华东师范大学软件工程学院实验报告

实验课程:	计算机网络	年　　级:	2022 级
实验编号:	Lab 01	实验名称:	Protocol Layer
姓　　名:	李鹏达	学　　号:	10225101460

1　实验目的

1) 学会通过 Wireshark 获取各协议层的数据包
2) 掌握协议层数据包结构
3) 分析协议开销

2　实验内容与实验步骤

2.1　实验内容

2.1.1　获取协议层的数据包

使用 `wget` 命令发起 `HTTP` 请求，然后使用 `Wireshark` 抓包。

2.1.2　绘制数据包结构

分析 `HTTP GET` 协议包的内容，并绘制协议包，分别标出 `Ethernet`，`IP` 和 `TCP` 协议的头部的位置、大小以及其负载的范围。

2.1.3　分析协议开销

根据实验结构，分析 `HTTP` 应用协议额外开销。估计上面捕获的 `HTTP` 协议的额外开销。假设 `HTTP` 数据（头部和消息）是有用的，而 `TCP`，`IP` 和 `Ethernet` 头部认为是开销。对于下载的主要部分中的每一个包，我们需要分析 `Ethernet`，`IP` 和 `TCP` 的开销，和有用的 `HTTP` 数据的开销。根据以上的定义来估计下载协议的开销，你认为这种开销是必要的吗?

2.1.4　分析解复用键

解复用指找到正确的上一层协议来处理到达的包。
观察下载的以太网和 `IP` 包头部信息，回答下面问题:

1. 以太网头部中哪一部分是解复用键并且告知它的下一个高层指的是 `IP`，在这一包内哪一个值可以表示 `IP`?

2. IP 头部中哪一部分是解复用键并且告知它的下一一个高层指的是 TCP，在这一包内哪一个值可以表示 TCP?

2.1.5 问题讨论

1. Look at a short TCP packet that carries no higher-layer data. To what entity is this packet destined? After all, if it carries no higher-layer data then it does not seem very useful to a higher layer protocol such as HTTP!
2. In a classic layered model, one message from a higher layer has a header appended by the lower layer and becomes one new message. But this is not always the case. Above, we saw a trace in which the web response (one HTTP message comprised of an HTTP header and an HTTP payload) was converted into multiple lower layer messages (being multiple TCP packets). Imagine that you have drawn the packet structure (as in step 2) for the first and last TCP packet carrying the web response. How will the drawings differ?
3. In the classic layered model described above, lower layers append headers to the messages passed down from higher layers. How will this model change if a lower layer adds encryption?
4. In the classic layered model described above, lower layers append headers to the messages passed down from higher layers. How will this model change if a lower layer adds compression?

2.2 实验步骤

1) 安装实验所需软件（为方便安装，我们使用 Windows 下的包管理器 winget）

```
1 PS> winget install wget
2 PS> winget install wireshark
```

2) 打开 Wireshark，在菜单栏的 捕获 -> 选项中进行设置，选择已连接的网络，设置捕获过滤器为 tcp port 80，将混杂模式设为关闭, 勾选 enable network name resolution，然后开始捕获。
3) 使用 wget 命令发起 HTTP 请求

```
1 PS> wget http://www.baidu.com
```

4) 在 Wireshark 中停止捕获。
5) 分析 HTTP GET 协议包的内容，并绘制协议包。
6) 分析协议开销
7) 分析解复用键
8) 问题讨论

3 实验环境

- 操作系统：Windows 11 家庭中文版 23H2 22631.2715

- 网络适配器：Killer(R) Wi-Fi 6 AX1650i 160MHz Wireless Network Adapter(201NGW)
- Wireshark：Version 4.2.0 (v4.2.0-0-g54eedfc63953)
- wget：GNU Wget 1.21.4 built on mingw32

4 实验过程与分析

4.1 获取协议层的数据包

首先，我们打开 Wireshark，在菜单栏的 **捕获 -> 选项**中进行设置，选择已连接的网络，设置捕获过滤器为 tcp port 80，将混杂模式设为关闭，勾选 enable network name resolution，然后开始捕获。

