

# 廈門大學



## 信息學院軟件工程系

### 《計算機網絡》實驗報告

題 目 實驗三 基於 PCAP 庫偵聽並分析網絡流量

班 級 數字媒體技術 2023 級 1 班

姓 名 張林

學 号 37220232203919

實驗時間 2025 年 10 月 0 日

2025 年 10 月 22 日

# 填写说明

- 1、本文件为 Word 模板文件，建议使用 Microsoft Word 2024 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、在实验课结束 14 天内，按实验报告提交到我校课程网站的指定位置，源代码等主要材料上传在公开的代码托管平台上。
- 5、鼓励同学之间探讨，鼓励合理使用人工智能平台，提升效率，但不应滥用相关资源，如抄袭代码和代写作业。

## 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监听相关的编程；掌握用 Libpcap 或 WinPcap 库监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

|Windows11 |

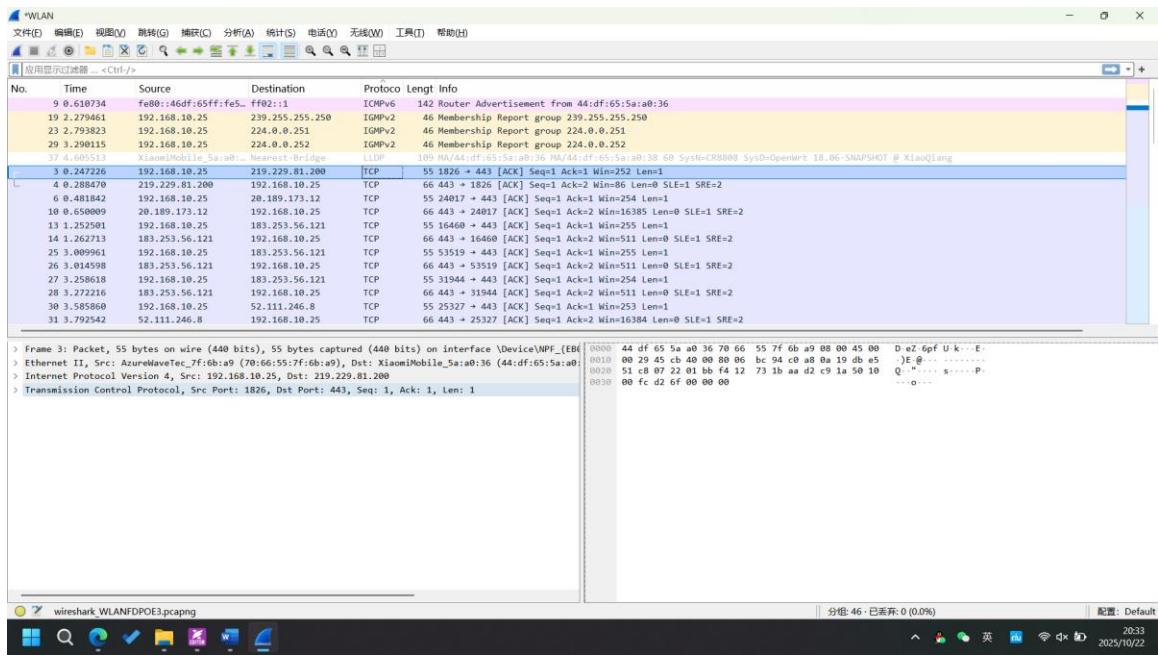
## 3 实验结果

### 1、用监听解析软件观察数据格式

用 Wireshark 或 Omnipcap 等网络监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，验证帧格式、IP 报文格式、TCP 段格式和 FTP 协议命令和响应的格式，验证 MAC 地址、IP 地址、TCP 端口等协议地址格式。

