厦門大學

信息学院软件工程系

《计算机网络》实验报告

题　　目 实验三　基于 PCAP 库侦听并分析网络流量

班　　级 软件工程 2021 级卓越班

姓　　名 王明皓

学　　号 37220222203769

实验时间 2023 年 4 月 18 日

2023 年 4 月日

填写说明

1、本文件为 Word 模板文件，建议使用 Microsoft Word 2021 打开，在可填写的区域中如实填写；

2、填表时勿改变字体字号，保持排版工整，打印为 PDF 文件提交；

3、文件总大小尽量控制在 1MB 以下，最大勿超过 5MB；

4、应将材料清单上传在代码托管平台上；

5、在实验课结束 14 天内，按原文件发送至课程 FTP 指定位置。

1　实验目的

通过完成实验，理解数据链路层、网络层、传输层和应用层的基本原理。掌握用 Wireshark 观察网络流量并辅助网络侦听相关的编程；掌握用 Libpcap 或 WinPcap 库侦听并处理以太网帧和 IP 报文的方法；熟悉以太网帧、IP 报文、TCP 段和 FTP 命令的格式概念，掌握 TCP 协议的基本机制；熟悉帧头部或 IP 报文头部各字段的含义。熟悉 TCP 段和 FTP 数据协议的概念，熟悉段头部各字段和 FTP 控制命令的指令和数据的含义。