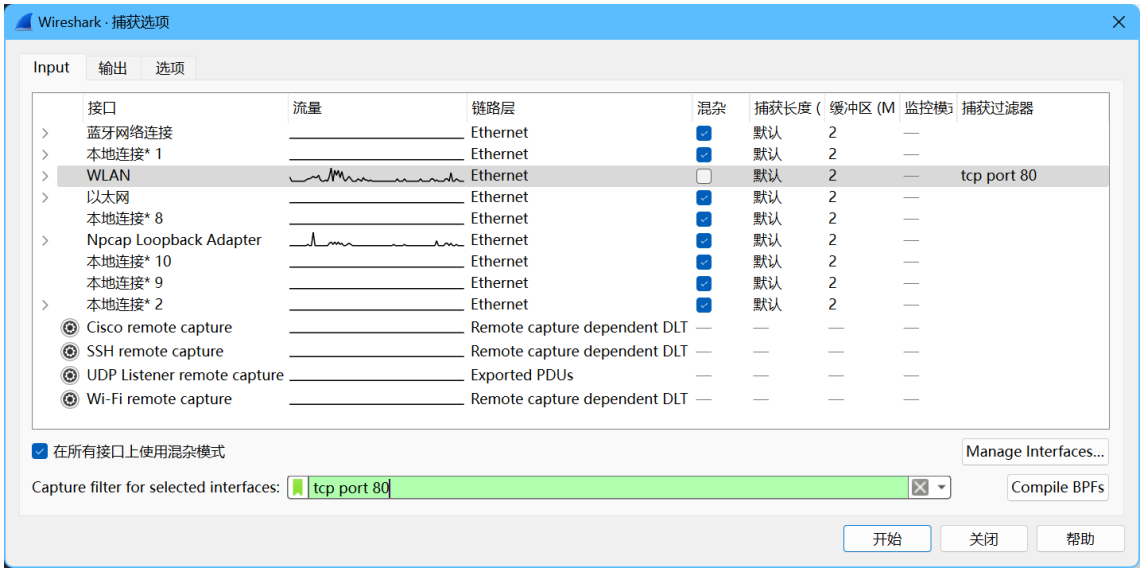

图 1: 开始捕获

然后，我们使用 wget 命令对 http://www.baidu.com 发起 HTTP 请求

图 2: 使用 wget 命令发起 HTTP 请求

最后，我们在 Wireshark 中停止捕获，得到如下结果：

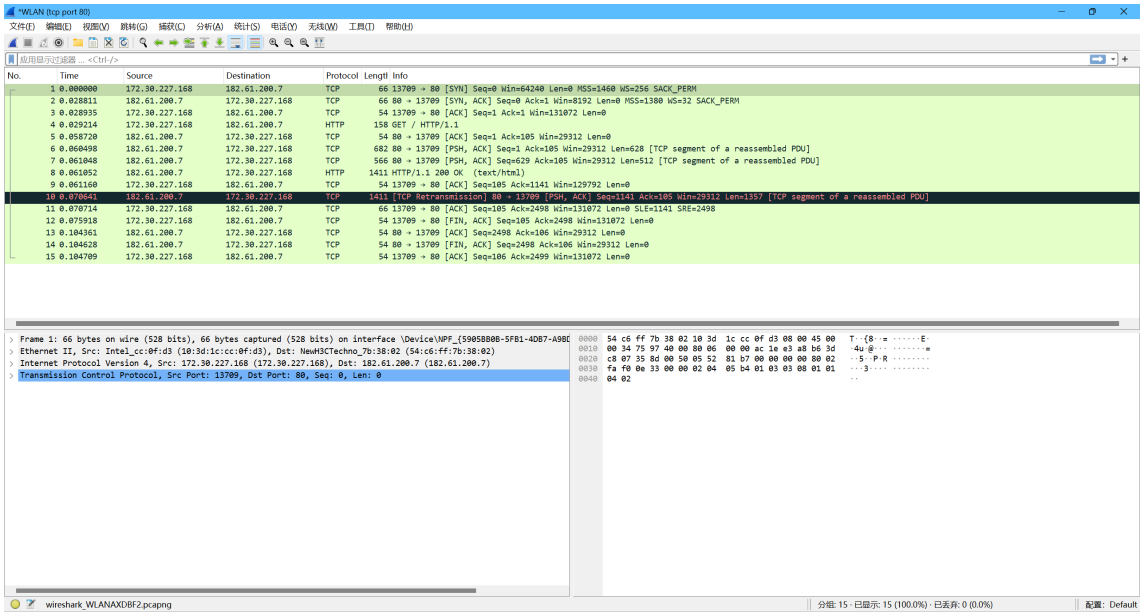

图 3: 捕获结果

4.2 绘制数据包结构

接下来，我们分析数据包的结构。在 Wireshark 中，我们选择 HTTP GET 协议包，可以看到其内容，如下图所示：

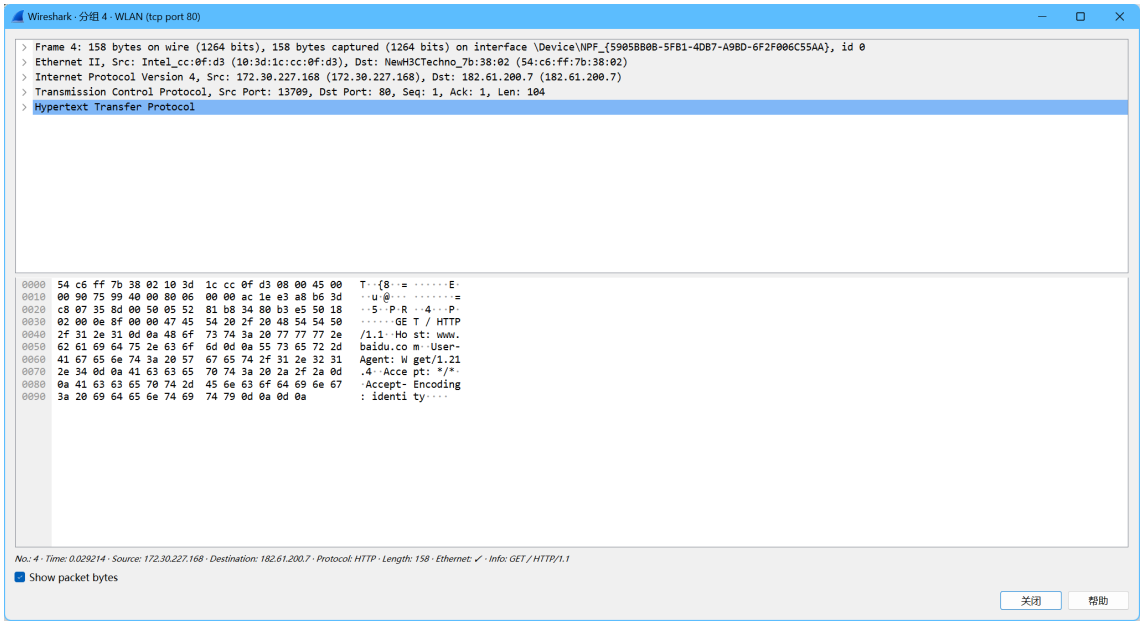

图 4: HTTP GET 协议包

其中，第一个块是 Frame，这不是一个协议，而是一个记录，描述有关数据包的整体信息，包括捕获时间和长度等。我们可以看到，整个数据包的长度是 158 字节。第二个块是 Ethernet II，这是以太网协议，可以看到这个标头的长度是 14 字节。第三个块是 Internet Protocol Version 4，这是 IP 协议，可以看到这个标头的长度是 20 字节。第四个块是 Transmission Control Protocol，这是 TCP 协议，可以看到这个标头的长度是 20 字节。第五个块是 Hypertext Transfer Protocol，这是 HTTP 协议，可以看到这个标头的长度是 104 字节。