运行结果：

启动 Wireshark 进行抓包



## 1) 帧格式

```

▼ Frame 3: Packet, 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{EB04230A-A35B-4624-A78D-5EEFDCC8421A}
  Section number: 1
  > Interface id: 0 (\Device\NPF_{EB04230A-A35B-4624-A78D-5EEFDCC8421A})
    Encapsulation type: Ethernet (1)
    Arrival Time: Oct 22, 2025 20:31:12.223642000 中国标准时间
    UTC Arrival Time: Oct 22, 2025 12:31:12.223642000 UTC
    Epoch Arrival Time: 1761136272.223642000
    [Time shift for this packet: 0.00000000 seconds]
    [Time delta from previous captured frame: 129.441000 milliseconds]
    [Time delta from previous displayed frame: 129.441000 milliseconds]
    [Time since reference or first frame: 247.226000 milliseconds]
    Frame Number: 3
    Frame Length: 55 bytes (440 bits)
    Capture Length: 55 bytes (440 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:tcp]
    Character encoding: ASCII (0)
  
```

## 2) IP 报文格式

```

> Frame 3: Packet, 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{EB6...
> Ethernet II, Src: AzureWaveTec_7f:6b:a9 (70:66:55:7f:6b:a9), Dst: XiaomiMobile_5a:a0:36 (44:df:65:5a:a0:...
< Internet Protocol Version 4, Src: 192.168.10.25, Dst: 219.229.81.200
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 41
  Identification: 0x45cb (17867)
  > 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 128
  Protocol: TCP (6)
  Header Checksum: 0xbc94 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 192.168.10.25
  Destination Address: 219.229.81.200
  [Stream index: 1]
  > Transmission Control Protocol, Src Port: 1826, Dst Port: 443, Seq: 1, Ack: 1, Len: 1

```

### 3) TCP 段格式

```

> Frame 3: Packet, 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{...
> Ethernet II, Src: AzureWaveTec_7f:6b:a9 (70:66:55:7f:6b:a9), Dst: XiaomiMobile_5a:a0:36 (44:df:65:5a:...
> Internet Protocol Version 4, Src: 192.168.10.25, Dst: 219.229.81.200
< Transmission Control Protocol, Src Port: 1826, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
  Source Port: 1826
  Destination Port: 443
  [Stream index: 0]
  [Stream Packet Number: 1]
  > [Conversation completeness: Incomplete (12)]
  [TCP Segment Len: 1]
  Sequence Number: 1      (relative sequence number)
  Sequence Number (raw): 4094849819
  [Next Sequence Number: 2      (relative sequence number)]
  Acknowledgment Number: 1      (relative ack number)
  Acknowledgment number (raw): 2865940762
  0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
  Window: 252

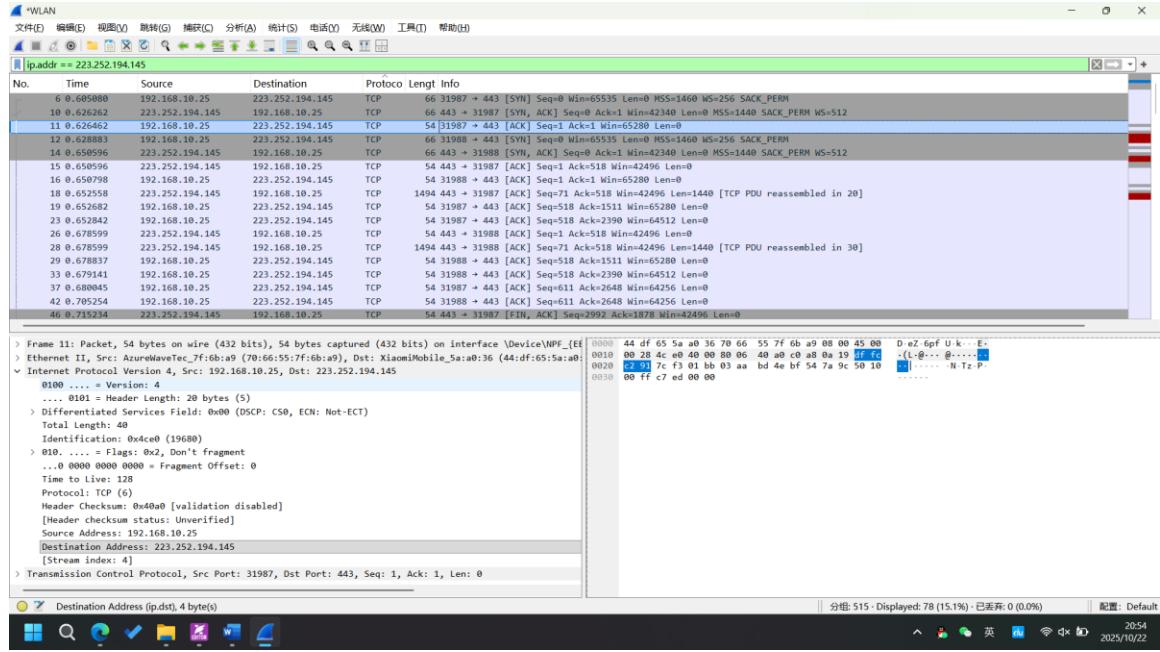
```