2　实验环境

操作系统：Windows11

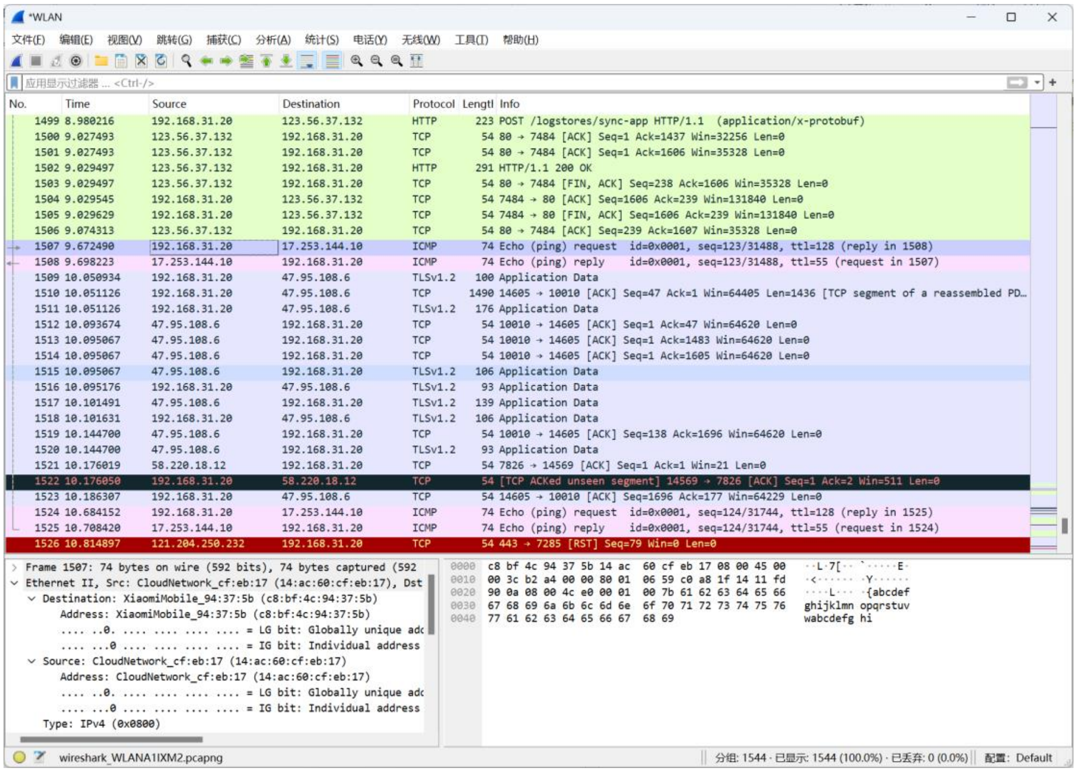
使用软件：Wireshark4.2.4

编程语言等。

3　实验结果

1、用侦听解析软件观察数据格式

用 Wireshark 或 Omnipeek 等网络侦听软件网络上的数据流，验证理论课讲授的网络协议层次嵌套



*WLAN
文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)
应用显示过滤器 … <Ctrl-/>

```
No.   Time      Source           Destination     Protocol Lengtl Info
 1499 8.980216  192.168.31.20    123.56.37.132   HTTP     223 POST /logstores/sync-app HTTP/1.1  (application/x-protobuf)
 1500 9.027493  123.56.37.132    192.168.31.20   TCP       54 80 → 7484 [ACK] Seq=1 Ack=1437 Win=32256 Len=0
 1501 9.027493  123.56.37.132    192.168.31.20   TCP       54 80 → 7484 [ACK] Seq=1 Ack=1606 Win=35328 Len=0
 1502 9.029497  123.56.37.132    192.168.31.20   HTTP     291 HTTP/1.1 200 OK
 1503 9.029497  123.56.37.132    192.168.31.20   TCP       54 80 → 7484 [FIN, ACK] Seq=238 Ack=1606 Win=35328 Len=0
 1504 9.029545  192.168.31.20    123.56.37.132   TCP       54 7484 → 80 [ACK] Seq=1606 Ack=239 Win=131840 Len=0
 1505 9.029629  192.168.31.20    123.56.37.132   TCP       54 7484 → 80 [FIN, ACK] Seq=1606 Ack=239 Win=131840 Len=0
 1506 9.074313  123.56.37.132    192.168.31.20   TCP       54 80 → 7484 [ACK] Seq=239 Ack=1607 Win=35328 Len=0
 1507 9.672490  192.168.31.20    17.253.144.10   ICMP      74 Echo (ping) request  id=0x0001, seq=123/31488, ttl=128 (reply in 1508)
 1508 9.698223  17.253.144.10    192.168.31.20   ICMP      74 Echo (ping) reply    id=0x0001, seq=123/31488, ttl=55 (request in 1507)
 1509 10.050934 192.168.31.20    47.95.108.6     TLSv1.2  100 Application Data
 1510 10.051126 192.168.31.20    47.95.108.6     TCP     1490 14605 → 10010 [ACK] Seq=47 Ack=1 Win=64405 Len=1436 [TCP segment of a reassembled PD…
 1511 10.051126 192.168.31.20    47.95.108.6     TLSv1.2  176 Application Data
 1512 10.093674 47.95.108.6      192.168.31.20   TCP       54 10010 → 14605 [ACK] Seq=1 Ack=47 Win=64620 Len=0
 1513 10.095067 47.95.108.6      192.168.31.20   TCP       54 10010 → 14605 [ACK] Seq=1 Ack=1483 Win=64620 Len=0
 1514 10.095067 47.95.108.6      192.168.31.20   TCP       54 10010 → 14605 [ACK] Seq=1 Ack=1605 Win=64620 Len=0
 1515 10.095067 47.95.108.6      192.168.31.20   TLSv1.2  106 Application Data
 1516 10.095176 192.168.31.20    47.95.108.6     TLSv1.2   93 Application Data
 1517 10.101491 47.95.108.6      192.168.31.20   TLSv1.2  139 Application Data
 1518 10.101631 192.168.31.20    47.95.108.6     TLSv1.2  106 Application Data
 1519 10.144700 47.95.108.6      192.168.31.20   TCP       54 10010 → 14605 [ACK] Seq=138 Ack=1696 Win=64620 Len=0
 1520 10.144700 47.95.108.6      192.168.31.20   TLSv1.2   93 Application Data
 1521 10.176019 58.220.18.12     192.168.31.20   TCP       54 7826 → 14569 [ACK] Seq=1 Ack=1 Win=21 Len=0
 1522 10.176050 192.168.31.20    58.220.18.12    TCP       54 [TCP ACKed unseen segment] 14569 → 7826 [ACK] Seq=1 Ack=2 Win=511 Len=0
 1523 10.186307 192.168.31.20    47.95.108.6     TCP       54 14605 → 10010 [ACK] Seq=1696 Ack=177 Win=64229 Len=0
 1524 10.684152 192.168.31.20    17.253.144.10   ICMP      74 Echo (ping) request  id=0x0001, seq=124/31744, ttl=128 (reply in 1525)
 1525 10.708420 17.253.144.10    192.168.31.20   ICMP      74 Echo (ping) reply    id=0x0001, seq=124/31744, ttl=55 (request in 1524)
 1526 10.814897 121.204.250.232  192.168.31.20   TCP       54 443 → 7285 [RST] Seq=79 Win=0 Len=0
```

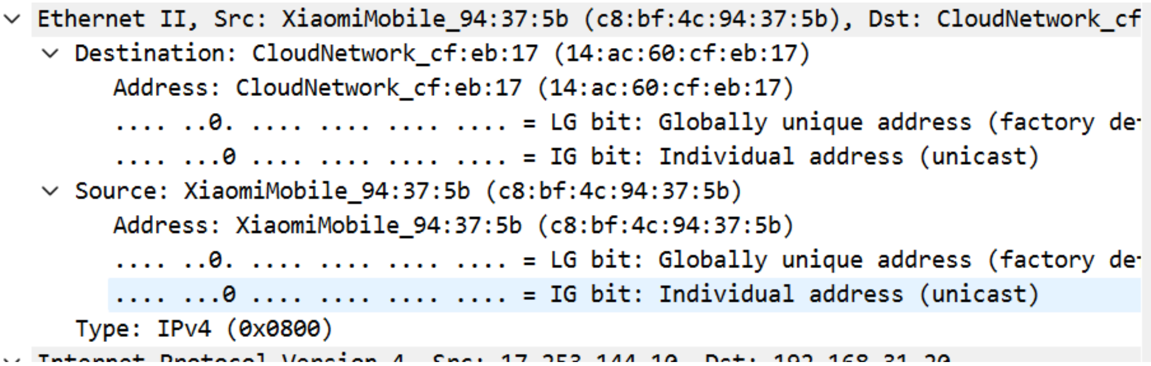
```
> Frame 1507: 74 bytes on wire (592 bits), 74 bytes captured (592
v Ethernet II, Src: CloudNetwork_cf:eb:17 (14:ac:60:cf:eb:17), Dst
  v Destination: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b)
      Address: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b)
      .... ..0. .... .... .... .... = LG bit: Globally unique adc
      .... ...0 .... .... .... .... = IG bit: Individual address
  v Source: CloudNetwork_cf:eb:17 (14:ac:60:cf:eb:17)
      Address: CloudNetwork_cf:eb:17 (14:ac:60:cf:eb:17)
      .... ..0. .... .... .... .... = LG bit: Globally unique adc
      .... ...0 .... .... .... .... = IG bit: Individual address
    Type: IPv4 (0x0800)
```

```
0000  c8 bf 4c 94 37 5b 14 ac  60 cf eb 17 08 00 45 00   ··L·7[·· `·····E·
0010  00 3c b2 a4 00 00 80 01  06 59 c0 a8 1f 14 11 fd   ·<······ ·Y······
0020  90 0a 08 00 4c e0 00 01  00 7b 61 62 63 64 65 66   ····L··· ·{abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

wireshark_WLANA1IXM2.pcapng
分组: 1544 · 已显示: 1544 (100.0%) · 已丢弃: 0 (0.0%)
配置: Default

验证帧格式

由源地址、目的地址、数据类型组成



