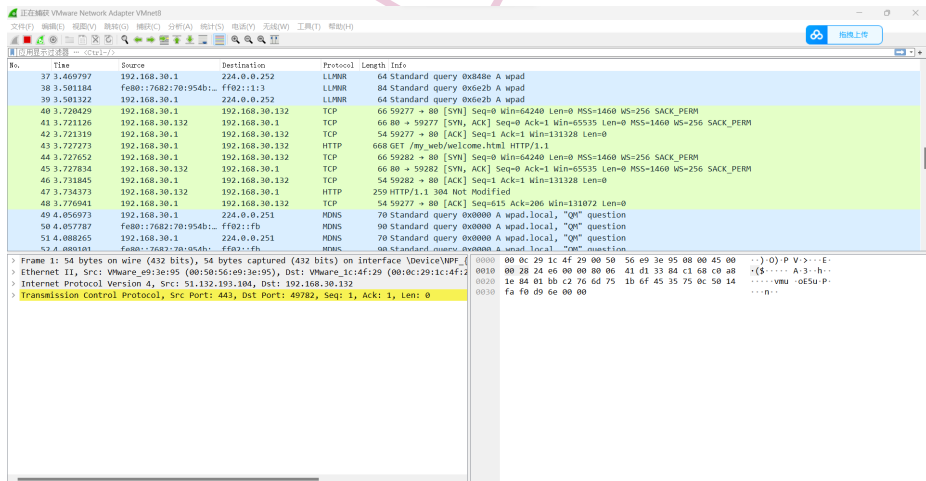


图 7: 捕捉界面

(三) 页面链接与断开

页面链接时将会首先进行如下的三次握手：

32.392681	192.168.30.1	192.168.30.132	TCP	66.56104 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
42.392855	192.168.30.132	192.168.30.1	TCP	54.56104 → 80 [FIN, ACK] Seq=1 Ack=1 Min=131328 Len=0
52.392961	192.168.30.1	192.168.30.132	TCP	54.56104 → 80 [ACK] Seq=1 Ack=1 Min=131328 Len=0
62.404925	192.168.30.1	192.168.30.132	TCP	66.56106 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
72.405241	192.168.30.132	192.168.30.1	TCP	66.80 → 56106 [SYN, ACK] Seq=0 Ack=1 Min=65535 Len=0 MSS=1460 WS=256 SACK_PERM
82.405332	192.168.30.1	192.168.30.132	TCP	54.56106 → 80 [ACK] Seq=1 Ack=1 Min=131328 Len=0

图 8: 三次握手

断开时将会进行如下的四次挥手：

97.104218	192.168.30.1	192.168.30.132	TCP	54.56104 → 80 [FIN, ACK] Seq=1 Ack=1 Min=131328 Len=0
107.104262	192.168.30.1	192.168.30.132	TCP	54.56106 → 80 [FIN, ACK] Seq=1 Ack=1 Min=131328 Len=0
117.104640	192.168.30.132	192.168.30.1	TCP	54.80 → 56104 [ACK] Seq=1 Ack=2 Win=2097920 Len=0
127.104784	192.168.30.132	192.168.30.1	TCP	54.80 → 56106 [ACK] Seq=1 Ack=2 Win=2097920 Len=0
137.104927	192.168.30.132	192.168.30.1	TCP	54.80 → 56106 [FIN, ACK] Seq=1 Ack=2 Win=2097920 Len=0
147.105109	192.168.30.132	192.168.30.1	TCP	54.80 → 56106 [FIN, ACK] Seq=1 Ack=2 Win=2097920 Len=0
157.105660	192.168.30.1	192.168.30.132	TCP	54.56104 → 80 [ACK] Seq=2 Ack=2 Min=131328 Len=0
167.105793	192.168.30.1	192.168.30.132	TCP	54.56106 → 80 [ACK] Seq=2 Ack=2 Min=131328 Len=0