## 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。

运行结果：

抓包截图：



### 1) 三次握手:

6 0.605080	192.168.10.25	223.252.194.145	TCP	66 31987 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
10 0.626262	223.252.194.145	192.168.10.25	TCP	66 443 → 31987 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1440 SACK_PERM WS=512
11 0.626462	192.168.10.25	223.252.194.145	TCP	54 31987 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0

	SYN	ACK	Seq	Ack
第一次握手	1	0	0	0
第二次握手	1	1	0	1
第三次握手	0	1	1	1

### 2) 四次挥手

456 16.592575	192.168.10.25	120.220.145.116	TCP	54 1304 → 443 [FIN, ACK] Seq=1 Ack=1 Win=65280 Len=0
490 16.628876	120.220.145.116	192.168.10.25	TCP	54 443 → 1304 [ACK] Seq=1 Ack=2 Win=64256 Len=0
492 16.628876	120.220.145.116	192.168.10.25	TCP	54 443 → 1304 [FIN, ACK] Seq=1 Ack=2 Win=64256 Len=0
494 16.629064	192.168.10.25	120.220.145.116	TCP	54 1304 → 443 [ACK] Seq=2 Ack=2 Win=65280 Len=0

	FIN	ACK	Seq	Ack
第一次挥手	1	1	1	1
第二次挥手	0	1	1	2

第三次挥手	1	1	1	2
第四次挥手	0	1	2	2

### 3、用 Libpcap 或 WinPcap 库侦听网络数据

用 Libpcap 或 WinPcap 库侦听网络上的数据流，解析发送方与接收方的 MAC 和 IP 地址，并作记录与统计。程序在文件上输出形如下列 CSV 格式的日志：

时间、源 MAC、源 IP、目标 MAC、目标 IP、帧长度（以逗号间隔）

2015-03-14 13:05:16,60-36-DD-7D-D5-21,192.168.33.1,60-36- DD-7D-D5-72,192.168.33.2,1536

每隔一段时间（如 1 分钟），程序统计来自不同 MAC 和 IP 地址的通信数据长度，统计发至不同 MAC 和 IP 地址的通信数据长度。

运行结果：

使用 scapy 的 sniff 进行监听抓包，每 60s 抓一次

```
packets = sniff(timeout=timeout)
```

收集数据并写入 csv 中

```
for pkt in packets:
    if not pkt.haslayer(Ether): continue
    eth = pkt[Ether]
    src_mac = eth.src
    dst_mac = eth.dst
    length = len(pkt)
    src_ip, dst_ip = "", ""
    if pkt.haslayer(IP):
        ip = pkt[IP]
        src_ip, dst_ip = ip.src, ip.dst
    t = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    log_rows.append([t, src_mac, src_ip, dst_mac, dst_ip, length])
    counter[(src_ip, dst_ip)] += length
```