```
v Ethernet II, Src: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b), Dst: CloudNetwork_cf
  v Destination: CloudNetwork_cf:eb:17 (14:ac:60:cf:eb:17)
      Address: CloudNetwork_cf:eb:17 (14:ac:60:cf:eb:17)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory de
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  v Source: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b)
      Address: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory de
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 17.253.144.10, Dst: 192.168.31.20
```

IP 报文格式

版本号：4

头文件长度：20 bytes

区分服务字段

总长度：60

标识：0xb04f

标志：0

偏移量：0

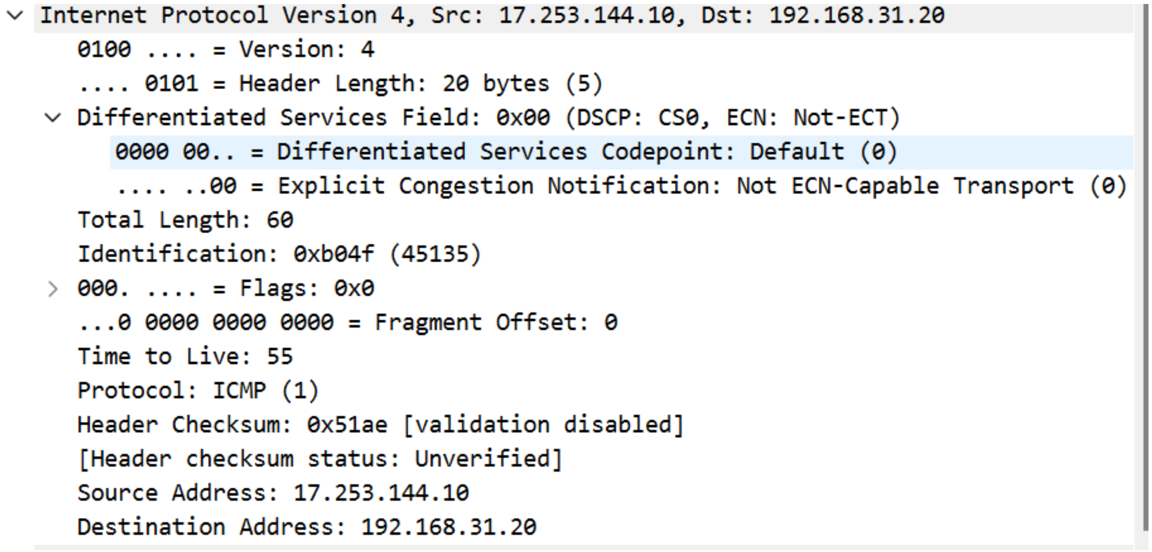
生存时间：55

上层协议：ICMP

头文件校验和：0x51ae

源地址：17.253.144.10

目的地址：192.168.31.20



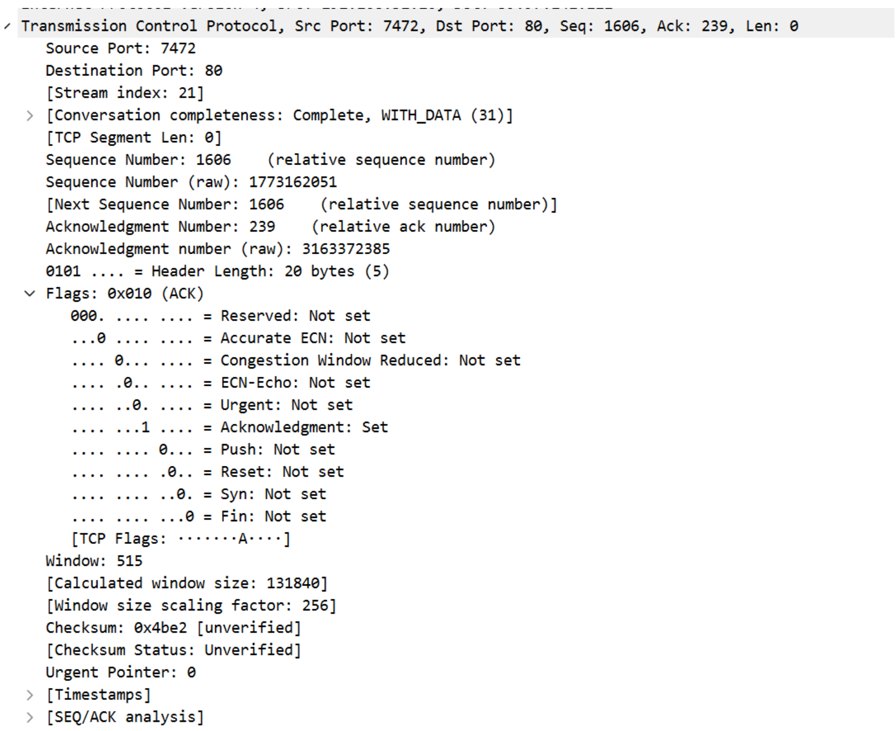
```
∨ Internet Protocol Version 4, Src: 17.253.144.10, Dst: 192.168.31.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ∨ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      0000 00.. = Differentiated Services Codepoint: Default (0)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 60
    Identification: 0xb04f (45135)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 55
    Protocol: ICMP (1)
    Header Checksum: 0x51ae [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 17.253.144.10
    Destination Address: 192.168.31.20
```

TCP 段格式

1. **源端口 Source Port: 7472**
   - 表示报文的来源端口号。
2. **目的端口 Destination Port: 80**
   - 表示报文的目的端口号，这里是 80 端口，通常用于 HTTP。
3. **Stream index: 21**
   - 流索引，可能是用于识别某个流的编号。
4. **Conversation completeness: Complete, WITH_DATA (31)**
   - 对话完成度，完整的对话，并且包含数据。
5. **TCP Segment Len: 0**

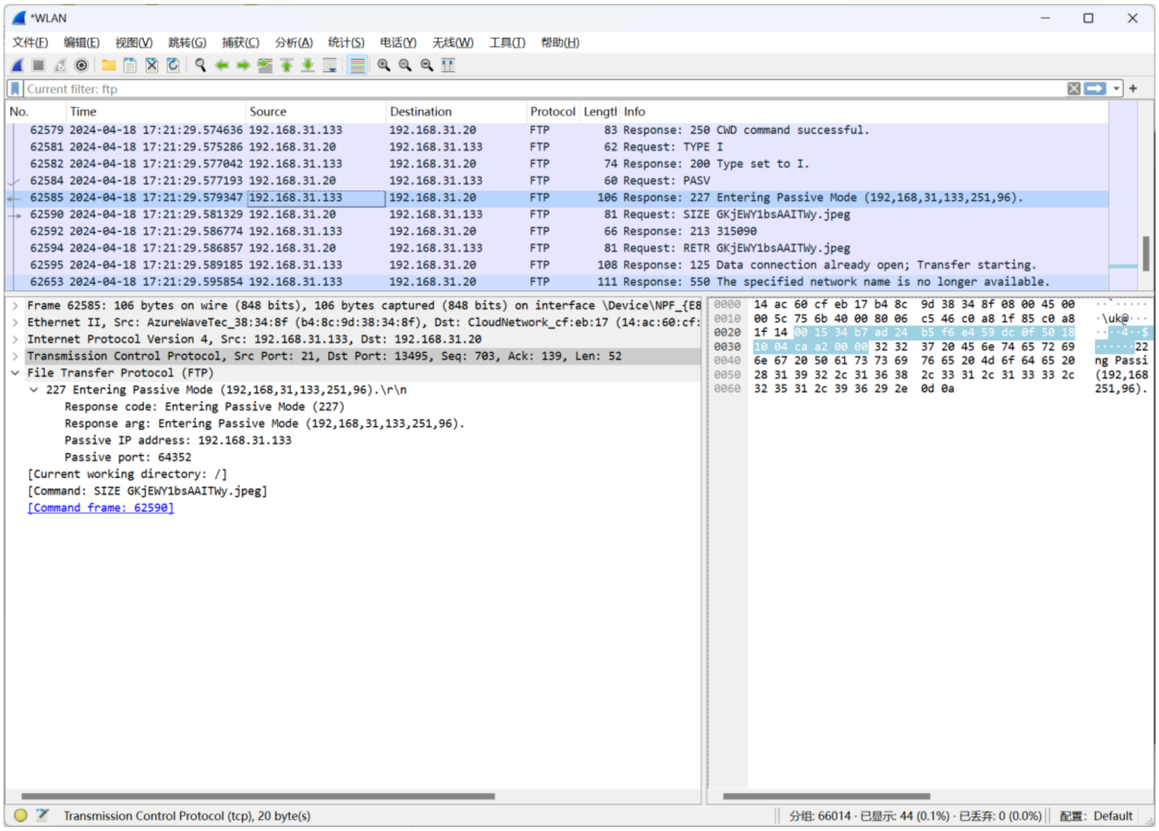
   - TCP 段的长度为 0，即没有有效载荷数据。

6. **Sequence Number: 1606 (relative sequence number)**
   - 表示相对序列号为 1606，用于数据包的顺序重组。
7. **Sequence Number (raw): 1773162051**
   - 原始序列号，以字节表示。
8. **Next Sequence Number: 1606 (relative sequence number)**
   - 下一个期望的序列号。
9. **Acknowledgment Number: 239 (relative ack number)**
   - 表示确认号，即期望收到的下一个序列号。
10. **Acknowledgment number (raw): 31633723850101**
    - 原始确认号。
11. **Header Length: 20 bytes (5)**
    - 报头长度为 20 字节。
12. **Flags: Ox010 (ACK)**
    - 标志位：ACK，表示这是一个确认报文。
13. **Window: 515**
    - 窗口大小为 515 字节，用于流量控制。
14. **Calculated window size: 131840**
    - 计算得到的窗口大小。
15. **Window size scaling factor: 256**
    - 窗口大小的缩放因子。
16. **Checksum: 0x4be2 [unverified]**
    - 校验和，用于检查数据包在传输过程中是否被损坏。
17. **Checksum Status: Unverified**
    - 校验和的状态是未验证的。
18. **Urgent Pointer: 0**
    - 紧急指针，用于指示紧急数据的位置。
19. **Timestamps**
    - 时间戳，可能用于测量报文的传输延迟等。
20. **SEQ/ACK analysis**
    - 序列号和确认号的分析。



