厦門大學

信息学院软件工程系

《计算机网络》实验报告

题　　目 实验五　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 黄子安

学　　号 **22920212204396**

实验时间 **2023** 年 **4** 月 **6** 日

2023 年 **4** 月 **10** 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

Windows11、C

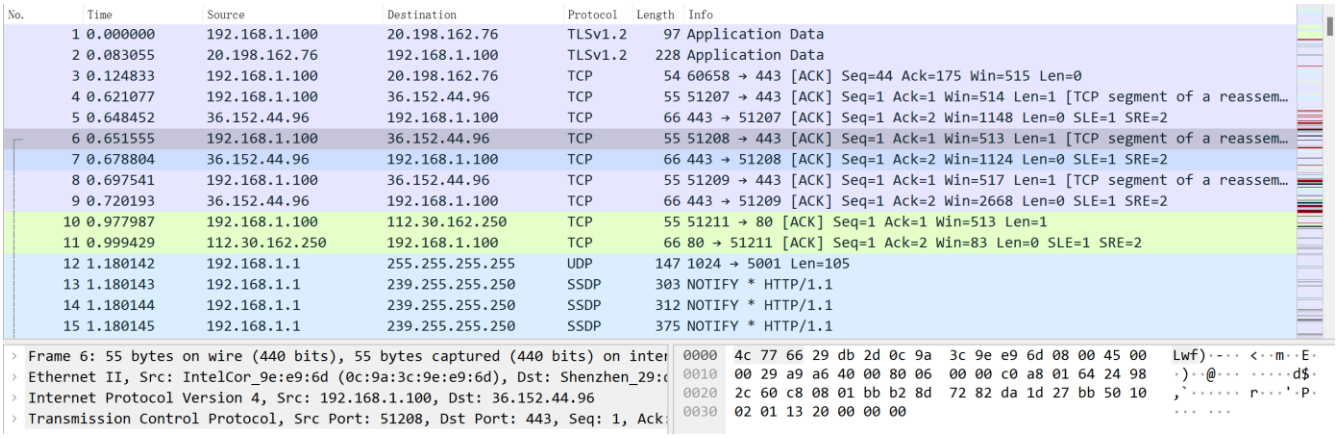
3 实验结果

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，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

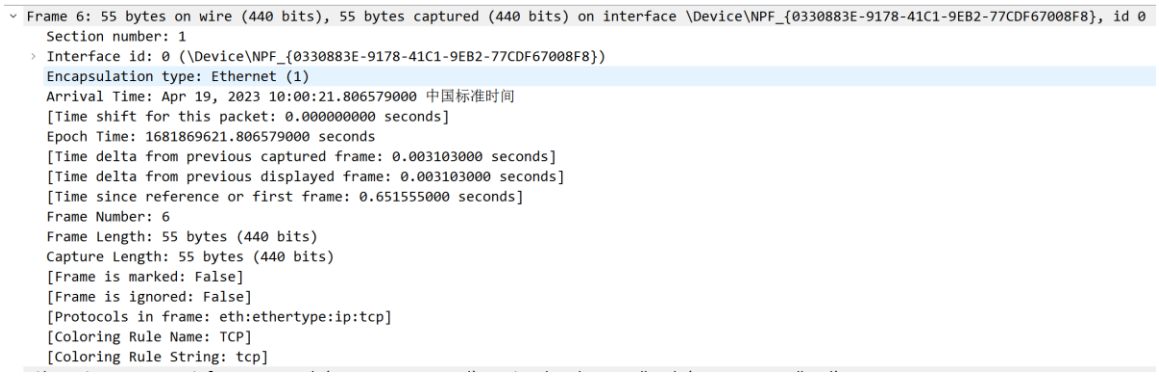
运行结果：

在 Wireshark 中输出了每一个数据包的如下几部分



接下来以图中灰色选中数据作为研究对象验证以上各种格式

(1)**帧格式**



其中我们可以获取到以下信息：

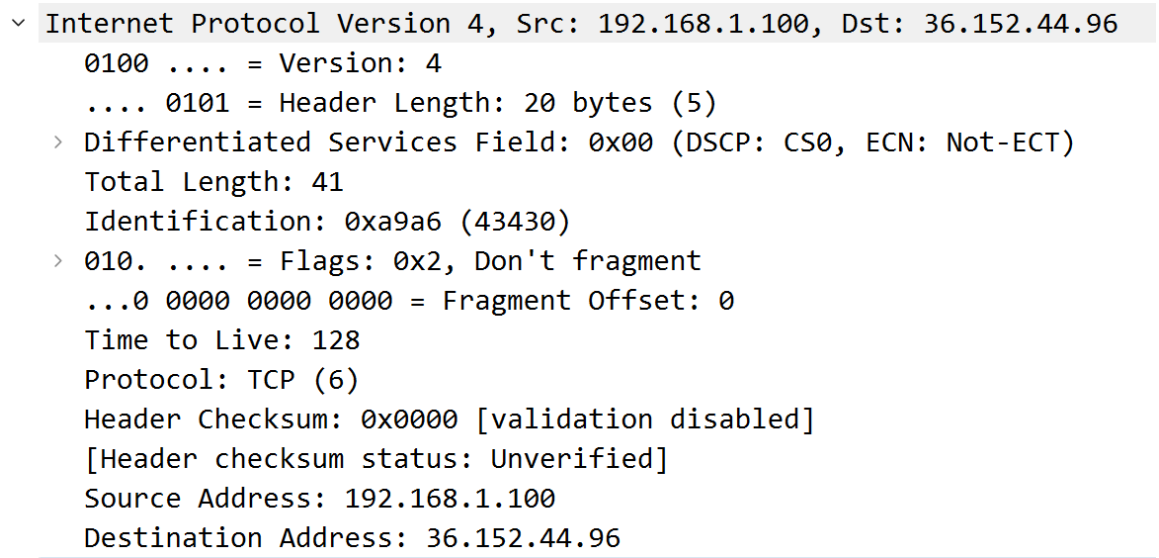
该帧是第 6 号数据帧，在线路上总共有 55 字节，Wireshark 成功捕获 55 字节数据

