计算机网络实验报告

院系	数据科学与计算机学院	班 级	软件工程教务五	组长	杨羽菲
学号	15331416				
学生	赵寒旭				
实验分工					

【实验题目】网络嗅探与协议分析实验

【实验目的】 通过网络嗅探了解网络数据类型、了解网络工作原理；学习相关工具的使用。

【实验内容】

(1) HTTP 协议分析：完成实验教程实例 2-3 的实验，回答实验提出的问题及实验思考。(P62)

(2) FTP 协议分析：完成实验教程实例 2-4。回答实验提出的问题及实验思考。(P66)

【实验要求】

一些重要信息信息需给出截图。

注意实验步骤的前后对比。

【实验记录】(如有实验拓扑请自行画出，要求自行画出拓扑图)

1. HTTP 协议分析实验

1.1 实验过程

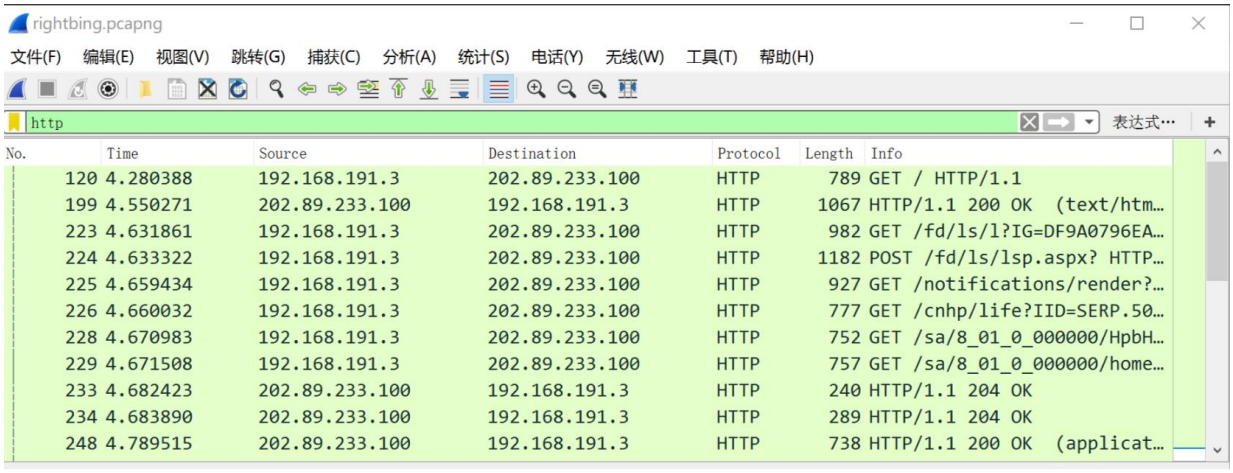
(1) 找到待连接网址 www.bing.com 的 IP 地址

```
C:\Users\lenovo>ping www.bing.com

正在 Ping cn-0001.cn-msedge.net [202.89.233.100] 具有 32 字节的数据:
来自 202.89.233.100 的回复: 字节=32 时间=153ms TTL=111
来自 202.89.233.100 的回复: 字节=32 时间=149ms TTL=111
来自 202.89.233.100 的回复: 字节=32 时间=154ms TTL=111
来自 202.89.233.100 的回复: 字节=32 时间=144ms TTL=111

202.89.233.100 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 144ms，最长 = 154ms，平均 = 150ms
```

(2) 确定过滤方式

ip.dst==202.89.233.100 || ip.src==202.89.233.100 && http

(3)打开浏览器，输入网址，捕获数据包



(4)分析捕获的数据包

1.2 实验问题：

(1) 在捕获的报文中共有几种 HTTP 报文？客户机与服务器之间共建立了几个连接？服务器和客户机分别使用了哪几个端口？

Ans:

1) 共有两种 HTTP 报文，一种是客户机到服务器的请求报文，一种是服务器到客户机的响应报文。
2) 一次请求一次响应为一次连接。
   访问 www.bing.com
   共建立了 12 个连接
3) 服务器和客户机分别使用哪几个端口

服务器和客户机使用端口	
服务器端口号	客户机端口号
80	63825
80	63826
80	63828
80	63830
80	63831
80	63829

(2) 在捕获的 HTTP 报文中，选择一个 HTTP 请求报文和对应的 HTTP 应答报文，分析结果如下表

HTTP请求报文					
方法	GET	版本	HTTP/1.1	URL	http://www.bing.com
首部字段名	字段值	字段所表达的信息			
Host	cn.bing.com	请求资源所在服务器			
Connection	keep-alive	逐跳首部、连接的管理			
Upgrade	1	升级为其他协议			
User-Agent	Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36	HTTP 客户端程序的信息			
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8	用户代理可处理的媒体类型			
Accept-Encoding	gzip, deflate	优先的内容编码			
Accept-Language	zh-CN,zh;q=0.8	优先的语言（自然语言）			