```
⌄ Transmission Control Protocol, Src Port: 7472, Dst Port: 80, Seq: 1606, Ack: 239, Len: 0
    Source Port: 7472
    Destination Port: 80
    [Stream index: 21]
  > [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 1606    (relative sequence number)
    Sequence Number (raw): 1773162051
    [Next Sequence Number: 1606    (relative sequence number)]
    Acknowledgment Number: 239    (relative ack number)
    Acknowledgment number (raw): 3163372385
    0101 .... = Header Length: 20 bytes (5)
  ⌄ Flags: 0x010 (ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Accurate ECN: Not set
      .... 0... .... = Congestion Window Reduced: Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
      .... .... ...0 = Fin: Not set
      [TCP Flags: ·······A····]
    Window: 515
    [Calculated window size: 131840]
    [Window size scaling factor: 256]
    Checksum: 0x4be2 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
```

FTP 协议命令和响应的格式


*WLAN
文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)
Current filter: ftp

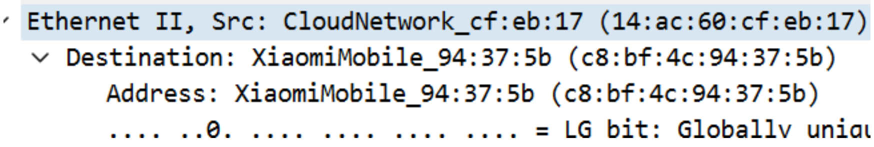
```
No.    Time                       Source          Destination     Protocol Lengtl Info
62579 2024-04-18 17:21:29.574636 192.168.31.133  192.168.31.20   FTP       83 Response: 250 CWD command successful.
62581 2024-04-18 17:21:29.575286 192.168.31.20   192.168.31.133  FTP       62 Request: TYPE I
62582 2024-04-18 17:21:29.577042 192.168.31.133  192.168.31.20   FTP       74 Response: 200 Type set to I.
62584 2024-04-18 17:21:29.577193 192.168.31.20   192.168.31.133  FTP       60 Request: PASV
62585 2024-04-18 17:21:29.579347 192.168.31.133  192.168.31.20   FTP      106 Response: 227 Entering Passive Mode (192,168,31,133,251,96).
62590 2024-04-18 17:21:29.581329 192.168.31.20   192.168.31.133  FTP       81 Request: SIZE GKjEWY1bsAAITWy.jpeg
62592 2024-04-18 17:21:29.586774 192.168.31.133  192.168.31.20   FTP       66 Response: 213 315090
62594 2024-04-18 17:21:29.586857 192.168.31.20   192.168.31.133  FTP       81 Request: RETR GKjEWY1bsAAITWy.jpeg
62595 2024-04-18 17:21:29.589185 192.168.31.133  192.168.31.20   FTP      108 Response: 125 Data connection already open; Transfer starting.
62653 2024-04-18 17:21:29.595854 192.168.31.133  192.168.31.20   FTP      111 Response: 550 The specified network name is no longer available.
```