该帧到达时间是中国标准时间的 Apr 19, 2023 10:00:21.806579000

该帧与上一捕获、上一个发送的和第一帧的时间间隔

该帧中所封装的网络协议层次嵌套：eth：ethertype：ip：tcp

(2) IP 报文格式



根据 IP 数据报头部的格式可以和图中结果进行一一对应

版本号：由图中结果可知使用的是 IPv4

头部长度：该数据报的头部长度为 20 字节

服务类型：

```
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)
```

总长度：数据报的总长度是 41 个字节，结合头部长度可知载荷的长度为 21 字节

标识：标识为 16 位长，该 IP 数据报的标识为 43430，翻阅前后抓包的数据可以发现第 4 帧的标识为 43429，第 6 帧的标识为 4341，可以知道这些数据报来源于一个数据报，在重装的时候会被合到一块

标志：该数据报不可以被分片

```
010. .... = Flags: 0x2, Don't fragment
  0... .... = Reserved bit: Not set
  .1.. .... = Don't fragment: Set
  ..0. .... = More fragments: Not set
```

偏移地址：指示了本片在所属的原始报文中的位置为 0（**如果不是 0 要乘以 8**）

生存期：该数据报生存期还剩余 128，每个路由器处理数据报会将其-1，说明该数据报是有效的

类型：指明载荷所用的协议为 TCP

头部校验和：该数据报未被校验

源端 IP，目的端 IP：该数据报的源 IP 地址为 192.168.1.100，目的地址为 36.152.44.96

IP 可选项没有被使用，此时也无需使用填充来让头部长度达到 32 的倍数

(3)TCP 段格式

```
Transmission Control Protocol, Src Port: 51209, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
    Source Port: 51209
    Destination Port: 443
    [Stream index: 3]
    [Conversation completeness: Incomplete (60)]
    [TCP Segment Len: 1]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 1014779654
    [Next Sequence Number: 2    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 2294681927
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
    Window: 517
    [Calculated window size: 517]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x1320 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
    TCP payload (1 byte)
    [Reassembled PDU in frame: 275]
    TCP segment data (1 byte)
```

源端口号：51209　目的端口号：443　滑动窗口大小：517　未使用紧急指针

(4) FTP 协议命令和响应的格式

命令：

如图第一条是一个命令协议，noop 表示无动作，除了来自服务器上的承认

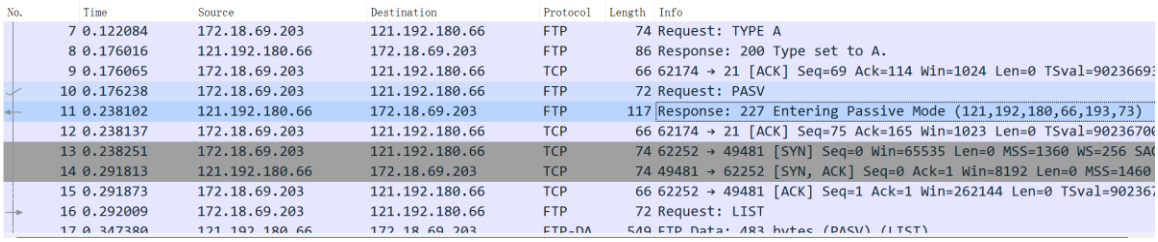
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.18.69.203	121.192.180.66	FTP	72	Request: noop
2	0.053236	121.192.180.66	172.18.69.203	FTP	85	Response: 200 Command okay.
3	0.053270	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=7 Ack=20 Win=1019
4	0.053432	172.18.69.203	121.192.180.66	FTP	120	Request: CWD /■■ʌ◆µ◆/◆ź■Й/◆◆◆◆◆◆
5	0.121935	121.192.180.66	172.18.69.203	FTP	140	Response: 250 Directory changed to /■■ʌ
6	0.121993	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=61 Ack=94 Win=1018
7	0.122084	172.18.69.203	121.192.180.66	FTP	74	Request: TYPE A
8	0.176016	121.192.180.66	172.18.69.203	FTP	86	Response: 200 Type set to A.
9	0.176065	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=69 Ack=114 Win=102
10	0.176238	172.18.69.203	121.192.180.66	FTP	72	Request: PASV

响应：

如图第 8 条数据是一个响应请求，200 表示命令 OK，

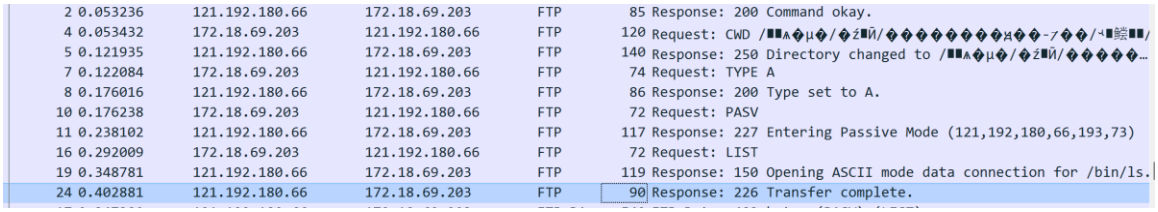
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.18.69.203	121.192.180.66	FTP	72	Request: noop
2	0.053236	121.192.180.66	172.18.69.203	FTP	85	Response: 200 Command okay.
3	0.053270	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=7 Ack=20 Win=1019
4	0.053432	172.18.69.203	121.192.180.66	FTP	120	Request: CWD /■■ʌ◆µ◆/◆ź■Й/◆◆◆◆◆
5	0.121935	121.192.180.66	172.18.69.203	FTP	140	Response: 250 Directory changed to /■■
6	0.121993	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=61 Ack=94 Win=101
7	0.122084	172.18.69.203	121.192.180.66	FTP	74	Request: TYPE A
8	0.176016	121.192.180.66	172.18.69.203	FTP	86	Response: 200 Type set to A.
9	0.176065	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=69 Ack=114 Win=10
10	0.176238	172.18.69.203	121.192.180.66	FTP	72	Request: PASV

