

计算机网络-实验1：Wireshark

姓名：翟一航 学号：23020011046

一、 实验目的

了解协议和分层在数据包中的表示方式。它们是构建文本中涵盖的网络的关键概念。

Wireshark：本练习使用 Wireshark 软件工具捕获和检查数据包跟踪。数据包跟踪是网络上某个位置的流量记录，就好像对通过特定线路的所有位进行快照一样。数据包跟踪记录每个数据包的时间戳，以及构成数据包的位，从较低层标头到较高层内容。

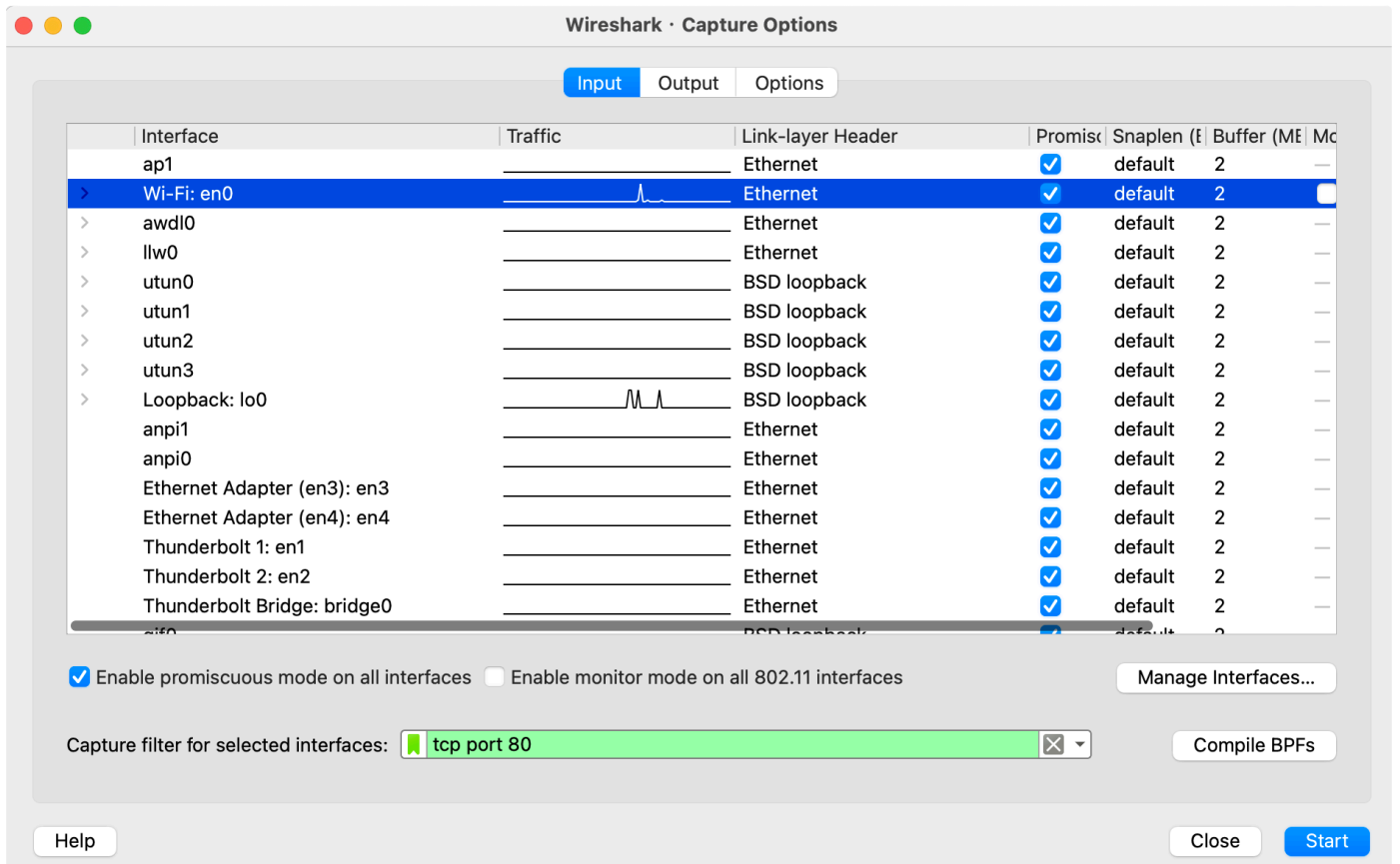
二、 实验环境

环境	说明
操作系统	MacOS
抓包工具	Wireshark
网络接口	Wi-Fi:en0

三、 实验内容

Step1:Capture Traces

首先，在 zsh 终端中通过 homebrew 工具下载 Wireshark 并启动软件，进入设置界面选择 Wi-Fi:en0 接口，并使用 filter—tcp port 80，开始运行。