```
> Frame 62585: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface \Device\NPF_{E8
> Ethernet II, Src: AzureWaveTec_38:34:8f (b4:8c:9d:38:34:8f), Dst: CloudNetwork_cf:eb:17 (14:ac:60:cf:
> Internet Protocol Version 4, Src: 192.168.31.133, Dst: 192.168.31.20
> Transmission Control Protocol, Src Port: 21, Dst Port: 13495, Seq: 703, Ack: 139, Len: 52
⌄ File Transfer Protocol (FTP)
  ⌄ 227 Entering Passive Mode (192,168,31,133,251,96).\r\n
        Response code: Entering Passive Mode (227)
        Response arg: Entering Passive Mode (192,168,31,133,251,96).
        Passive IP address: 192.168.31.133
        Passive port: 64352
  [Current working directory: /]
  [Command: SIZE GKjEWY1bsAAITWy.jpeg]
  [Command frame: 62590]
```

```
0000  14 ac 60 cf eb 17 b4 8c  9d 38 34 8f 08 00 45 00   ··`·····
0010  00 5c 75 6b 40 00 80 06  c5 46 c0 a8 1f 85 c0 a8   ·\uk@···
0020  1f 14 00 15 34 b7 ad 24  b5 f6 e4 59 dc 0f 50 18   ····4··$
0030  10 04 ca a2 00 00 32 32  37 20 45 6e 74 65 72 69   ······22
0040  6e 67 20 50 61 73 73 69  76 65 20 4d 6f 64 65 20   ng Passi
0050  28 31 39 32 2c 31 36 38  2c 33 31 2c 31 33 33 2c   (192,168
0060  32 35 31 2c 39 36 29 2e  0d 0a                     251,96).
```

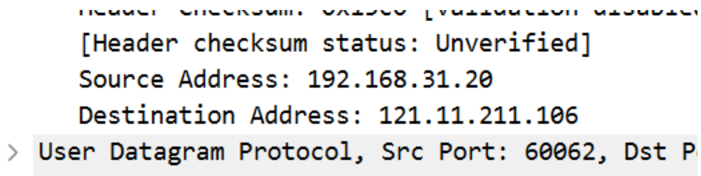
Transmission Control Protocol (tcp), 20 byte(s)   分组: 66014 · 已显示: 44 (0.1%) · 已丢弃: 0 (0.0%)   配置: Default

验证 MAC 地址



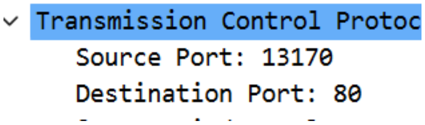
```
Ethernet II, Src: CloudNetwork_cf:eb:17 (14:ac:60:cf:eb:17)
  Destination: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b)
      Address: XiaomiMobile_94:37:5b (c8:bf:4c:94:37:5b)
      .... ..0. .... .... .... .... = LG bit: Globally uniq
```

验证 IP 地址