227 表示进入被动模式



No.	Time	Source	Destination	Protocol	Length	Info
7	0.122084	172.18.69.203	121.192.180.66	FTP	74	Request: TYPE A
8	0.176016	121.192.180.66	172.18.69.203	FTP	86	Response: 200 Type set to A.
9	0.176065	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=69 Ack=114 Win=1024 Len=0 TSval=9023669
10	0.176238	172.18.69.203	121.192.180.66	FTP	72	Request: PASV
11	0.238102	121.192.180.66	172.18.69.203	FTP	117	Response: 227 Entering Passive Mode (121,192,180,66,193,73)
12	0.238137	172.18.69.203	121.192.180.66	TCP	66	62174 → 21 [ACK] Seq=75 Ack=165 Win=1023 Len=0 TSval=9023670
13	0.238251	172.18.69.203	121.192.180.66	TCP	74	62252 → 49481 [SYN] Seq=0 Win=65535 Len=0 MSS=1360 WS=256 SA
14	0.291813	121.192.180.66	172.18.69.203	TCP	74	49481 → 62252 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
15	0.291873	172.18.69.203	121.192.180.66	TCP	66	62252 → 49481 [ACK] Seq=1 Ack=1 Win=262144 Len=0 TSval=90236
16	0.292009	172.18.69.203	121.192.180.66	FTP	72	Request: LIST
17	0.347380	121.192.180.66	172.18.69.203	FTP-DA	549	FTP Data: 483 bytes (PASV) (LIST)

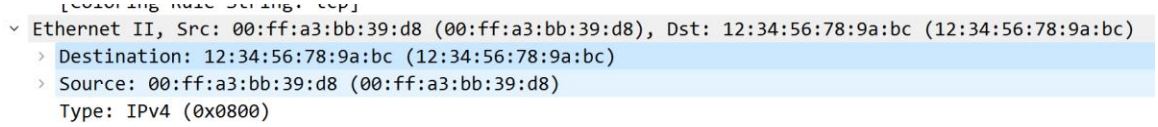
226 表示结束数据连接



2	0.053236	121.192.180.66	172.18.69.203	FTP	85	Response: 200 Command okay.
4	0.053432	172.18.69.203	121.192.180.66	FTP	120	Request: CWD /■■ʌ◆µ◆/◆źЙ/◆◆◆◆◆◆◆◆ɥ◆◆-ｱ◆◆/ｸ■鯵■■/
5	0.121935	121.192.180.66	172.18.69.203	FTP	140	Response: 250 Directory changed to /■■ʌ◆µ◆/◆źЙ/◆◆◆◆◆…
7	0.122084	172.18.69.203	121.192.180.66	FTP	74	Request: TYPE A
8	0.176016	121.192.180.66	172.18.69.203	FTP	86	Response: 200 Type set to A.
10	0.176238	172.18.69.203	121.192.180.66	FTP	72	Request: PASV
11	0.238102	121.192.180.66	172.18.69.203	FTP	117	Response: 227 Entering Passive Mode (121,192,180,66,193,73)
16	0.292009	172.18.69.203	121.192.180.66	FTP	72	Request: LIST
19	0.348781	121.192.180.66	172.18.69.203	FTP	119	Response: 150 Opening ASCII mode data connection for /bin/ls.
24	0.402881	121.192.180.66	172.18.69.203	FTP	90	Response: 226 Transfer complete.

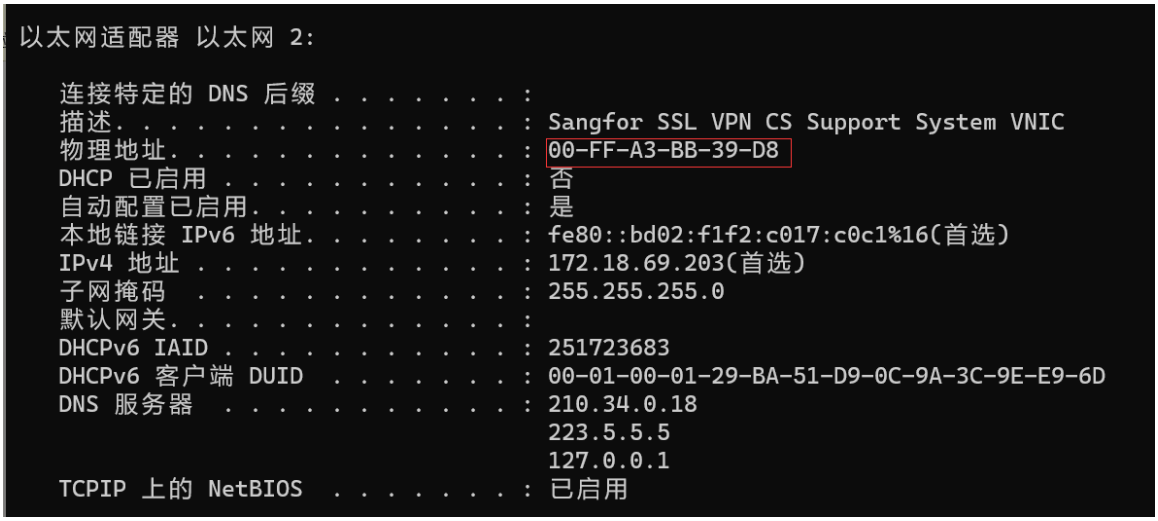
(5)验证 MAC 地址

在抓包数据的以太网格式可以找到本地的 MAC 地址



```
Ethernet II, Src: 00:ff:a3:bb:39:d8 (00:ff:a3:bb:39:d8), Dst: 12:34:56:78:9a:bc (12:34:56:78:9a:bc)
  Destination: 12:34:56:78:9a:bc (12:34:56:78:9a:bc)
  Source: 00:ff:a3:bb:39:d8 (00:ff:a3:bb:39:d8)
  Type: IPv4 (0x0800)
```

在命令行使用命令 ipconfig/all 可以查看本机的 MAC 地址，



```
以太网适配器 以太网 2:

   连接特定的 DNS 后缀 . . . . . :
   描述. . . . . . . . . . . . . . : Sangfor SSL VPN CS Support System VNIC
   物理地址. . . . . . . . . . . . : 00-FF-A3-BB-39-D8
   DHCP 已启用 . . . . . . . . . . : 否
   自动配置已启用. . . . . . . . . : 是
   本地链接 IPv6 地址. . . . . . . : fe80::bd02:f1f2:c017:c0c1%16(首选)
   IPv4 地址 . . . . . . . . . . . : 172.18.69.203(首选)
   子网掩码 . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . : 251723683
   DHCPv6 客户端 DUID . . . . . . . : 00-01-00-01-29-BA-51-D9-0C-9A-3C-9E-E9-6D
   DNS 服务器 . . . . . . . . . . . : 210.34.0.18
                                       223.5.5.5
                                       127.0.0.1
   TCPIP 上的 NetBIOS . . . . . . . : 已启用
```