接着，打开一个使用 https 进行加密的网址进行测试（此处测试中国海洋大学学术期刊网），结果如下：

Capturing from Wi-Fi: en0 (tcp port 80)

Apply a display filter ... <=>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	240c:ca02:2140:72a...	2402:4e00:1020:241...	TCP	98	62574 → 80 [SYN, ECE, CWR] Seq=0 Win=655
2	0.059527	2402:4e00:1020:241...	240c:ca02:2140:72a...	TCP	86	80 → 62574 [SYN, ACK, ECE] Seq=0 Ack=1 W
3	0.059786	240c:ca02:2140:72a...	2402:4e00:1020:241...	TCP	74	62574 → 80 [ACK] Seq=1 Ack=1 Win=262144
4	0.060973	240c:ca02:2140:72a...	2402:4e00:1020:241...	HTTP	827	POST /mmtls/335039cd HTTP/1.1
5	0.120838	2402:4e00:1020:241...	240c:ca02:2140:72a...	TCP	74	80 → 62574 [ACK] Seq=1 Ack=754 Win=64256
6	0.136718	2402:4e00:1020:241...	240c:ca02:2140:72a...	HTTP	421	HTTP/1.1 200 OK
7	0.136722	2402:4e00:1020:241...	240c:ca02:2140:72a...	TCP	74	80 → 62574 [FIN, ACK] Seq=348 Ack=754 Wi
8	0.136908	240c:ca02:2140:72a...	2402:4e00:1020:241...	TCP	74	62574 → 80 [ACK] Seq=754 Ack=348 Win=261
9	0.136999	240c:ca02:2140:72a...	2402:4e00:1020:241...	TCP	74	62574 → 80 [ACK] Seq=754 Ack=349 Win=261
10	0.138310	240c:ca02:2140:72a...	2402:4e00:1020:241...	TCP	74	62574 → 80 [FIN, ACK] Seq=754 Ack=349 Wi
11	0.211753	2402:4e00:1020:241...	240c:ca02:2140:72a...	TCP	74	80 → 62574 [RST] Seq=349 Win=0 Len=0
12	61.999698	10.140.223.186	23.221.197.78	TCP	78	49723 → 80 [SYN] Seq=0 Win=65535 Len=0 M
13	62.107575	23.221.197.78	10.140.223.186	TCP	74	80 → 49723 [SYN, ACK] Seq=0 Ack=1 Win=65
14	62.108078	10.140.223.186	23.221.197.78	TCP	66	49723 → 80 [ACK] Seq=1 Ack=1 Win=131776
15	62.108397	10.140.223.186	23.221.197.78	HTTP	311	GET /hotspot-detect.html HTTP/1.1
16	62.222131	23.221.197.78	10.140.223.186	TCP	66	80 → 49723 [ACK] Seq=1 Ack=246 Win=65024
17	62.222133	23.221.197.78	10.140.223.186	HTTP	259	HTTP/1.1 200 OK (text/html)
18	62.226552	10.140.223.186	23.221.197.78	TCP	66	49723 → 80 [ACK] Seq=246 Ack=194 Win=131

> Frame 15: Packet, 311 bytes on wire (2488 bits), 311 bytes
 > Ethernet II, Src: da:fb:99:03:af:09 (da:fb:99:03:af:09), Ds
 > Internet Protocol Version 4, Src: 10.140.223.186, Dst: 23.2
 > Transmission Control Protocol, Src Port: 49723, Dst Port: 8
 > Hypertext Transfer Protocol

```

0000  00 00 5e 00 01 01 da fb 99 03 af 09 08 00 45 00
0010  01 29 00 00 40 00 40 06 72 5d 0a 8c df ba 17 dd
0020  c5 4e c2 3b 00 50 23 8c b1 c8 c0 dd d3 c5 80 18
0030  08 0b 1c 02 00 00 01 01 08 0a 62 83 7f 04 4f 33
0040  82 9c 47 45 54 20 2f 68 6f 74 73 70 6f 74 2d 64
0050  65 74 65 63 74 2e 68 74 6d 6c 20 48 54 54 50 2f
0060  31 2e 31 0d 0a 48 6f 73 74 3a 20 63 61 70 74 69
0070  76 65 2e 61 70 70 6c 65 2e 63 6f 6d 0d 0a 43 61
0080  63 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6e 6f 2d
0090  63 61 63 68 65 0d 0a 41 63 63 65 70 74 3a 20 2a
00a0  2f 2a 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20
00b0  43 61 70 74 69 76 65 4e 65 74 77 6f 72 6b 53 75
00c0  70 70 6f 72 74 2d 34 39 31 2e 31 32 30 2e 34 20
00d0  77 69 73 70 72 0d 0a 41 63 63 65 70 74 2d 4c 61
00e0  6e 67 75 61 67 65 3a 20 7a 68 2d 43 4e 2c 7a 68
00f0  2d 48 61 6e 73 3b 71 3d 30 2e 39 0d 0a 41 63 63
0100  65 70 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a
0110  69 70 2c 20 64 65 66 6c 61 74 65 0d 0a 43 6f 6e
0120  6e 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d 61 6c
0130  69 76 65 0d 0a 0d 0a
  