HTTP应答报文					
版本	HTTP/1.1	状态码	200	短语	OK
首部字段名	字段值	字段所表达的信息			
Cache-Control	private, max-age=0	响应输出到客户端后，服务端通过该报文头属告诉客户端如何控制响应内容的缓存。			
Content_Length	Content-Length: 47140	响应正文长度			
Content-Type	text/html; charset=utf-8	响应正文的类型（是图片还是二进制字符串）			
Content_Encoding	gzip	响应正文使用的数据压缩格式			
Vary	Accept-Encoding	代理服务器缓存的管理信息			
Date	Sun, 15 Oct 2017 06:25:44 GMT	消息发送的时间			

(3) 综合分析捕获的报文，理解 http 协议的工作过程，将结果填入下表

HTTP协议工作过程

客户端口号	服务器端口号	所包括的报文号	工作过程
63825	80	28	客户机向服务器发起连接请求
63826	80	29	客户机请求获取资源
63828	80	31	客户机请求获取资源
63830	80	33	客户机请求获取资源
63831	80	34	客户机请求获取资源
63829	80	32	服务器响应请求，释放TCP连接

(4) 在第 1 个和第 3 个 HTTP 会话中，Web 服务器对 Web 客户端 GET 请求的响应是什么？

响应均为：

响应码：200

短语：OK

1.3 实验思考

(1) 实验中哪台计算机启动了 HTTP 会话，是如何启动的？

Destination	Protocol	Length	Info
202.89.233.100	TCP	66	63825 → 80 [SYN] Seq=0 Win=17520 Len=0 MSS=1460 WS=256 SACK_PERM=1
192.168.191.3	TCP	66	80 → 63825 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1440 WS=256 SACK_PERM=1
202.89.233.100	TCP	54	63825 → 80 [ACK] Seq=1 Ack=1 Win=17408 Len=0
202.89.233.100	HTTP	789	GET / HTTP/1.1

192.168.191.3

通过三次 TCP 握手连接

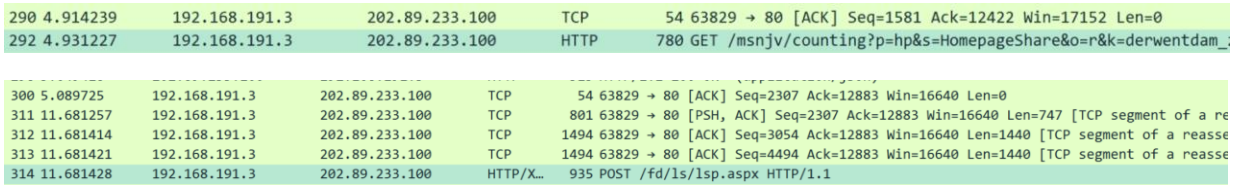
(2) 哪台计算机首先发出了结束 HTTP 会话的信号？是如何发出的？

FIN 表示没有数据需要发送了（在关闭 TCP 连接的时候使用）：任何一方都可以关闭一个 TCP 连接，要求双方发送一个 FIN 信号关闭自己的通讯频道。一方可以在另一方之前关闭，或者双方同时关闭。

当一方发送一个 FIN 信号时，另一方可发送“FIN+ACK”，开始关闭自己一方的通信并且确认收到了第一个FIN 信号。发送第一个FIN 信号的人接下来再发送一个“FIN+ACK”信息，确认收到第二个 FIN 信号。另一方就知道这个连接已经关闭了，并且关闭了自己的连接。发送第一个 FIN 的人没有办法收到最后一个 ACK 信号的确认信息。这时它会进入“TIME_WAIT”（等待时间）状态并启动一个定时器，防止另一方没有收到 ACK 信息并且认为连接仍是打开的。一般来说，这个状态会持续 1 至 2 分钟。

注：连接 bing 的过程中，并没有结束 HTTP 会话的信号发出。通常是由客户机对服务器发出终端连接请求。

(3) GET 方法取回由 Request-URL 标识的信息，POST 方法可以用于提交表单。请寻找一个有表单提交特征的网页，访问该网页，捕获数据包并分析请求方法中的 GET 和 POST 方法。