经过对比可以发现该 MAC 地址就是抓包数据中的源地址，代表这条数据是由本机发送出去

（6）验证 IP 地址

```
Internet Protocol Version 4, Src: 172.18.69.203, Dst: 121.192.180.66
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 52
    Identification: 0x6f09 (28425)
  > 010. .... = Flags: 0x2, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: TCP (6)
    Header Checksum: 0x6bda [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.18.69.203
    Destination Address: 121.192.180.66
```

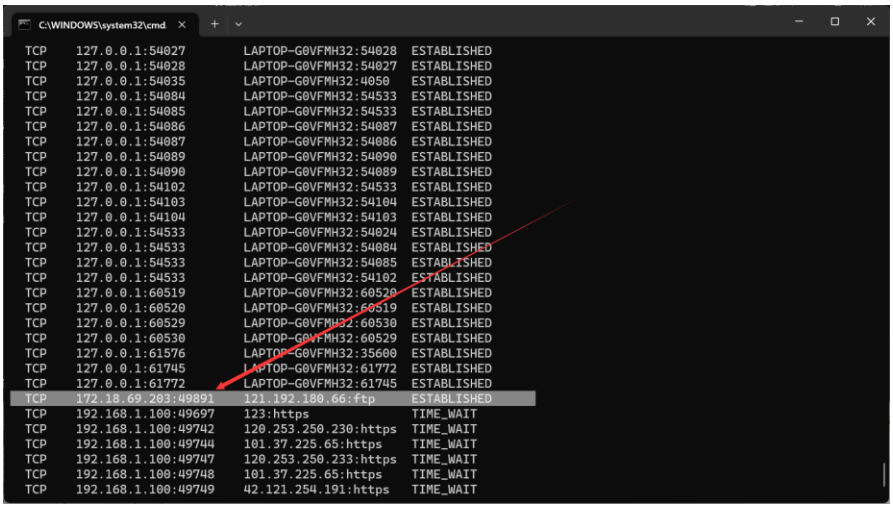
```
以太网适配器 以太网 2:

   连接特定的 DNS 后缀 . . . . . . . :
   描述. . . . . . . . . . . . . . . : Sangfor SSL VPN CS Support System VNIC
   物理地址. . . . . . . . . . . . . : 00-FF-A3-BB-39-D8
   DHCP 已启用 . . . . . . . . . . . : 否
   自动配置已启用. . . . . . . . . . : 是
   本地链接 IPv6 地址. . . . . . . . : fe80::bd02:f1f2:c017:c0c1%16(首选)
   IPv4 地址 . . . . . . . . . . . . : 172.18.69.203(首选)
   子网掩码 . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 251723683
   DHCPv6 客户端 DUID . . . . . . . . : 00-01-00-01-29-BA-51-D9-0C-9A-3C-9E-E9-6D
   DNS 服务器 . . . . . . . . . . . : 210.34.0.18
                                       223.5.5.5
                                       127.0.0.1
   TCPIP 上的 NetBIOS . . . . . . . . : 已启用
```

同理可以看到抓包数据中的 IP 源地址就是本机的 IP 地址

（7）验证 TCP 端口

在命令行中输入 Netstat -p TCP 查看所有 TCP 端口，可以通过 IP 地址定位到本机端口，与抓取的数据报一致，同时也可以知道该数据报请求的目的端口是 FTP

```
Transmission Control Protocol, Src Port: 49891, Dst Port: 21, Seq: 75, Ack: 166, Len: 0
    Source Port: 49891
    Destination Port: 21
    [Stream index: 0]
    [Conversation completeness: Incomplete (12)]
    [TCP Segment Len: 0]
    Sequence Number: 75    (relative sequence number)
    Sequence Number (raw): 2604934539
    [Next Sequence Number: 75    (relative sequence number)]
```

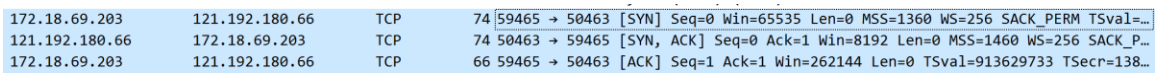


2、用侦听解析软件观察 TCP 机制

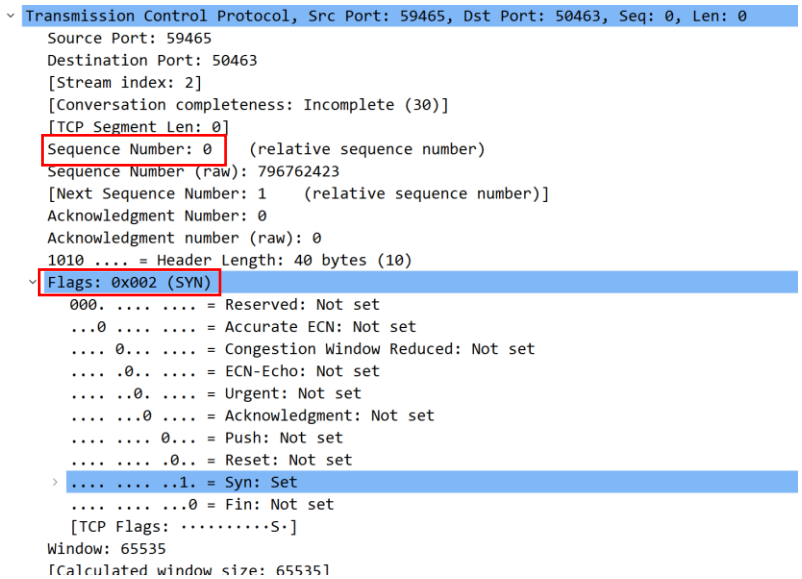
用 Wireshark 侦听并观察 TCP 数据段。观察其建立和撤除连接的过程，观察段 ID、窗口机制和拥塞控制机制等。