```

wireshark_Wi-FiMZK6D3.pcapng Packets: 21 Profile: Default

Step2:Inspect the Trace

点击一个“协议”为http，而“信息为”GET的数据包，查看其具体信息：

Capturing from Wi-Fi: en0 (tcp port 80)						
http						
No.	Time	Source	Destination	Protocol	Length	Info
4	0.074470	240c:ca02:2140:72a...	2402:4e00:1020:241...	HTTP	827	POST /mmtls/2ff18351 HTTP/1.1
6	0.162719	2402:4e00:1020:241...	240c:ca02:2140:72a...	HTTP	421	HTTP/1.1 200 OK
15	160.363395	10.140.223.186	222.192.187.243	HTTP	409	GET /ME8wTTBLMEkwRzAHBgUrDgMCGgQU0dKLcf4d...
17	160.379625	222.192.187.243	10.140.223.186	OCSP	1051	Response
27	243.378032	10.140.223.186	114.132.174.99	HTTP	928	POST /mmtls/302fd4d6 HTTP/1.1
30	243.389087	240c:ca02:2140:72a...	2402:4e00:1020:241...	HTTP	827	POST /mmtls/302fd4d6 HTTP/1.1
32	243.449612	114.132.174.99	10.140.223.186	HTTP	963	HTTP/1.1 200 OK
38	243.478729	2402:4e00:1020:241...	240c:ca02:2140:72a...	HTTP	421	HTTP/1.1 200 OK
51	263.907326	240c:ca02:2140:72a...	2402:4e00:1620:161...	HTTP	299	POST /mmtls/3034f5e2 HTTP/1.1
55	264.005503	2402:4e00:1620:161...	240c:ca02:2140:72a...	HTTP	389	HTTP/1.1 200 OK
64	278.098668	240c:ca02:2140:72a...	2402:4e00:1620:161...	HTTP	795	POST /mmtls/3038ceab HTTP/1.1
66	278.169333	2402:4e00:1620:161...	240c:ca02:2140:72a...	HTTP	615	HTTP/1.1 200 OK
75	335.030220	240c:ca02:2140:72a...	2402:4e00:1020:241...	HTTP	785	POST /mmtls/30476cda HTTP/1.1
78	335.192443	2402:4e00:1020:241...	240c:ca02:2140:72a...	HTTP	524	HTTP/1.1 200 OK

> Frame 15: Packet, 409 bytes on wire (3272 bits), 409 bytes captured (3272 bits) on interface 0

> Ethernet II, Src: da:fb:99:03:af:09 (da:fb:99:03:af:09), Dst: 02:00:00:00:00:00

> Internet Protocol Version 4, Src: 10.140.223.186, Dst: 222.192.187.243

> Transmission Control Protocol, Src Port: 49552, Dst Port: 80

> Hypertext Transfer Protocol

0000 00 00 5e 00 01 01 da fb 99 03 af 09 08 00 45 00 ...

0010 01 8b 00 00 40 00 40 06 b4 72 0a 8c df ba de c0 ...

0020 bb f3 c1 90 00 50 05 fc 2b a2 f6 bc 1f 38 50 18 ...

0030 10 00 4f d7 00 00 47 45 54 20 2f 4d 45 38 77 54 ...