```
    [Header checksum status: Unverified]
    Source Address: 192.168.31.20
    Destination Address: 121.11.211.106
> User Datagram Protocol, Src Port: 60062, Dst P
```

验证 TCP 端口等协议地址格式



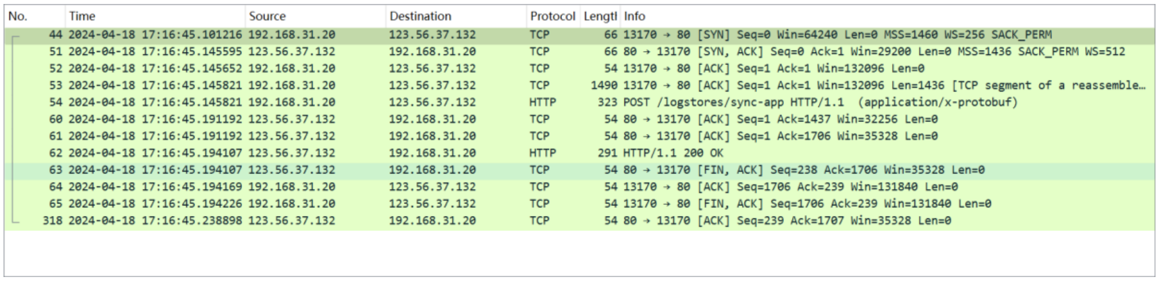
```
v Transmission Control Protoc
      Source Port: 13170
      Destination Port: 80
```

2、用侦听解析软件观察 TCP 机制

用 Wireshark 侦听并观察 TCP 数据段。

观察其建立和撤除连接的过程，观察段 ID、窗口机制和拥塞控制机制等。将该过程截图在报告中。

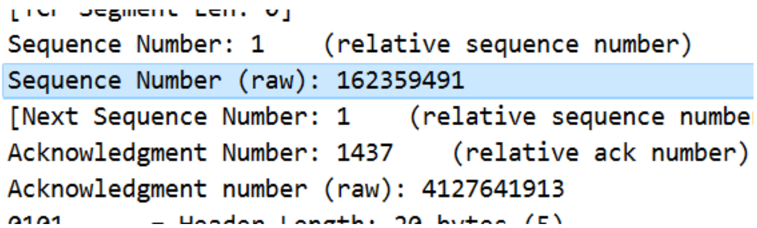
以下为其建立连接和撤出连接的过程：



No.	Time	Source	Destination	Protocol	Lengtl	Info
44	2024-04-18 17:16:45.101216	192.168.31.20	123.56.37.132	TCP	66	13170 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
51	2024-04-18 17:16:45.145595	123.56.37.132	192.168.31.20	TCP	66	80 → 13170 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1436 SACK_PERM WS=512
52	2024-04-18 17:16:45.145652	192.168.31.20	123.56.37.132	TCP	54	13170 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=0
53	2024-04-18 17:16:45.145821	192.168.31.20	123.56.37.132	TCP	1490	13170 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=1436 [TCP segment of a reassemble…
54	2024-04-18 17:16:45.145821	192.168.31.20	123.56.37.132	HTTP	323	POST /logstores/sync-app HTTP/1.1 (application/x-protobuf)
60	2024-04-18 17:16:45.191192	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [ACK] Seq=1 Ack=1437 Win=32256 Len=0
61	2024-04-18 17:16:45.191192	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [ACK] Seq=1 Ack=1706 Win=35328 Len=0
62	2024-04-18 17:16:45.194107	123.56.37.132	192.168.31.20	HTTP	291	HTTP/1.1 200 OK
63	2024-04-18 17:16:45.194107	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [FIN, ACK] Seq=238 Ack=1706 Win=35328 Len=0
64	2024-04-18 17:16:45.194169	192.168.31.20	123.56.37.132	TCP	54	13170 → 80 [ACK] Seq=1706 Ack=239 Win=131840 Len=0
65	2024-04-18 17:16:45.194226	192.168.31.20	123.56.37.132	TCP	54	13170 → 80 [FIN, ACK] Seq=1706 Ack=239 Win=131840 Len=0
318	2024-04-18 17:16:45.238898	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [ACK] Seq=239 Ack=1707 Win=35328 Len=0

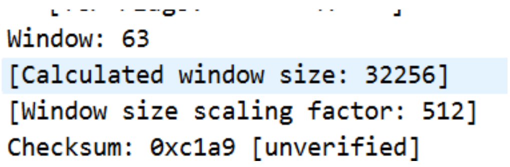
首先 TCP 机制会利用三次握手（客户端发起连接请求（SYN）、服务器确认连接请求（SYN-ACK）、客户端确认连接（ACK））来建立连接

段 ID：每个 TCP 段都有唯一的序列号



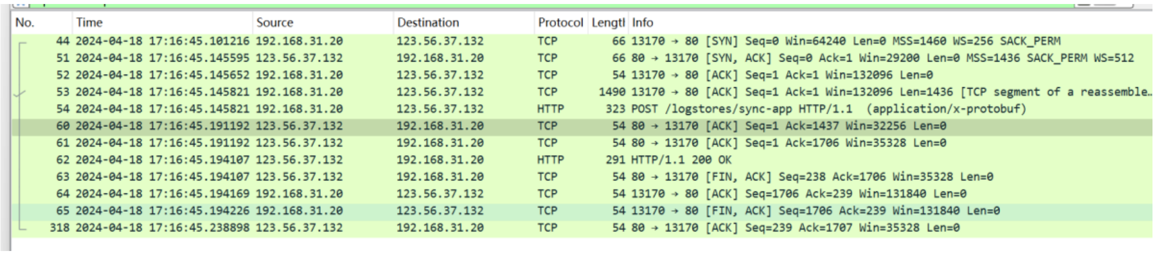
```
[TCP Segment Len: 0]
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 162359491
[Next Sequence Number: 1    (relative sequence numbe
Acknowledgment Number: 1437    (relative ack number)
Acknowledgment number (raw): 4127641913
0101 .... = Header Length: 20 bytes (5)
```

窗口机制：在 TCP 连接建立时，发送方和接收方会协商一个窗口大小。窗口大小表示接收方愿意接受的数据量，发送方根据这个窗口大小来发送数据。



```
Window: 63
[Calculated window size: 32256]
[Window size scaling factor: 512]
Checksum: 0xc1a9 [unverified]
```

拥塞控制机制：TCP 通过一系列的算法来调整数据发送速率，以避免网络拥塞和数据丢失



No.	Time	Source	Destination	Protocol	Lengtl	Info
44	2024-04-18 17:16:45.101216	192.168.31.20	123.56.37.132	TCP	66	13170 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
51	2024-04-18 17:16:45.145595	123.56.37.132	192.168.31.20	TCP	66	80 → 13170 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1436 SACK_PERM WS=512
52	2024-04-18 17:16:45.145652	192.168.31.20	123.56.37.132	TCP	54	13170 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=0
53	2024-04-18 17:16:45.145821	192.168.31.20	123.56.37.132	TCP	1490	13170 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=1436 [TCP segment of a reassemble…
54	2024-04-18 17:16:45.145821	192.168.31.20	123.56.37.132	HTTP	323	POST /logstores/sync-app HTTP/1.1 (application/x-protobuf)
60	2024-04-18 17:16:45.191192	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [ACK] Seq=1 Ack=1437 Win=32256 Len=0
61	2024-04-18 17:16:45.191192	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [ACK] Seq=1 Ack=1706 Win=35328 Len=0
62	2024-04-18 17:16:45.194107	123.56.37.132	192.168.31.20	HTTP	291	HTTP/1.1 200 OK
63	2024-04-18 17:16:45.194107	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [FIN, ACK] Seq=238 Ack=1706 Win=35328 Len=0
64	2024-04-18 17:16:45.194169	192.168.31.20	123.56.37.132	TCP	54	13170 → 80 [ACK] Seq=1706 Ack=239 Win=131840 Len=0
65	2024-04-18 17:16:45.194226	192.168.31.20	123.56.37.132	TCP	54	13170 → 80 [FIN, ACK] Seq=1706 Ack=239 Win=131840 Len=0
318	2024-04-18 17:16:45.238898	123.56.37.132	192.168.31.20	TCP	54	80 → 13170 [ACK] Seq=239 Ack=1707 Win=35328 Len=0

3、用 Libpcap 或 WinPcap 库侦听网络数据

部分关键代码：

利用 WinPcap 进行监听：