图 9: 四次挥手

(四) 捕捉

在输入内容后可以到达另一页面，我们将在这一页面中捕捉文字、图片和音视频。

662.236.716646	192.168.30.1	192.168.30.132	TCP	66.56526 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
663.236.717030	192.168.30.132	192.168.30.1	TCP	66.80 → 56526 [SYN, ACK] Seq=0 Ack=1 Min=65535 Len=0 MSS=1460 WS=256 SACK_PERM
664.236.717389	192.168.30.1	192.168.30.132	TCP	54.56526 → 80 [ACK] Seq=1 Ack=1 Min=131328 Len=0
665.236.816265	192.168.30.1	192.168.30.132	HTTP	850 POST /my_web/my_web_main.php HTTP/1.1 (application/x-www-form-urlencoded)
666.236.816642	192.168.30.1	192.168.30.132	TCP	66.56528 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
667.236.816852	192.168.30.132	192.168.30.1	TCP	66.80 → 56528 [SYN, ACK] Seq=0 Ack=1 Min=65535 Len=0 MSS=1460 WS=256 SACK_PERM
668.236.817402	192.168.30.1	192.168.30.132	TCP	54.56528 → 80 [ACK] Seq=1 Ack=1 Min=131328 Len=0
669.236.857615	192.168.30.132	192.168.30.1	HTTP	1027 HTTP/1.1 200 OK (text/html)
670.236.863084	192.168.30.1	192.168.30.132	HTTP	522 GET /my_web/image/3e73805973e53080a3e3080e912.gif HTTP/1.1
671.236.868860	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=9734 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
672.236.868877	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=2434 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
673.236.868882	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=3894 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
674.236.868895	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=5354 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
675.236.868901	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=6814 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
676.236.868919	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=8274 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
677.236.868934	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=9734 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
678.236.868947	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=11184 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
679.236.868958	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=12654 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
680.236.868971	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=14114 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
681.236.872084	192.168.30.1	192.168.30.132	TCP	54.56526 → 80 [ACK] Seq=1265 Ack=15574 Min=131328 Len=0
682.236.873130	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=15574 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
683.236.873141	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=17034 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
684.236.873147	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=18494 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
685.236.873152	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=19954 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
686.236.873162	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=21414 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
687.236.873173	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=22874 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
688.236.873181	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=24334 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
689.236.873190	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=25794 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]
690.236.873201	192.168.30.132	192.168.30.1	TCP	1514.80 → 56526 [ACK] Seq=27254 Ack=1265 Min=2096640 Len=1460 [TCP segment of a reassembled PDU]