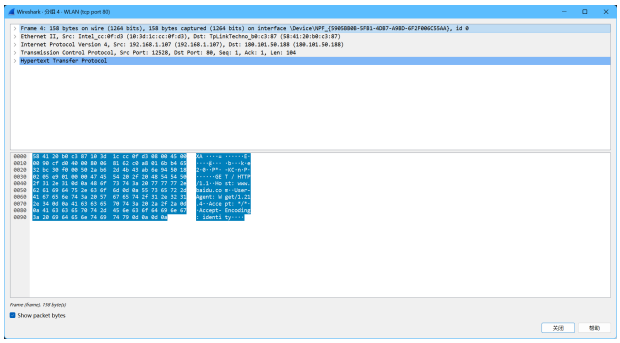

图 5: 数据包的整体信息

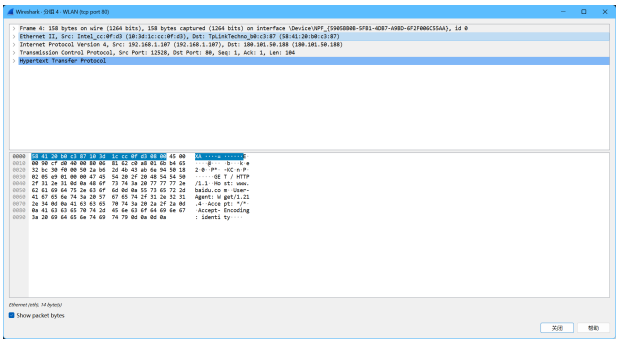

图 6: 以太网协议包

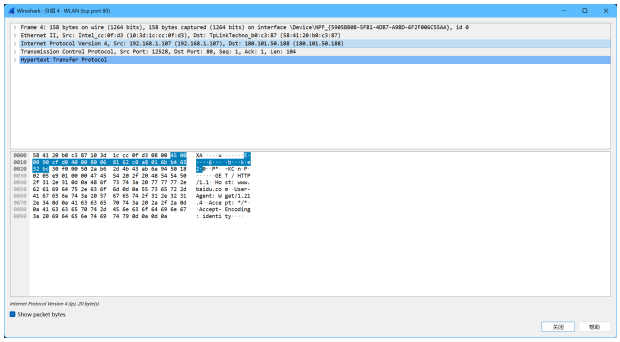

图 7: IP 协议包

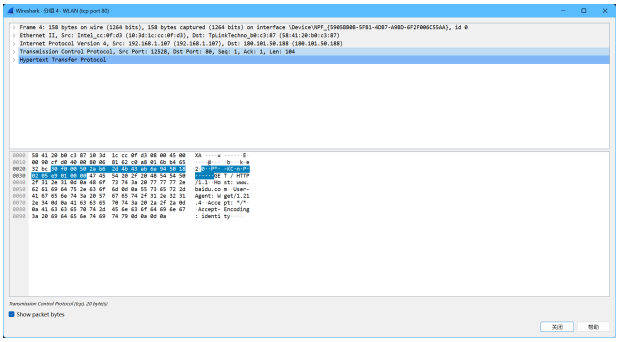

图 8: TCP 协议包

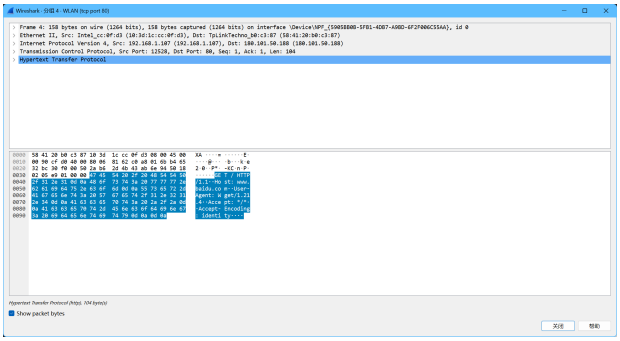

图 9: HTTP 协议包

协议包的结构如下图所示：

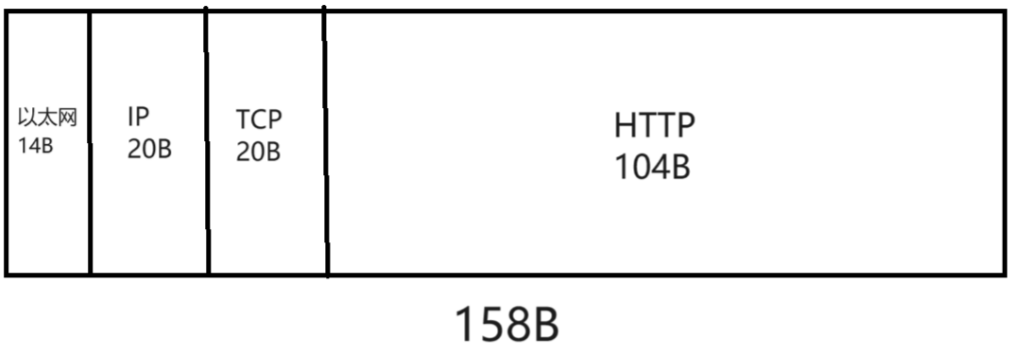

图 10: 协议包的结构

4.3 分析协议开销

以太网的开销为

$$\frac{14}{158} = 8.86\%$$

IP 的开销为

$$\frac{20}{158} = 12.66\%$$

TCP 的开销为

$$\frac{20}{158} = 12.66\%$$

整个捕获过程（从 SYN ACK 开始，到 HTTP 后的第一个 TCP 数据包结束）的开销为

$$\frac{14 + 20 + 20 + 66 + 54 \times 2}{158 + 66 + 54 \times 2} = 68.7\%$$

有用的 HTTP 数据的开销为

$$\frac{104}{158 + 66 + 54 \times 2} = 31.3\%$$

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.107	180.101.50.188	TCP	66	12528 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
2	0.009812	180.101.50.188	192.168.1.107	TCP	66	80 → 12528 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1440 WS=32 SACK_PERM