```
pcap_loop(adhandle, 0, packet_handler, NULL);
```

获取报文头：

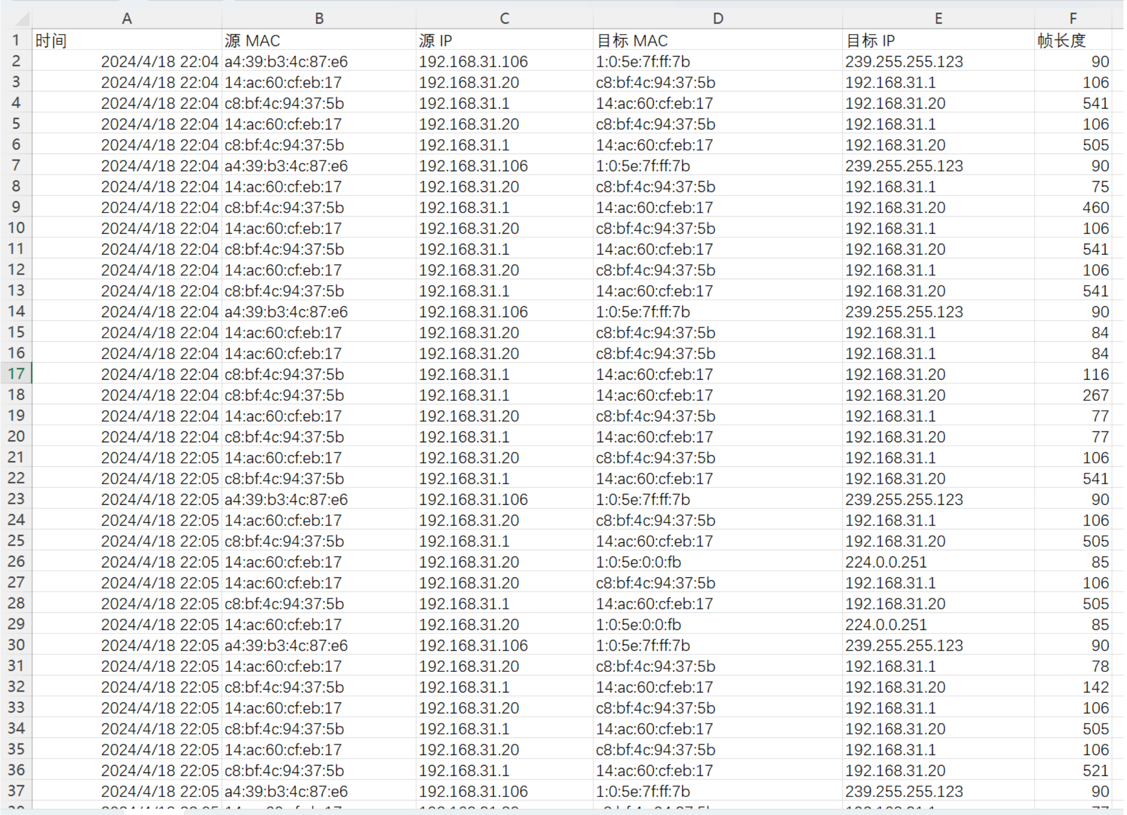
```
ih = (ip_header*)(pkt_data +
    14); //length of ethernet header


mh = (mac_header*)(pkt_data);
```

/* retireve the position of the udp header */

ip_len = (ih->ver_ihl & 0xf) * 4;

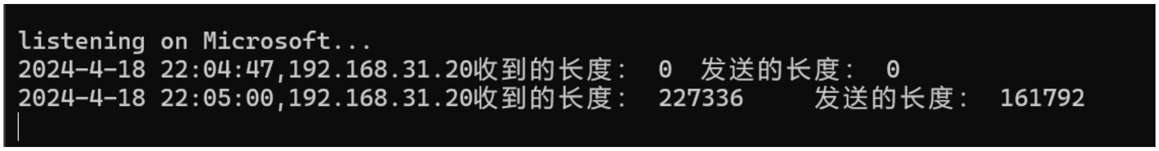
uh = (udp_header*)((u_char*)ih + ip_len);

利用 WinPcap 库侦听到的网络数据如下：



	A	B	C	D	E	F
1	时间	源 MAC	源 IP	目标 MAC	目标 IP	帧长度
2	2024/4/18 22:04	a4:39:b3:4c:87:e6	192.168.31.106	1:0:5e:7f:ff:7b	239.255.255.123	90
3	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
4	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	541
5	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
6	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	505
7	2024/4/18 22:04	a4:39:b3:4c:87:e6	192.168.31.106	1:0:5e:7f:ff:7b	239.255.255.123	90
8	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	75
9	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	460
10	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
11	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	541
12	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
13	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	541
14	2024/4/18 22:04	a4:39:b3:4c:87:e6	192.168.31.106	1:0:5e:7f:ff:7b	239.255.255.123	90
15	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	84
16	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	84
17	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	116
18	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	267
19	2024/4/18 22:04	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	77
20	2024/4/18 22:04	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	77
21	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
22	2024/4/18 22:05	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	541
23	2024/4/18 22:05	a4:39:b3:4c:87:e6	192.168.31.106	1:0:5e:7f:ff:7b	239.255.255.123	90
24	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
25	2024/4/18 22:05	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	505
26	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	1:0:5e:0:0:fb	224.0.0.251	85
27	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
28	2024/4/18 22:05	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	505
29	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	1:0:5e:0:0:fb	224.0.0.251	85
30	2024/4/18 22:05	a4:39:b3:4c:87:e6	192.168.31.106	1:0:5e:7f:ff:7b	239.255.255.123	90
31	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	78
32	2024/4/18 22:05	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	142
33	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
34	2024/4/18 22:05	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	505
35	2024/4/18 22:05	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	192.168.31.1	106
36	2024/4/18 22:05	c8:bf:4c:94:37:5b	192.168.31.1	14:ac:60:cf:eb:17	192.168.31.20	521
37	2024/4/18 22:05	a4:39:b3:4c:87:e6	192.168.31.106	1:0:5e:7f:ff:7b	239.255.255.123	90

进行统计：