(1)TCP 三次握手

所用测试服务器 IP 为 121.192.180.66，本机 IP 为 172.18.69.203



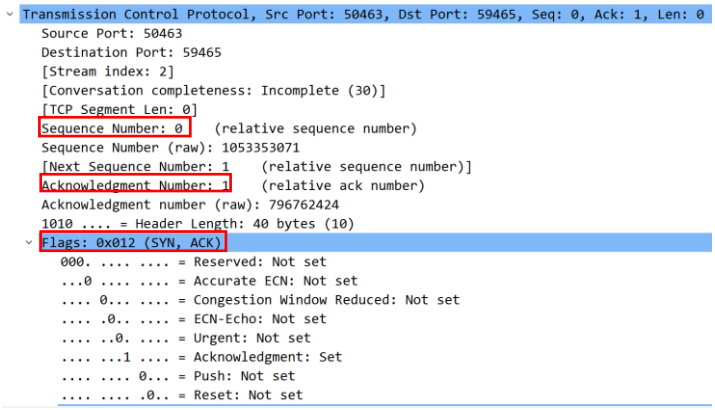
第一次握手：本机的 TCP 向服务器发送请求连接报文，首部中的同步位 SYN=1，并选择序号 seq=0



第二次握手：

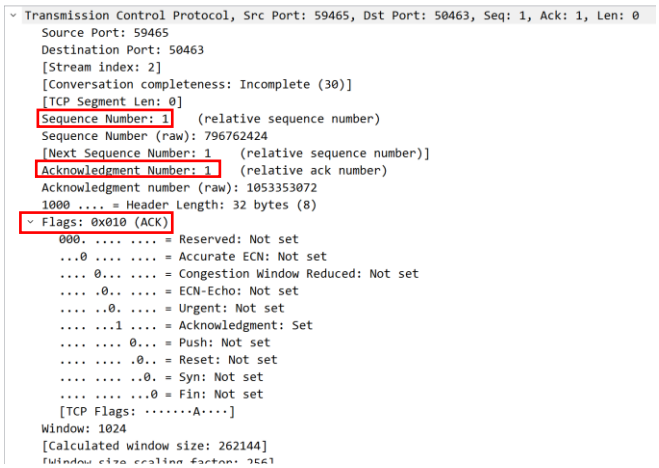
服务器的 TCP 收到连接请求报文段后同意并发回确认

确认报文段中 SYN=1，ACK=1，确认号 ack=1，自己选择的序号 seq=0



第三次握手：

本机收到此报文段之后再次向服务器确认，ACK=1，确认号 seq=1，ack,1



	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

```
172.18.69.203    121.192.180.66    TCP    66 59454 → 21 [FIN, ACK] Seq=85 Ack=407 Win=1020 Len=0 TSval=914191429 TSec…
121.192.180.66   172.18.69.203     TCP    66 21 → 59454 [ACK] Seq=407 Ack=86 Win=257 Len=0 TSval=1384055166 TSecr=914…
121.192.180.66   172.18.69.203     TCP    66 21 → 59454 [FIN, ACK] Seq=407 Ack=86 Win=257 Len=0 TSval=1384055166 TSec…
172.18.69.203    121.192.180.66    TCP    66 59454 → 21 [ACK] Seq=86 Ack=408 Win=1020 Len=0 TSval=914191434 TSecr=138…
```

第一次挥手

本机的应用进程先向其 TCP 发出连接释放报文段，并停止再发送数据，主动关闭 TCP 连接，主机设置连接释放报文段首部的 FIN = 1，其序号 seq = 85，等待服务器的确认

```
Transmission Control Protocol, Src Port: 59454, Dst Port: 21, Seq: 85, Ack: 407, Len: 0
   Source Port: 59454
   Destination Port: 21
   [Stream index: 0]
   [Conversation completeness: Incomplete (28)]
   [TCP Segment Len: 0]
   Sequence Number: 85    (relative sequence number)
   Sequence Number (raw): 2322519121
   [Next Sequence Number: 86    (relative sequence number)]
   Acknowledgment Number: 407    (relative ack number)
   Acknowledgment number (raw): 2761644453
   1000 .... = Header Length: 32 bytes (8)
 ⌄ Flags: 0x011 (FIN, ACK)
```

第二次挥手

服务器发出确认 ACK=1，确认号 ack=85+1=86，报文段的序号 seq = 407

```
⌄ Transmission Control Protocol, Src Port: 21, Dst Port: 59454, Seq: 407, Ack: 86, Len: 0
    Source Port: 21
    Destination Port: 59454
    [Stream index: 0]
    [Conversation completeness: Incomplete (28)]
    [TCP Segment Len: 0]
    Sequence Number: 407    (relative sequence number)
    Sequence Number (raw): 2761644453
    [Next Sequence Number: 407    (relative sequence number)]
    Acknowledgment Number: 86    (relative ack number)
    Acknowledgment number (raw): 2322519122
    1000 .... = Header Length: 32 bytes (8)
  ⌄ Flags: 0x010 (ACK)
```

第三次挥手

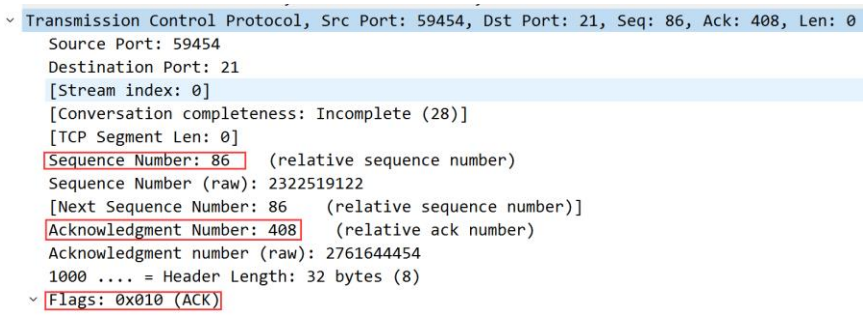
服务器已经没有向本机发送的数据，其应用进程通知 TCP 释放连接，FIN=1，ACK=1，序号 seq=407，ack=86

```
Transmission Control Protocol, Src Port: 21, Dst Port: 59454, Seq: 407, Ack: 86, Len: 0
   Source Port: 21
   Destination Port: 59454
   [Stream index: 0]
   [Conversation completeness: Incomplete (28)]
   [TCP Segment Len: 0]
   Sequence Number: 407    (relative sequence number)
   Sequence Number (raw): 2761644453
   [Next Sequence Number: 408    (relative sequence number)]
   Acknowledgment Number: 86    (relative ack number)
   Acknowledgment number (raw): 2322519122
   1000 .... = Header Length: 32 bytes (8)
 ⌄ Flags: 0x011 (FIN, ACK)
```