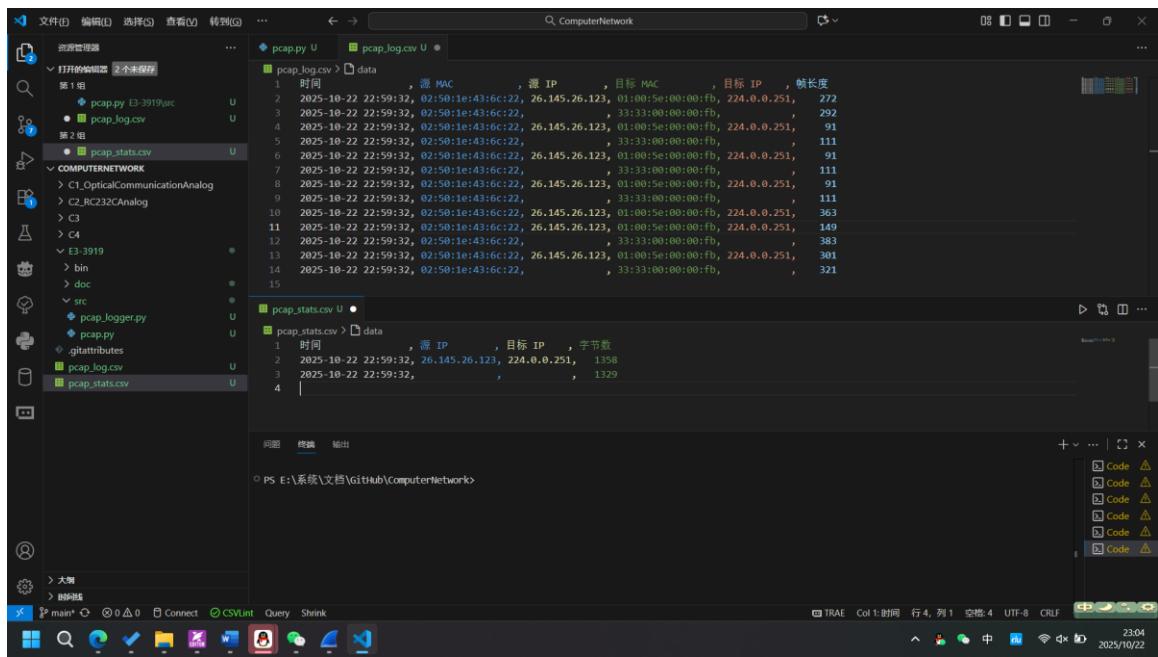
```

# 写日志
header_needed = not os.path.exists(LOG_CSV)
with open(LOG_CSV, "a", newline='', encoding='utf-8') as f:
    writer = csv.writer(f)
    if header_needed:
        writer.writerow(["时间", "源 MAC", "源 IP", "目标 MAC", "目标 IP", "帧长度"])
    writer.writerows(log_rows)

# 写统计
header_needed = not os.path.exists(STATS_CSV)
with open(STATS_CSV, "a", newline='', encoding='utf-8') as f:
    w = csv.writer(f)
    if header_needed:
        w.writerow(["时间", "源 IP", "目标 IP", "字节数"])
    for k, v in counter.items():
        w.writerow([datetime.now().strftime("%Y-%m-%d %H:%M:%S"), *k, v])