0040 54 42 4c 4d 45 6b 77 52 7a 41 48 42 67 55 72 44 TBI

0050 67 4d 43 47 67 51 55 4f 64 4b 4c 63 66 34 64 47 gM

0060 62 5a 66 73 25 32 46 45 6f 6a 79 4f 38 42 46 6c bZ

0070 63 51 35 55 45 46 45 34 69 56 43 41 59 6c 65 62 cQ

0080 6a 62 75 59 50 25 32 42 76 71 35 45 75 30 47 46 jbu

0090 34 38 35 41 68 41 4f 45 67 66 79 4e 43 74 49 4a 48

00a0 76 30 70 71 57 4c 57 38 7a 77 45 20 48 54 54 50 v0

00b0 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6f 63 73 70 /1

00c0 2e 64 69 67 69 63 65 72 74 2e 63 6e 0d 0a 58 2d .d

00d0 41 70 70 6c 65 2d 52 65 71 75 65 73 74 2d 55 55 Ap

00e0 49 44 3a 20 37 35 38 35 34 45 36 33 2d 42 41 32 ID

00f0 42 2d 34 42 39 46 2d 42 34 31 39 2d 42 42 33 35 B-

0100 34 46 41 41 31 39 45 34 0d 0a 41 63 63 65 70 74 4F

0110 3a 20 2a 2f 2a 0d 0a 55 73 65 72 2d 41 67 65 6e :>

0120 74 3a 20 63 6f 6d 2e 61 70 70 6c 65 2e 74 72 75 t:

0130 73 74 64 2f 33 2e 30 0d 0a 41 63 63 65 70 74 2d st

0140 4c 61 6e 67 75 61 67 65 3a 20 7a 68 2d 43 4e 2c Lar

0150 7a 68 2d 48 61 6e 73 3b 71 3d 30 2e 39 0d 0a 41 zh-

wireshark_Wi-FIRXCGE3.pcapngPackets: 83 · Displayed: 14 (16.9%)Profile: Default

发现数据包标头详细信息窗口显示了该网络数据包对应 TCP/IP 协议的五层协议栈，它们从底层到高层依次为：

Wireshark block	对应网络协议层	包含信息/作用
Frame	非协议层	Wireshark的捕获元数据，包含整个数据帧的信息
Ethernet II	网络接口层	定义如何在本地网络传输信息
IPv4	网络层	将数据包在网络间路由
TCP	传输层	提供可靠数据传输
HTTP	应用层	为用户提供网络服务

Step3:Packet Structure

单击对应数据包中不同层的数据块可以查看其大小，绘制出 HTTP GET 数据包的结构图如下：

Ethernet II Header	14 bytes
IPv4 Header	20 bytes
TCP Header	32 bytes
HTTP Data	245 bytes

可以发现 HTTP 层对应的数据明显大于其余层的头部，这是因为显示的是 HTTP 的信息——主要显示头部，但是如果存在且可识别，那么也会显示正文，而其余各层均只显示头部信息。

Step4:Demultiplexing Keys

Q1:哪个以太网标头字段是解复用密钥，告诉它下一个更高层是 IP? 此字段中使用什么值来表示IP?

答:

在 Ethernet II 帧格式中，Type 字段是解复用密钥，这个字段使用 0x0800 来指示 IP。

```

✓ Ethernet II, Src: da:fb:99:03:af:09 (da:fb:99:03:af:09), Ds
  ✓ Destination: IETF-VRRP-VRID_01 (00:00:5e:00:01:01)
    .... ..0. .... = LG bit: Globally unique
    .... ..0 .... = IG bit: Individual address
  ✓ Source: da:fb:99:03:af:09 (da:fb:99:03:af:09)
    .... ..1. .... = LG bit: Locally administered
    .... ..0 .... = IG bit: Individual address
Type: IPv4 (0x0800)
[Stream index: 0]

```

Q2:哪个 IP 标头字段是告诉它下一个更高层是 TCP 的解复用密钥? 此字段中使用什么值来表示TCP?

答:

在 IP 标头中，Protocol 字段是解复用密钥，这个字段使用 6 来表示 TCP。

```
✓ Internet Protocol Version 4, Src: 10.140.223.186, Dst: 23.2
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not
  Total Length: 297
  Identification: 0x0000 (0)
  > 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 64
  Protocol: TCP (6)
  Header Checksum: 0x725d [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.140.223.186
  Destination Address: 23.221.197.78
  [Stream index: 0]
```

四、 总结建议

通过此次实验，我对计算机网络协议的分层和数据包的结构有了更深刻的理解。在实验中，通过使用 Wireshark 抓取并分析 HTTP 数据包，了解了 TCP/IP 协议各层的功能和解复用机制。

通过实验使用过滤器 `tcp port 80`，我了解到 80 是 HTTP 协议的标准端口（HTTPS 协议的标准端口是 443），也学习到过滤器的作用——可以过滤掉大量的其他网络流量，专注分析目标协议，排除干扰。

此外，通过查阅资料，我弄清楚了 Frame 与 Ethernet II 的实际关系。二者并非同一层，Frame 本质是 Wireshark 的元数据记录，它是包含以太网络层的容器，只在 Wireshark 中存在，而 Ethernet II 却是真正的数据链路层协议，实际在网络线路上传输。