

# 廈門大學



## 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 2019 级 4 班

姓    名 郑志豪

学    号 22920192204336

实验时间 2021 年 6 月 05 日

2021 年 6 月 05 日

# 填写说明

- 1、本文件为 Word 模板文件，建议使用 Microsoft Word 2019 打开，在可填写的区域中如实填写；
- 2、填表时，勿破坏排版，勿修改字体字号，打印成 PDF 文件提交；
- 3、文件总大小尽量控制在 1MB 以下，勿超过 5MB；
- 4、应将材料清单上传在代码托管平台上；
- 5、在学期最后一节课前按要求打包发送至 [cni21@qq.com](mailto:cni21@qq.com)。

## 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。

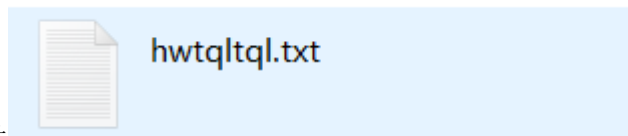
## 2 实验环境

Windows10

C 语言

## 3 实验结果

实验文件操作：在本机搭建一个 <ftp://192.168.100> 的服务器后，在局域网内用 IP 为 192.168.102



的客户端上传文件  
用 wireshark 侦听并获取报文

\*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)

tcp

No.	Time	Source	Destination	Protocol	Length	Info
23	4.820154	192.168.1.102	192.168.1.100	FTP	60	Request
24	4.820290	192.168.1.100	192.168.1.102	FTP	84	Response
25	4.822490	192.168.1.102	192.168.1.100	TCP	54	56062 →
26	4.822492	192.168.1.102	192.168.1.100	FTP	61	Request
27	4.822741	192.168.1.100	192.168.1.102	FTP	83	Response
28	4.830085	192.168.1.102	192.168.1.100	TCP	54	56062 →
29	4.832206	192.168.1.102	192.168.1.100	FTP	60	Request
30	4.832332	192.168.1.100	192.168.1.102	FTP	84	Response
31	4.833626	192.168.1.102	192.168.1.100	TCP	54	56062 →
32	4.834315	192.168.1.102	192.168.1.100	FTP	61	Request
33	4.834506	192.168.1.100	192.168.1.102	FTP	83	Response

## 1、 用侦听解析软件观察数据格式。

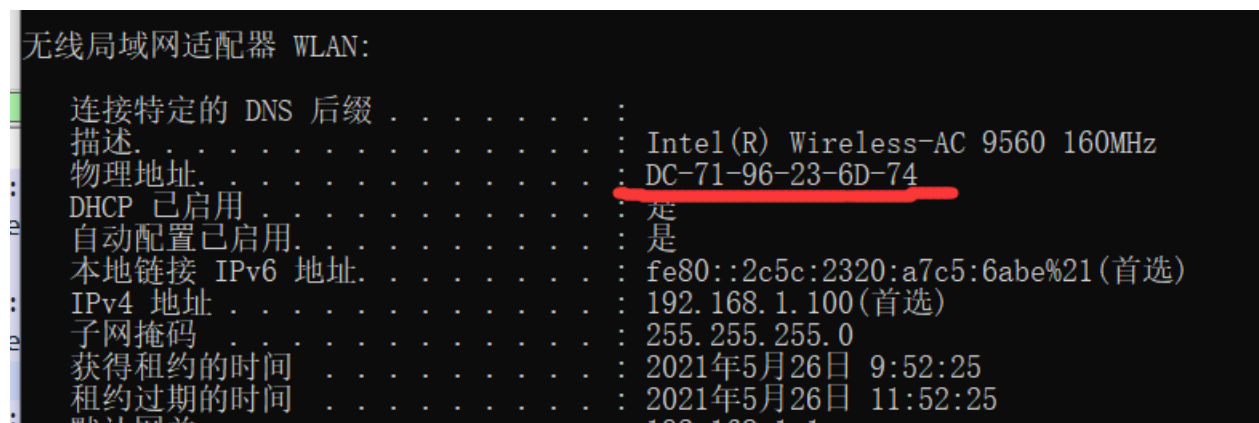
随意点开一个 TCP 报文查看格式：

- 1) MAC 帧格式，前 6 字节为目的地址（本机 mac 地址），再 6 字节为源地址，再 2 字节为类型说明（IPV4）

Ethernet II, Src: IntelCor\_3d:00:95 (90:78:41:3d:00:95), Dst: IntelCor\_23:6d:74 (dc:71:96:23:6d:74)