第四次挥手

本机在收到连接释放报文段后，必须发出确认。

在确认报文段中 ACK=1，确认号 ack=408，序号 seq=86

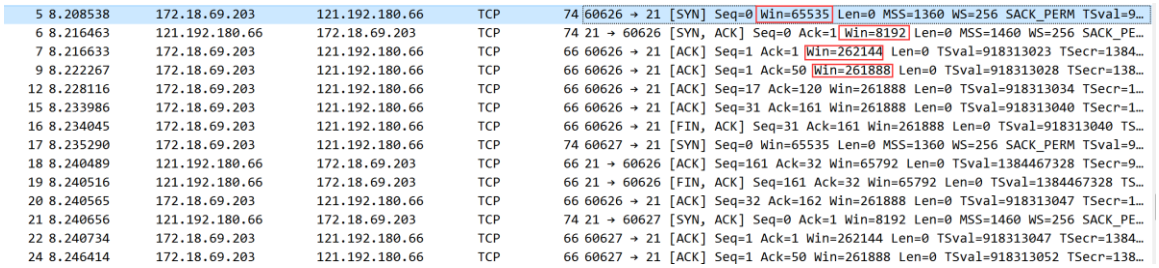


	FIN	ACK	seq	ack
第一次挥手	1	1	85	407
第二次挥手	0	1	407	86
第三次挥手	1	1	407	86
第四次挥手	0	1	86	408

(3)段 ID、窗口机制、拥塞控制机制

Win 表示窗口大小，TCP 利用滑动窗口机制对连接进行流量控制，发送方不能超过接收方给出的接受窗口值，

拥塞控制：TCP 使用拥塞窗口机制来时先拥塞空值，拥塞窗口的大小取决于网络的拥塞程度，并且会动态的发生改变。在开始建立连接时拥塞窗口较小，之后可以发现拥塞窗口变大来帮助发送更多分组，之后会根据拥塞程度调整窗口大小



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。

修改提供的 UDP 抓包代码，设置一个 ether_header 表示以太网帧头部，其中包含源 MAC 地址，目的 MAC 地址和帧类型

```
typedef struct ether_header {
    u_char src_mac[ETHER_ADDR_LEN];    // 目标 MAC 地址
    u_char dst_mac[ETHER_ADDR_LEN];    // 源 MAC 地址
    u_short ether_type;                // 以太网帧类型
}ether_header;
```

修改时间戳输出格式，补充上年月日信息，删除直接输出时间戳

```
/* convert the timestamp to readable format */
local_tv_sec = header->ts.tv_sec;
ltime = localtime(&local_tv_sec);
strftime(timestr, sizeof timestr, "%Y-%m-%d %H:%M:%S", ltime);
```

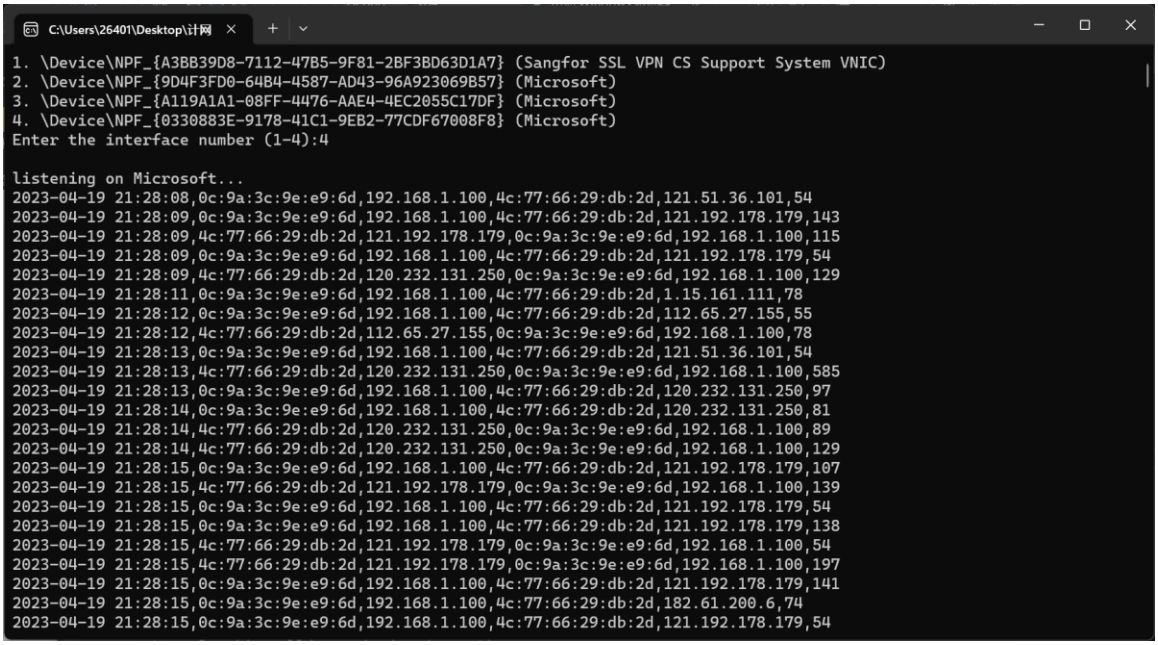
通过指针强制类型转换获取抓包数据头部中的以太网帧头部

```
/* retireve the position of the ether header */
eh = (ether_header*) pkt_data ;

/* retireve the position of the ip header */
ih = (ip_header*)(pkt_data +14); //length of ethernet header

/* convert from network byte order to host byte order */

/* print ip addresses and udp ports */

printf("%02x:%02x:%02x:%02x:%02x:%02x,",
    eh->src_mac[0], eh->src_mac[1], eh->src_mac[2],
    eh->src_mac[3], eh->src_mac[4], eh->src_mac[5]);

fprintf(fp,"%02x:%02x:%02x:%02x:%02x:%02x,",
    eh->src_mac[0], eh->src_mac[1], eh->src_mac[2],
    eh->src_mac[3], eh->src_mac[4], eh->src_mac[5]);
```