290	4.914239	192.168.191.3	202.89.233.100	TCP	54 63829 → 80 [ACK] Seq=1581 Ack=12422 Win=17152 Len=0
292	4.931227	192.168.191.3	202.89.233.100	HTTP	780 GET /msnjv/counting?p=hp&s=HomepageShare&o=r&k=derwentdam_
300	5.089725	192.168.191.3	202.89.233.100	TCP	54 63829 → 80 [ACK] Seq=2307 Ack=12883 Win=16640 Len=0
311	11.681257	192.168.191.3	202.89.233.100	TCP	801 63829 → 80 [PSH, ACK] Seq=2307 Ack=12883 Win=16640 Len=747 [TCP segment of a re
312	11.681414	192.168.191.3	202.89.233.100	TCP	1494 63829 → 80 [ACK] Seq=3054 Ack=12883 Win=16640 Len=1440 [TCP segment of a reasse
313	11.681421	192.168.191.3	202.89.233.100	TCP	1494 63829 → 80 [ACK] Seq=4494 Ack=12883 Win=16640 Len=1440 [TCP segment of a reasse
314	11.681428	192.168.191.3	202.89.233.100	HTTP/X…	935 POST /fd/ls/lsp.aspx HTTP/1.1

GET：从服务器上获取数据，也就是所谓的查，仅仅是获取服务器资源，不进行修改。

1）GET 方式是以实体的方式得到由请求 URL 所指定资源的信息，如果请求 URL 只是一个数据产生过程，那么最终要在响应实体中返回的是处理过程的结果所指向的资源，而不是处理过程的描述。也就是说，GET 的到的信息是资源，而不是资源的处理过程。

2）请的求的数据会附加在 URL 之后，以？分隔 URL 和传输数据，多个参数用&连接。URL 编码格式采用的是 ASCII 编码，而不是 Unicode，即所有的非 ASCII 字符都要编码之后再传输。

3）因为 URL 的长度限制，GET 方式传输的数据大小有所限制，传送的数据量不超过 2KB。

4）GET 方式服务器端用 Request.QueryString 获取变量的值。

5）GET 方式传输的参数安全性低，因为传输的数据会显示在请求的 URL 中。

POST：向服务器提交数据，这就涉及到了数据的更新，也就是更改服务器的数据。

1）用来向目的服务器发出请求，要求它接收被附在请求后的实体，并把它当做请求队列中请求 URL 所指定资源的附加新子项。

2）POST 方式将表单内各个字段和内容放置在 HTML HEADER 中一起传送到 Action 属性所指定的 URL 地址，用户是看不到这个过程的。

3）POST 方式传送的数据量比较大，一般被默认为没有限制，但是根据 IIS 的配置，传输量也是不同的。

4）POST 方式在服务器端用 Request.Form 获取提交的数据。

5）POST 方式传输的数据安全性较高，因为数据传输不是明显显示的。

2. FTP 协议分析

2.1 实验过程

(1) 在命令行窗口登录 FTP 服务器，捕获 ftp 报文

```
C:\>ftp 222.200.180.18
连接到 222.200.180.18。
220-FileZilla Server 0.9.53 beta
220 涓北澶ｙ宓屽叆寮忕郴缁熷疄楠屽 FTP
202 UTF8 mode is always enabled. No need to send this command.
用户(222.200.180.18:(none)): net2017
331 Password required for net2017
密码:
230 Logged on
ftp> quit
221 Goodbye

C:\>
```