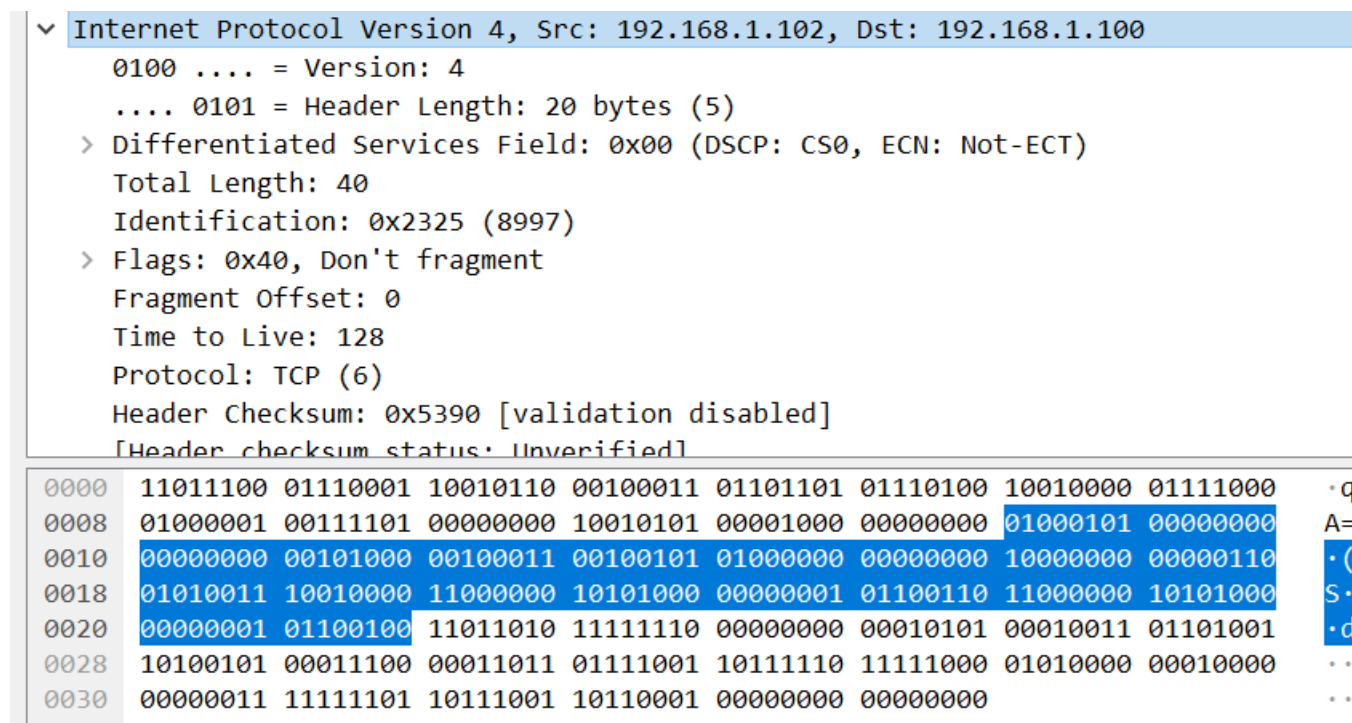
- > Destination: IntelCor\_23:6d:74 (dc:71:96:23:6d:74)
- > Source: IntelCor\_3d:00:95 (90:78:41:3d:00:95)
- Type: IPv4 (0x0800)
- > Internet Protocol Version 4, Src: 192.168.1.102, Dst: 192.168.1.100
- > Transmission Control Protocol, Src Port: 56062, Dst Port: 21, Seq: 14, Ack: 60, Len: 0