通过文件指针创建或打开一个 CSV 文件之后，将抓包数据中对应的内容进行输出

```
fp = fopen("MACandIP.csv", "a+");
if (!fp) exit(0);
```

运行结果：



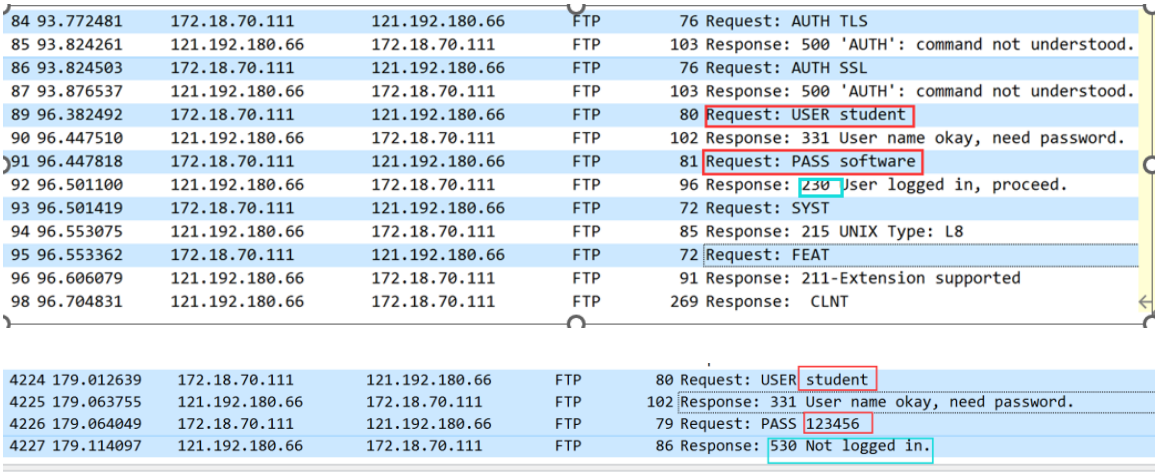
生成的 CSV 文件：

	A	B	C	D	E	F	G	H
1	时间	源 MAC	源 IP	目标 MAC	目标 IP	帧长度		
2	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.51.36.101	54		
3	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	143		
4	2023-4-19 21:28	4c:77:66:29:db:2d	121.192.178.179	0c:9a:3c:9e:e9:6d	192.168.1.100	115		
5	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	54		
6	2023-4-19 21:28	4c:77:66:29:db:2d	120.232.131.250	0c:9a:3c:9e:e9:6d	192.168.1.100	129		
7	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	1.15.161.111	78		
8	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	112.65.27.155	55		
9	2023-4-19 21:28	4c:77:66:29:db:2d	112.65.27.155	0c:9a:3c:9e:e9:6d	192.168.1.100	78		
10	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.51.36.101	54		
11	2023-4-19 21:28	4c:77:66:29:db:2d	120.232.131.250	0c:9a:3c:9e:e9:6d	192.168.1.100	585		
12	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	120.232.131.250	97		
13	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	120.232.131.250	81		
14	2023-4-19 21:28	4c:77:66:29:db:2d	120.232.131.250	0c:9a:3c:9e:e9:6d	192.168.1.100	89		
15	2023-4-19 21:28	4c:77:66:29:db:2d	120.232.131.250	0c:9a:3c:9e:e9:6d	192.168.1.100	129		
16	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	107		
17	2023-4-19 21:28	4c:77:66:29:db:2d	121.192.178.179	0c:9a:3c:9e:e9:6d	192.168.1.100	139		
18	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	54		
19	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	138		
20	2023-4-19 21:28	4c:77:66:29:db:2d	121.192.178.179	0c:9a:3c:9e:e9:6d	192.168.1.100	54		
21	2023-4-19 21:28	4c:77:66:29:db:2d	121.192.178.179	0c:9a:3c:9e:e9:6d	192.168.1.100	197		
22	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	141		
23	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	182.61.200.6	74		
24	2023-4-19 21:28	0c:9a:3c:9e:e9:6d	192.168.1.100	4c:77:66:29:db:2d	121.192.178.179	54		
25	2023-4-19 21:28	4c:77:66:29:db:2d	121.192.178.179	0c:9a:3c:9e:e9:6d	192.168.1.100	54		

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。

通过 Wireshark 侦听 FTP 数据，通过观察可以在其中找到登录名在“USER ”后面，口令在“PASS ”后面，之后“230”表示用户成功登录，如果是“530”则表示用户登录失败

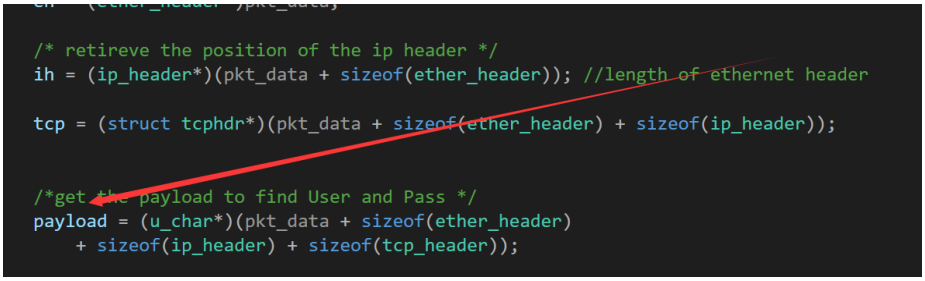


代码实现抓包：

设置过滤器为 FTP 的 TCP 默认端口

```
pcap_if_t* alldevs;
pcap_if_t* d;
int inum;
int i = 0;
pcap_t* adhandle;
char errbuf[PCAP_ERRBUF_SIZE];
u_int netmask;
char packet_filter[] = "port 21";
struct bpf_program fcode;

fp = fopen("FTP.csv", "w+");
if (!fp) exit(0);
fprintf(fp, "时间,源 MAC,源 IP,目标 MAC,目标 IP,登录名,口令,成功与否\n");
/* Retrieve the device list */
if (pcap_findalldevs(&alldevs, errbuf) == -1)
```