3	0.009924	192.168.1.107	180.101.50.188	TCP	54	12528 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
4	0.010554	192.168.1.107	180.101.50.188	HTTP	158	GET / HTTP/1.1
5	0.018934	180.101.50.188	192.168.1.107	TCP	54	80 → 12528 [ACK] Seq=1 Ack=105 Win=78464 Len=0
6	0.019735	180.101.50.188	192.168.1.107	TCP	682	80 → 12528 [PSH, ACK] Seq=1 Ack=105 Win=78464 Len=628 [TCP segment of a reassembled PDU]
7	0.019736	180.101.50.188	192.168.1.107	TCP	566	80 → 12528 [PSH, ACK] Seq=629 Ack=105 Win=78464 Len=512 [TCP segment of a reassembled PDU]
8	0.019738	180.101.50.188	192.168.1.107	TCP	1078	80 → 12528 [PSH, ACK] Seq=1141 Ack=105 Win=78464 Len=1024 [TCP segment of a reassembled PDU]
9	0.019738	180.101.50.188	192.168.1.107	HTTP	387	HTTP/1.1 200 OK (text/html)

图 11: 整个捕获过程

4.4 分析解复用键

1. 以太网头部中哪一部分是解复用键并且告知它的下一个高层指的是 IP，在这一包内哪一个值可以表示 IP?