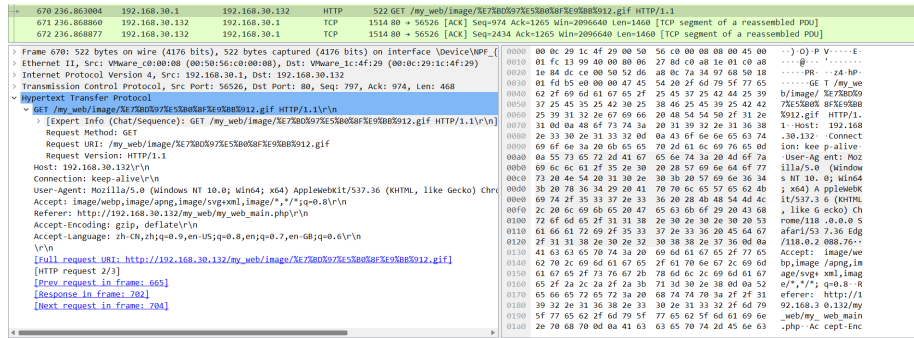
图 10: 捕捉另一页面

1. 捕捉文字

669.236.857615	192.168.30.132	192.168.30.1	HTTP	1027 HTTP/1.1 200 OK (text/html)
Frame 669: 1027 bytes on wire (8216 bits), 1027 bytes captured (8216 bits) on interface VDeviceW				0000 00 50 56 c0 00 08 00 0c 29 1c 4f 29 08 00 45 00 PV.....)O)E
Ethernet II, Src: VMware_1c:4f:29 (00:0c:29:1c:4f:29), Dst: VMware_c8:00:08 (00:50:56:c8:00:08)				0010 03 f5 81 1d 00 00 80 06 08 0f c0 a8 1e 84 c0 a8 ...@.....
Internet Protocol Version 4, Src: 192.168.30.132, Dst: 192.168.30.1				0020 1e 01 00 50 56 c8 7a 34 03 00 52 46 a8 0c 50 18 ...P.2A.R..p.
Transmission Control Protocol, Src Port: 80, Dst Port: 56526, Seq: 1, Ack: 797, Len: 973				0030 20 00 70 20 00 00 48 54 54 50 2f 31 2e 31 20 32 ...p..HT TP/1.1.2
Hypertext Transfer Protocol				0040 30 30 20 4f 4b 0d 0a 44 61 74 65 3a 20 54 64 75 00 OK..D ate: 1
Line-based text data: text/html (21 lines)				0050 2c 20 28 32 20 4e 6f 76 20 32 30 32 31 20 31 34 ..Nov. 2023 14
<DOCTYPE html>\n				0060 34 30 30 3a 33 36 20 47 4d 54 0d 0a 53 65 72 76 ..00:36 G NT-Serv
<html lang="en">\n				0070 65 72 3a 20 41 70 61 63 68 65 2f 32 2e 32 26 31 ..Apac he/2.2.1
<head>\n				0080 36 20 28 57 69 6e 33 32 29 20 50 48 50 2f 35 2e 6 (dnt)2 PMP/5.
<meta charset="utf-8">\n				0090 32 2e 31 34 0d 0a 58 2d 50 6f 77 65 72 65 64 2d 2.14-X- Powered:
<meta name="viewport" content="width=device-width, initial-scale=1.0">\n				00a0 45 6f 76 65 6e 7a 2d 4c 65 6e 67 74 68 3a 20 Content: Length:
<title>李道逸的小网页</title>\n				00b0 37 34 37 0d 0a 4b 65 65 70 2d 41 6c 69 76 65 3a 747-kee p-Alive:
</head>\n				00c0 20 74 69 6d 65 6f 75 74 3d 35 2c 20 6d 61 70 3d Timeout=5, max=
<body>\n				00d0 31 30 30 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a 100-Connection:
<h1>欢迎来到 计算机学院 的小明先生 </h1>\n				00e0 20 4b 65 65 70 2d 41 6c 69 76 65 0d 0a 43 6f 6e Keep-Alive: Con
<p>这是一个简单的WEB测试页面</p>\n				00f0 74 65 6e 74 2d 54 79 70 65 3a 20 74 65 7a 2f text-Type: text/
<p>开始学习啦</p>\n				0100 68 74 6d 6c 0d 0a 0d 0a 3c 21 44 4f 43 54 59 50 html..<DOCTYPE
<video width="320" height="240" autoplay>\n				0110 45 20 68 74 6d 6c 3e 0d 0a 3c 68 74 6d 6c 20 6c E html..<html 1
<div>\n				0120 61 6e 67 3d 22 65 6a 22 3e 0d 0a 3c 6d 65 61 64 ang="et"><head
\n				0130 3e 0d 0a 20 20 20 3c 6d 65 74 61 20 63 68 61 ..< meta cha
<div>\n				0140 72 73 65 74 3d 22 55 54 46 2d 38 22 3e 0d 0a 20 rset="UTF-8">..
<div>\n				0150 20 20 20 3c 6d 65 74 61 20 6e 61 6d 65 3d 22 76 ..meta name="v
<p>视频播放</p>\n				0160 69 65 77 70 6f 72 74 22 20 63 6f 6e 74 65 6e 74 iviewport" content
\n				0170 3d 22 77 69 64 74 68 3d 64 65 76 69 63 65 2d 77 "width=device-w
</div>\n				0180 69 64 74 68 2c 20 69 6e 69 74 69 61 6c 2d 77 63 idth, in trial-sc
</body>\n				0190 61 6c 65 3d 31 2e 30 22 3e 0d 0a 20 20 20 3c ale=1.0">..<