```

结果如图：

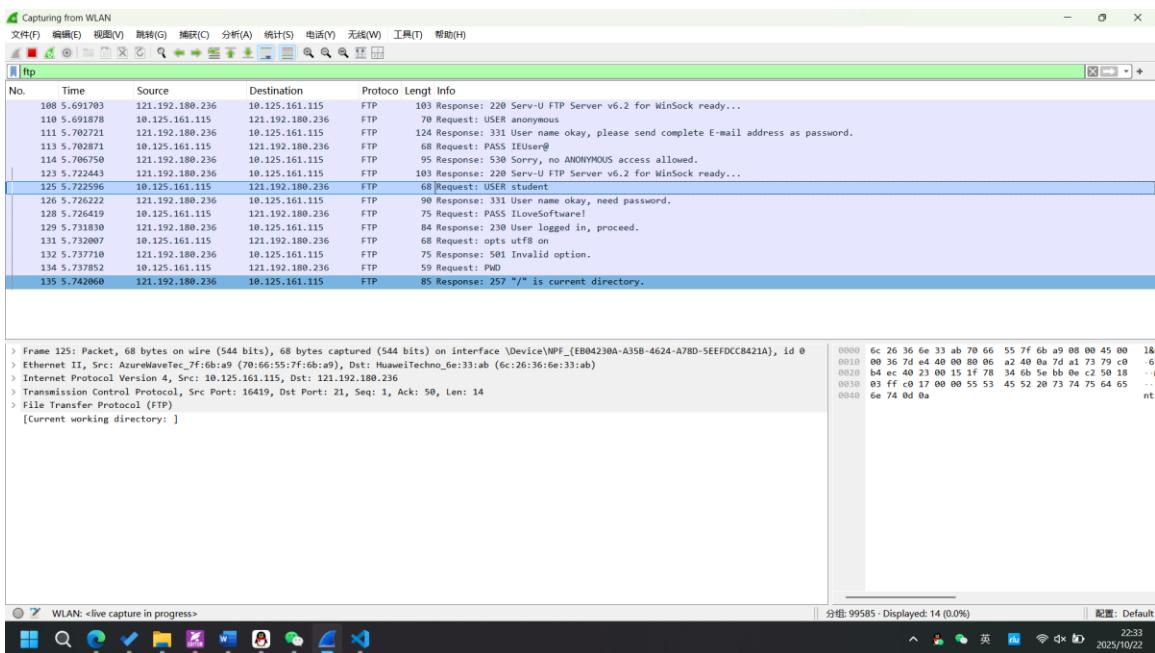


#### 4、解析侦听到的网络数据

用 Wireshark 侦听并观察 FTP 数据，分析其用户名密码所在报文的上下文特征，再总结出提取用户名密码的有效方法。解析协议内容，并作记录与统计。对用户登录行为进行记录。

运行结果：

Wireshark 抓包结果如图



```

68 Request: USER student
90 Response: 331 User name okay, need password.
75 Request: PASS ILoveSoftware!
84 Response: 230 User logged in, proceed.

68 Request: USER student
90 Response: 331 User name okay, need password.
67 Request: PASS 123456
74 Response: 530 Not logged in.

```

观察到，用户名相关：Request: USER student；密码相关：Request: PASS ILoveSoftware!；登录成功/失败信息：Response: 230 User logged in, proceed.；

## 4 实验代码

本次实验的代码已上传于以下代码仓库：  
[https://github.com/lzydroper/ComputerNetwork\\_Homework/tree/main/E3-3919](https://github.com/lzydroper/ComputerNetwork_Homework/tree/main/E3-3919)。（注意：建议使用码云，并设置公开权限；本学期暂不推荐使用 GitHub；如使用厦门大学私有 Git 服务，应将 whuang@xmu.edu.cn 加入项目成员备查，本段话删除。）

## 5 课后思考题

| (注明题号和题目文字，逐个回答课后思考题。如无，填写无。本段话删除。) |

## 6 实验总结

- 1.清晰的看到了 TCP 的三次握手、四次挥手过程，虽然还没教；
- 2.验证了以太网的帧格式，看到别人设计的这么详细丰富的头部感慨自己写的真实一坨；
- 3.scapy 真好用啊，比 winpcap 好用一万倍不止。