   以太网头部中的 type 字段是解复用键，它的值为 0x0800，表示 IP 协议。

2. IP 头部中哪一部分是解复用键并且告知它的下一一个高层指的是 TCP，在这一包内哪一个值可以表示 TCP?

   IP 头部中的 protocol 字段是解复用键，它的值为 0x06，表示 TCP 协议。

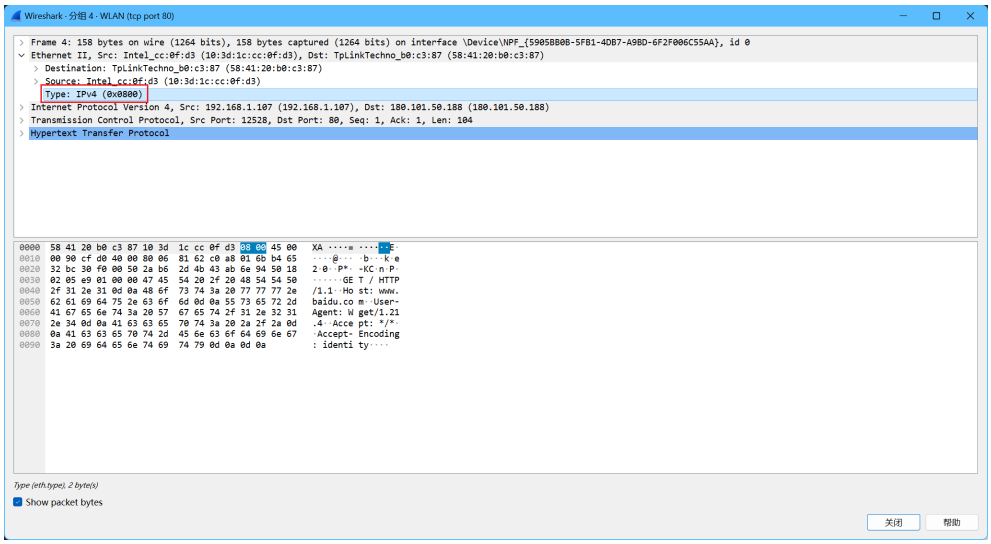

图 12: **以太网**头部

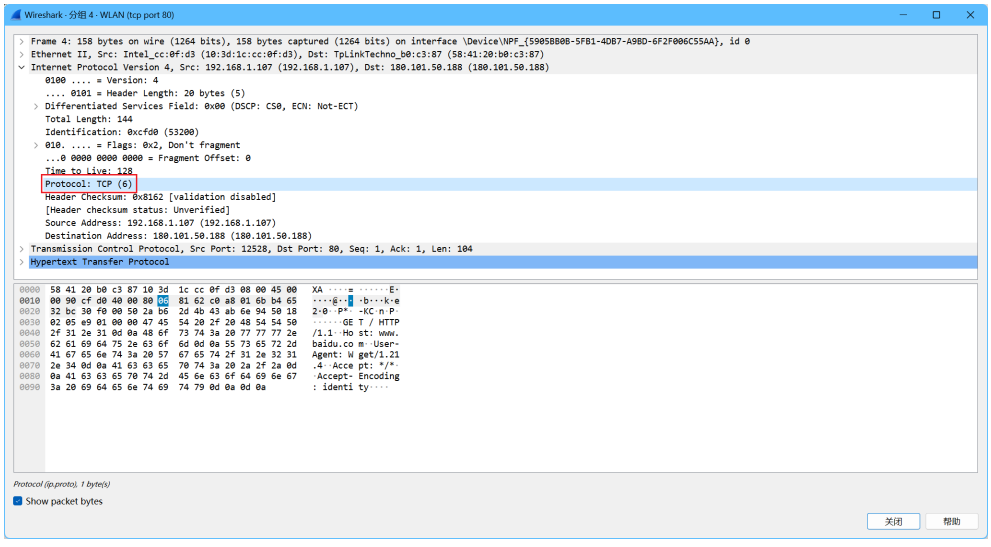

图 13: IP 头部

4.5 问题讨论

1. Look at a short TCP packet that carries no higher-layer data. To what entity is this packet destined? After all, if it carries no higher-layer data then it does not seem very useful to a higher layer protocol such as HTTP!
   这个包的目的地是我们要访问的网站或者我们自己，通常是确认和控制信息类报文，用于建立 TCP 连接，对 HTTP 协议是有用的。这是 TCP 三次握手的一部分，用于建立 TCP 连接。
2. In a classic layered model, one message from a higher layer has a header appended by the lower layer and becomes one new message. But this is not always the case. Above, we saw a trace in which the web response (one HTTP message comprised of an HTTP header and an HTTP payload) was converted into multiple lower layer messages (being multiple TCP packets). Imagine that you have drawn the packet structure (as in step 2) for the first and last TCP packet carrying the web response. How will the drawings differ?
   第一个 TCP 数据包的负载是 HTTP 协议的头部，而其余 TCP 数据包的负载是 HTTP 的有效载荷。
3. In the classic layered model described above, lower layers append headers to the messages passed down from higher layers. How will this model change if a lower layer adds encryption?
   双方需要协商加密算法，然后在传输数据时，接收方需要解密数据，否则无法解析数据包。
4. In the classic layered model described above, lower layers append headers to the messages passed down from higher layers. How will this model change if a lower layer adds compression?
   发送方可以将解压缩方法附在头部，接收方在接收到数据包后，可以解压缩数据包，然后解析数据包。

5 实验结果总结

通过本次实验，我学会了通过 Wireshark 获取各协议层的数据包，掌握了协议层数据包结构，分析了协议开销，分析了解复用键。对计算机网络中的协议层有了更深的理解。

6 附录

无