0000	11011100 01110001 10010110 00100011 01101101 01110100 10010000 01111000	·q·#mt·x
0008	01000001 00111101 00000000 10010101 00001000 00000000 01000101 00000000	A=····E·
0010	00000000 00101000 00100011 00100101 01000000 00000000 10000000 00000110	·(#%@···
0018	01010011 10010000 11000000 10101000 00000001 01100110 11000000 10101000	S····f··
0020	00000001 01100100 11011010 11111110 00000000 00010101 00010011 01101001	·d····i
0028	10100101 00011100 00011011 01111001 10111110 11111000 01010000 00010000	··y··P·
0030	00000011 11111101 10111001 10110001 00000000 00000000	·····



(图) 本机 mac 地址

## 2) ipv4 报文格式



第一个字节前半 0100 代表版本为 IPV4，后半 0101 代表头长度为 20 字节（5\*4 个 byte）

三四字节代表 IP 首部+后续数据之和的总长度 40 字节

五字节表示标识，表示已经产生了 8997 个数据报

六七字节前 3 位为分片标志符，中间位为 1 表示不分片，后 13 位为片偏移，不分片则为 0

八字节为生存时间 TTL 为 128

九字节为协议，6 表示 TCP

十十一字节为首部校验和。

后续字节为源地址与目的地址

### 3) TCP 报文格式

> Transmission Control Protocol, Src Port: 56062, Dst Port: 21, Seq: 14, Ack: 60,										
0000	11011100	01110001	10010110	00100011	01101101	01110100	10010000	01111000	00000000	...
0008	01000001	00111101	00000000	10010101	00001000	00000000	01000101	00000000	00000000	A=
0010	00000000	00101000	00100011	00100101	01000000	00000000	10000000	00000110	00000000	...
0018	01010011	10010000	11000000	10101000	00000001	01100110	11000000	10101000	00000000	S
0020	00000001	01100100	11011010	11111110	00000000	00010101	00010011	01101001	00000000	...
0028	10100101	00011100	00011011	01111001	10111110	11111000	01010000	00010000	00000000	...
0030	00000011	11111101	10111001	10110001	00000000	00000000	00000000	00000000	00000000	...

开头四个字节为源端口 56062 与目的端口 21（各占两字节）

再后四个字节为序号字段 seq

再后四个字节为确认号字段 ack

再后两字节中前四位为首部长度（数据偏移量），再六位为保留字段，最后六位为标志位，此处 ACK 为 1。

再后两字节为窗口字段，它指出了现在允许对方发送的数据量。

再后两字节为校验和，检验首部与数据。

再后两字节为紧急指针字段，指出文本中紧急数据共有多少字节。

因为只是一条确认报文，无数据段

## 2、用侦听解析软件观察 TCP 机制

以 time 为顺序参考，依次打开 [Stream index: 10] 报文

- 1) 客户端向服务端发送 SYN=1 的连接请求报文并附带 seq

```
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 828443538
[Next Sequence Number: 1 (relative sequence number)]
.....
> .... ..1. = Syn: Set
```

- 2) 服务端发送 ACK=1, SYN=1, ack (客户端的 next seq) 的确认报文并附带 seq

```
.... ..1 .... = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
> .... ..1. = Syn: Set
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 3179041440
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 828443539
```

- 3) 客户端向服务器发送确认包 ACK=1, ack (服务端的 next seq), 然后建立起 TCP 连接。

```
.... ..1 .... = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
.... .... ..0. = Syn: Not set
.... .... ...0 = Fin: Not set
```

- 4) 传送文件

```
> TRANSMISSION CONTROL PROTOCOL, Src Port: 5000
  ✓ File Transfer Protocol (FTP)
    ✓ STOR hwtqltql.txt\r\n
      Request command: STOR
      Request arg: hwtqltql.txt
      [Current working directory: /]
```