应用层没有数据头部，FTP 的相关命令和响应直接在载荷中，所以需要先获取数据报中的载荷，根据地址协议可知其在后面以太网头部、IP 头部、TCP 头部之后



```
/* retireve the position of the ip header */
ih = (ip_header*)(pkt_data + sizeof(ether_header)); //length of ethernet header

tcp = (struct tcphdr*)(pkt_data + sizeof(ether_header) + sizeof(ip_header));


/*get the payload to find User and Pass */
payload = (u_char*)(pkt_data + sizeof(ether_header)
    + sizeof(ip_header) + sizeof(tcp_header));
```

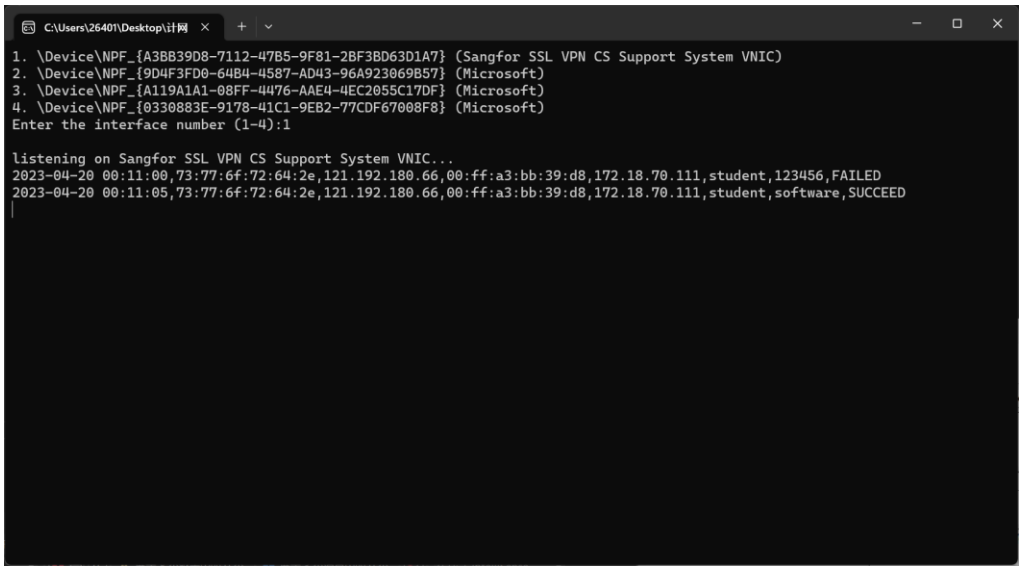
利用发现的数据报特点对登录名和口令以及登录是否成功进行获取，最后和之前一样写入 CSV 文件

```
static int find_username = 0, find_password = 0;
char* user = strstr(payload, "USER ");
if (user != NULL)
{
    char* end = strstr(user, "\r\n");
    if (end != NULL)
    {
        user += 5;
        int len = end - user;
        strncpy(username, user, len);
        username[len] = '\0';
        find_username = 1;
    }
}

char* pass = strstr(payload, "PASS ");
if (pass!=NULL)
{
    pass += 5;
    char* end = strstr(pass, "\r\n");
    if (end != NULL)
    {
        int len = end - pass;
        strncpy(password, pass, len);
        password[len] = '\0';
        find_password = 1;
    }
}
```

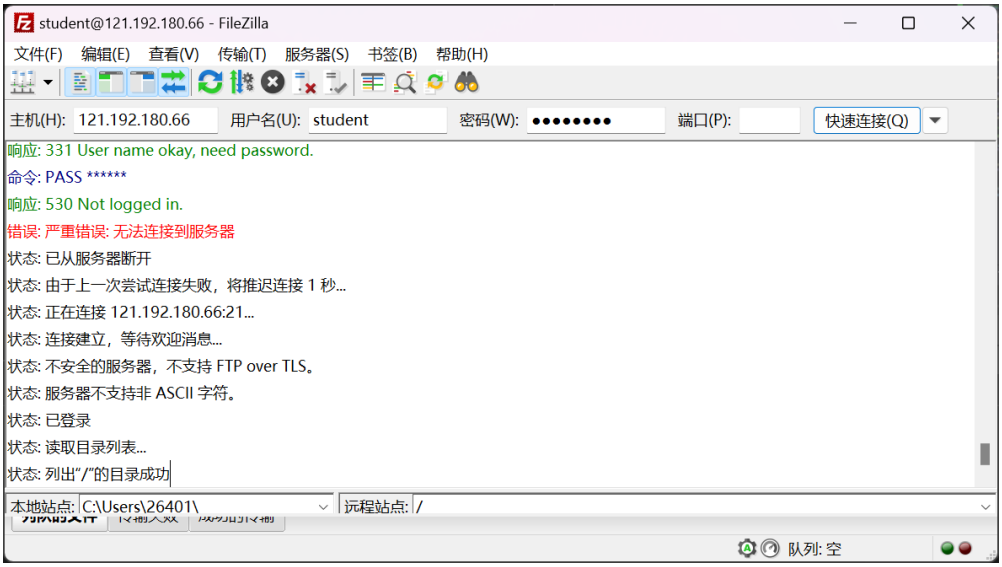
```
char* succeed = strstr(payload, "230");
if (succeed != NULL)
{
    is_successful = 1;
}

char* failed = strstr(payload, "530");
if (failed != NULL)
{
    is_successful = 0;
}
```

运行效果：

因为在寝室环境需要使用 VPN 才能登录学校 FTP，因此这里要切换成 1，这里抓取到了一次登录失败和登录成功，同时显示了对应输入的登录名和口令



FTP 软件登录信息：



CSV 文件输出：

	A	B	C	D	E	F	G	H	I
1	时间	源 MAC	源 IP	目标 MAC	目标 IP	登录名	口令	成功与否	
2	2023-4-20 0:11	73:77:6f:72:64:2e	121.192.180.66	00:ff:a3:bb:39:d8	172.18.70.111	student	123456	FAILED	
3	2023-4-20 0:11	73:77:6f:72:64:2e	121.192.180.66	00:ff:a3:bb:39:d8	172.18.70.111	student	software	SUCCEED	

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/aaaz718/ComputerInternet ）

5 实验总结

1.学习计算机网络要注意理论和实践相结合，这次实验用抓包软件观察到了TCP 三次握手、四次挥手等实际报文传输加深了对理论知识的理解。

2.通过对数据报的解读加深了应用层软件的理解，非常直观的看到了 FTP 的响应和命令。