图 11: 捕捉文字

2. 捕捉图片

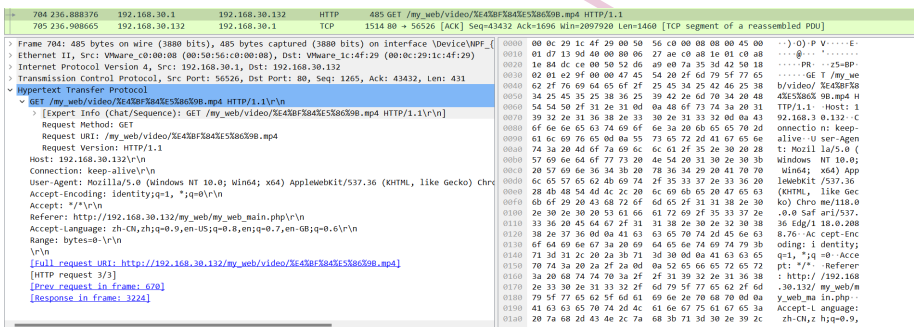


670.236.863804	192.168.30.1	192.168.30.132	HTTP	522 GET /my_web/image/Xe7800707e53808f3e9808012.gif HTTP/1.1
671.236.868800	192.168.30.132	192.168.30.1	TCP	1514 80 → 56526 [ACK] Seq=974 Ack=1265 Win=1460 [TCP segment of a reassembled PDU]
672.236.868877	192.168.30.132	192.168.30.1	TCP	1514 80 → 56526 [ACK] Seq=2434 Ack=1265 Win=2095660 Len=1460 [TCP segment of a reassembled PDU]
Frame 670: 522 bytes on wire (4176 bits), 522 bytes captured (4176 bits) on interface \Device\NPF_{...} Ethernet II, Src: VMware_c0:00:00:00:00:00, Dst: VMware_1c:4f:29:00:0c:29:1c:4f:29 Internet Protocol Version 4, Src: 192.168.30.1, Dst: 192.168.30.132 Transmission Control Protocol, Src Port: 56526, Dst Port: 80, Seq: 974, Len: 468				
Hypertext Transfer Protocol				
GET /my_web/image/Xe7800707e53808f3e9808012.gif HTTP/1.1\r\n				
[Expert Info (Chat/Sequence): GET /my_web/image/Xe7800707e53808f3e9808012.gif HTTP/1.1\r\n]				
Request Method: GET				
Request URI: /my_web/image/Xe7800707e53808f3e9808012.gif				
Request Version: HTTP/1.1				
Host: 192.168.30.132\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/69.0.3497.100 Safari/537.36\r\n				
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8\r\n				
Referer: http://192.168.30.132/my_web/my_main.php\r\n				
Accept-Encoding: gzip, deflate\r\n				
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6\r\n\r\n				
[Full request URI: http://192.168.30.132/my_web/image/Xe7800707e53808f3e9808012.gif]				
[HTTP request 2/3]				
[Prev request in frame: 665]				
[Response in frame: 292]				
[Host request in frame: 284]				

图 12: 捕捉图片

从图片路径与自己设置的路径一致以及 16 进制相同可以看出正确抓到。

3. 捕捉音视频



704.236.888376	192.168.30.1	192.168.30.132	HTTP	485 GET /my_web/video/Xe430f384f3e53808f3e9808012.mp4 HTTP/1.1
705.236.908665	192.168.30.132	192.168.30.1	TCP	1514 80 → 56526 [ACK] Seq=4342 Ack=1696 Win=2097920 Len=1460 [TCP segment of a reassembled PDU]
Frame 704: 485 bytes on wire (3880 bits), 485 bytes captured (3880 bits) on interface \Device\NPF_{...} Ethernet II, Src: VMware_c0:00:00:00:00:00, Dst: VMware_1c:4f:29:00:0c:29:1c:4f:29 Internet Protocol Version 4, Src: 192.168.30.1, Dst: 192.168.30.132 Transmission Control Protocol, Src Port: 56526, Dst Port: 80, Seq: 1265, Ack: 4342, Len: 431				
Hypertext Transfer Protocol				
GET /my_web/video/Xe430f384f3e53808f3e9808012.mp4 HTTP/1.1\r\n				
[Expert Info (Chat/Sequence): GET /my_web/video/Xe430f384f3e53808f3e9808012.mp4 HTTP/1.1\r\n]				
Request Method: GET				
Request URI: /my_web/video/Xe430f384f3e53808f3e9808012.mp4				
Request Version: HTTP/1.1				
Host: 192.168.30.132\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/69.0.3497.100 Safari/537.36\r\n				
Accept: */*\r\n				
Referer: http://192.168.30.132/my_web/my_main.php\r\n				
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6\r\n				
Range: bytes=0-1\r\n\r\n				
[Full request URI: http://192.168.30.132/my_web/video/Xe430f384f3e53808f3e9808012.mp4]				
[HTTP request 3/3]				
[Prev request in frame: 676]				
[Response in frame: 3224]				

图 13: 捕捉音视频

从音视频路径与自己设置的路径一致以及 16 进制相同可以看出正确抓到。

五、 总结

本次实验编写了一个简单的 web 页面，并使用 wireshark 进行抓包，深入理解了相应原理，对计算机网络有了更深入的了解。具体过程了解如下：

- 客户端发出请求，服务器在局域网内（因为使用的是本地的虚拟机）发送查找网卡请求并找到 ip 对应的物理地址
- 三次握手建立连接
- 进行页面请求，服务器返回 html 内容
- 请求并返回图片和视频等内容
- 四次挥手断开连接