```
listening on Microsoft...
2024-4-18 22:04:47,192.168.31.20收到的长度： 0  发送的长度： 0
2024-4-18 22:05:00,192.168.31.20收到的长度： 227336     发送的长度： 161792
```

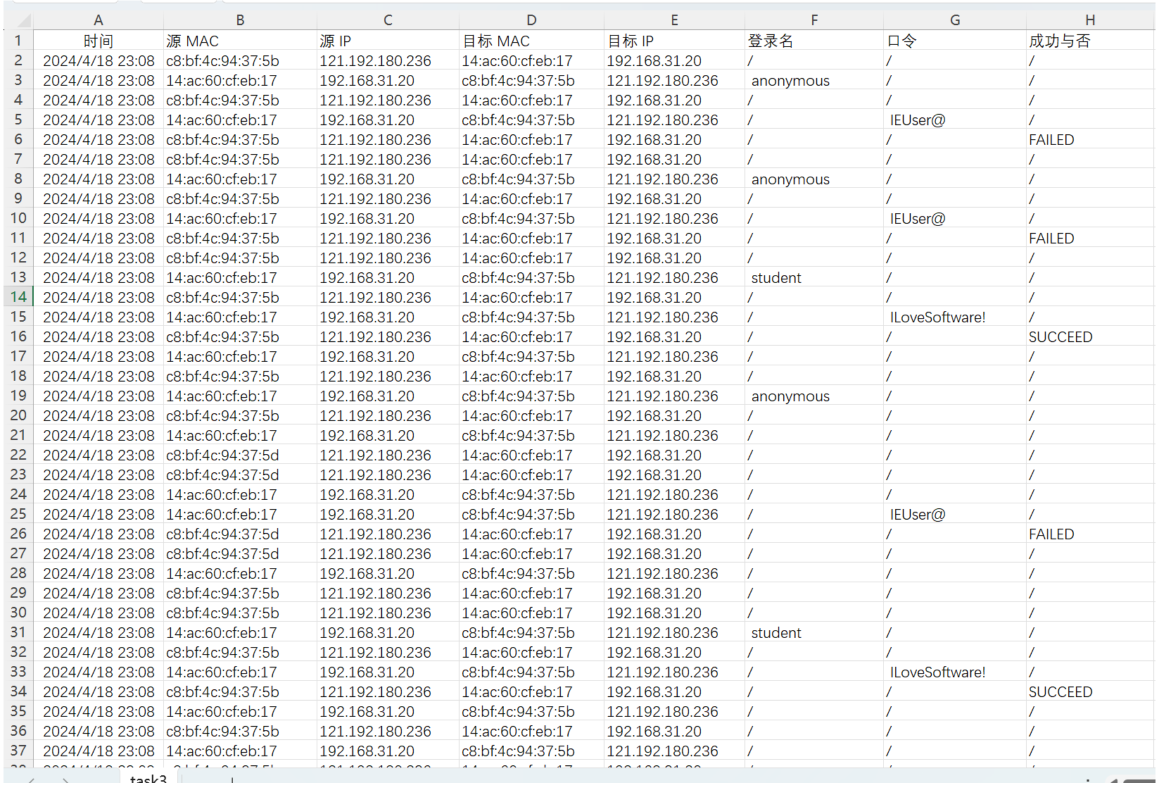
4、解析侦听到的网络数据

实现从数据中提取用户名密码的核心代码：

```
if (*data == 'U' && *(++data) == 'S' && *(++data) == 'E' && *(++data) == 'R') {
    while (*(++data) != 0x0d)    string[i++] = *data;
    string[i] = '\0';
    fprintf(out,"%s,/,/\n", string);
    return;
}

data = (char*)(pkt_data + 54);
if (*data == 'P' && *(++data) == 'A' && *(++data) == 'S' && *(++data) == 'S') {
    while (*(++data) != 0x0d)    string[i++] = *data;
    string[i] = '\0';
    fprintf(out, "/,%s,/\n", string);
    return;
}

data = (char*)(pkt_data + 54);
if (*data == '5' && *(++data) == '3' ) {
    fprintf(out, "/,/,FAILED\n");
    return;
}

data = (char*)(pkt_data + 54);
if (*data == '2' && *(++data) == '3') {
    fprintf(out, "/,/,SUCCEED\n");
    return;
}
```

fprintf(out, "/,/,/\n");

运行结果：



	A	B	C	D	E	F	G	H
1	时间	源 MAC	源 IP	目标 MAC	目标 IP	登录名	口令	成功与否
2	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
3	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	anonymous	/	/
4	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
5	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	IEUser@	/
6	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	FAILED
7	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
8	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	anonymous	/	/
9	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
10	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	IEUser@	/
11	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	FAILED
12	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
13	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	student	/	/
14	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
15	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	ILoveSoftware!	/
16	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	SUCCEED
17	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	/	/
18	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
19	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	anonymous	/	/
20	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
21	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	/	/
22	2024/4/18 23:08	c8:bf:4c:94:37:5d	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
23	2024/4/18 23:08	c8:bf:4c:94:37:5d	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
24	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	/	/
25	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	IEUser@	/
26	2024/4/18 23:08	c8:bf:4c:94:37:5d	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	FAILED
27	2024/4/18 23:08	c8:bf:4c:94:37:5d	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
28	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	/	/
29	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
30	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
31	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	student	/	/
32	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
33	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	ILoveSoftware!	/
34	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	SUCCEED
35	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	/	/
36	2024/4/18 23:08	c8:bf:4c:94:37:5b	121.192.180.236	14:ac:60:cf:eb:17	192.168.31.20	/	/	/
37	2024/4/18 23:08	14:ac:60:cf:eb:17	192.168.31.20	c8:bf:4c:94:37:5b	121.192.180.236	/	/	/

4　实验代码

本次实验的代码已上传于以下代码仓库：https://gitee.com/carribia/cn_exp03

5　实验总结

通过本次实验，我对于计算机网络各分层的相关协议和其格式有了更深刻的理解，明白了文件头不同部位的作用。同时，深入研究了 FTP 协议的数据格式，掌握了其用户名和密码的呈现形式。这既是对前期学到的内容的回顾，又为未来的学习打下了坚定的基础。