- 5) 客户端向服务端发送 FIN=1 的终止报文

```

.... ..0. .... = Urgent: Not set
.... ...1 .... = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
.... .... ..0. = Syn: Not set
> .... .... ...1 = Fin: Set

```

#### 6) 服务端发送 ACK=1 的确认报文

```

.... ..0. .... = Urgent: Not set
.... ...1 .... = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
.... .... ..0. = Syn: Not set
.... .... ...0 = Fin: Not set

```

#### 7) 服务端发送 FIN=1 的终止报文

```

.... ...1 .... = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
.... .... ..0. = Syn: Not set
✓ .... .... ...1 = Fin: Set

```

#### 7) 客户端发送 ACK=1 的确认终止报文

```

.... ...1 .... = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
.... .... ..0. = Syn: Not set
.... .... ...0 = Fin: Not set

```

### 3. 用 Libpcap 或 WinPcap 库侦听网络数据



csv.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```

2021/06/05 19:38:55, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 70
2021/06/05 19:38:55, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 70
2021/06/05 19:38:55, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 64
2021/06/05 19:38:55, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-7F-FF-FA, 239:255:255:250, 216
2021/06/05 19:38:55, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 64
2021/06/05 19:38:56, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 70
2021/06/05 19:38:56, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 70
2021/06/05 19:38:56, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-7F-FF-FA, 239:255:255:250, 216
2021/06/05 19:38:57, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-7F-FF-FA, 239:255:255:250, 216
2021/06/05 19:38:57, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 82
2021/06/05 19:38:58, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-7F-FF-FA, 239:255:255:250, 216
2021/06/05 19:38:58, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 82
2021/06/05 19:39:00, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92
2021/06/05 19:39:00, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 82
2021/06/05 19:39:00, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92
2021/06/05 19:39:01, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 74
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 77
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 74
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 74
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 77
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 68
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FB, 224: 0: 0:251, 74
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 71
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 68
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 68
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 68
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 71
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, 01-00-5E-00-00-FC, 224: 0: 0:252, 71
2021/06/05 19:39:02, 00-50-56-C0-00-08, 192.168.224: 1, FF-FF-FF-FF-FF-FF, 192.168.224:255, 92

```

## 4 观察 FTP 数据

ftp 数据报文是利用 tcp 协议传输的，登陆时所传输的登录信息是放在 tcp 报文的数据中。

以 530 开头，表示登录失败，以 230 开头表示登录成功。

其中登录信息

用户名和密码是分开传送，先认定用户名有效，后认定密码有效。

登录成功的数据报：

64	8.073368	192.168.2.100	121.192.180.66	FTP	68 Request: US
65	8.077648	121.192.180.66	192.168.2.100	FTP	90 Response: 3
66	8.077693	192.168.2.100	121.192.180.66	TCP	54 61292 → 21
67	8.077751	192.168.2.100	121.192.180.66	FTP	69 Request: PA
68	8.082516	121.192.180.66	192.168.2.100	FTP	84 Response: 2

登录失败的数据报：

244	9.621156	192.168.2.100	121.192.180.66	FTP	64 Request:
245	9.625924	121.192.180.66	192.168.2.100	FTP	90 Response:
246	9.626009	192.168.2.100	121.192.180.66	TCP	54 61340 → 2
247	9.626057	192.168.2.100	121.192.180.66	FTP	69 Request:
248	9.629725	121.192.180.66	192.168.2.100	FTP	74 Response:

## 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/zzh221/cnet-exp3>

## 5 实验总结

掌握了 tcp 报文格式与连接过程

掌握了 ftp 登录时的通信过程

掌握了怎么使用 wireshark 观察网络流量，并辅助进行网络监听相关的编程

掌握了使用 winpcap 库监听处理以太网帧和 ip 报文的方法等计算机网络的基础

知识和网络编程方法。