FTP-DOS

ftp

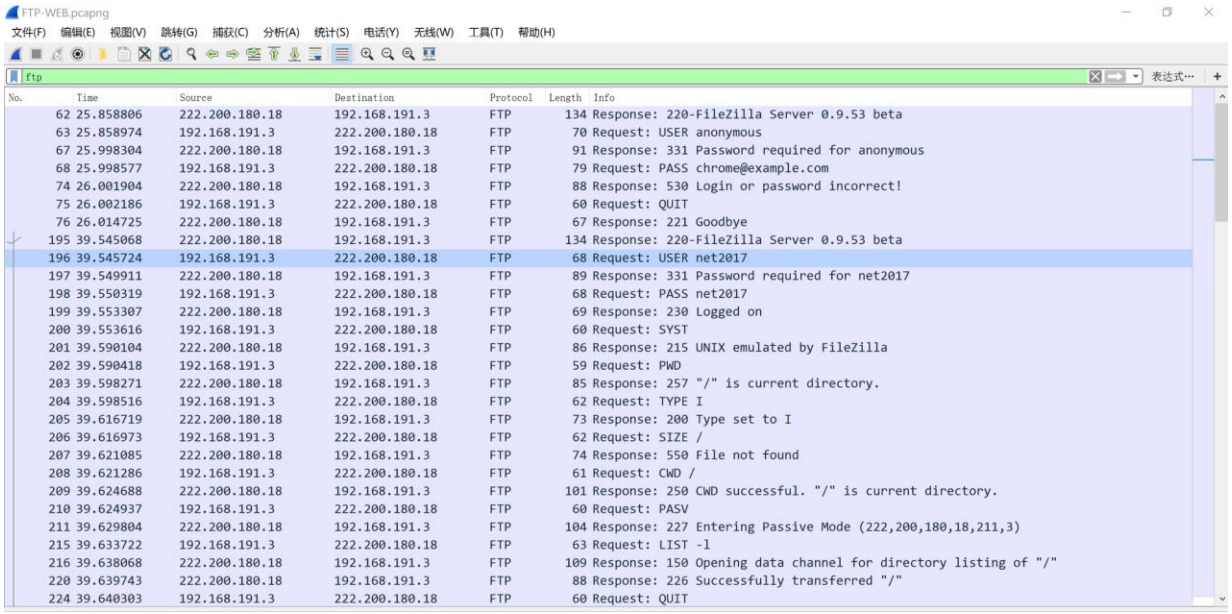
No.	Time	Source	Destination	Protocol	Length	Info
9	7.150558	222.200.180.18	192.168.191.3	FTP	134	Response: 220-FileZilla Server 0.9.53 beta
10	7.155004	192.168.191.3	222.200.180.18	FTP	68	Request: OPTS UTF8 ON
11	7.157161	222.200.180.18	192.168.191.3	FTP	118	Response: 202 UTF8 mode is always enabled. No need to send this command.
19	17.339774	192.168.191.3	222.200.180.18	FTP	68	Request: USER net2017
20	17.343949	222.200.180.18	192.168.191.3	FTP	89	Response: 331 Password required for net2017
22	21.318663	192.168.191.3	222.200.180.18	FTP	68	Request: PASS net2017
23	21.346029	222.200.180.18	192.168.191.3	FTP	69	Response: 230 Logged on
678	34.749178	192.168.191.3	222.200.180.18	FTP	60	Request: QUIT
680	34.756702	222.200.180.18	192.168.191.3	FTP	67	Response: 221 Goodbye

	Time	Source	Destination	Protocol	Length	Info
6	7.139602	192.168.191.3	222.200.180.18	TCP	66	51185 → 21 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=1 SACK_PERM=1
7	7.145561	222.200.180.18	192.168.191.3	TCP	66	21 → 51185 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
8	7.145644	192.168.191.3	222.200.180.18	TCP	54	51185 → 21 [ACK] Seq=1 Ack=1 Win=8192 Len=0
9	7.150558	222.200.180.18	192.168.191.3	FTP	134	Response: 220-FileZilla Server 0.9.53 beta
10	7.155004	192.168.191.3	222.200.180.18	FTP	68	Request: OPTS UTF8 ON
11	7.157161	222.200.180.18	192.168.191.3	FTP	118	Response: 202 UTF8 mode is always enabled. No need to send this command.
12	7.203840	192.168.191.3	222.200.180.18	TCP	54	51185 → 21 [ACK] Seq=15 Ack=145 Win=8048 Len=0
19	17.339774	192.168.191.3	222.200.180.18	FTP	68	Request: USER net2017
20	17.343949	222.200.180.18	192.168.191.3	FTP	89	Response: 331 Password required for net2017
21	17.392422	192.168.191.3	222.200.180.18	TCP	54	51185 → 21 [ACK] Seq=29 Ack=180 Win=8013 Len=0
22	21.318663	192.168.191.3	222.200.180.18	FTP	68	Request: PASS net2017
23	21.346029	222.200.180.18	192.168.191.3	FTP	69	Response: 230 Logged on
24	21.392825	192.168.191.3	222.200.180.18	TCP	54	51185 → 21 [ACK] Seq=43 Ack=195 Win=7998 Len=0
678	34.749178	192.168.191.3	222.200.180.18	FTP	60	Request: QUIT
680	34.756702	222.200.180.18	192.168.191.3	FTP	67	Response: 221 Goodbye
681	34.756703	222.200.180.18	192.168.191.3	TCP	60	21 → 51185 [FIN, ACK] Seq=208 Ack=49 Win=65536 Len=0
682	34.756825	192.168.191.3	222.200.180.18	TCP	54	51185 → 21 [ACK] Seq=49 Ack=209 Win=7985 Len=0
683	34.761833	192.168.191.3	222.200.180.18	TCP	54	51185 → 21 [FIN, ACK] Seq=49 Ack=209 Win=7985 Len=0
684	34.767116	222.200.180.18	192.168.191.3	TCP	60	21 → 51185 [ACK] Seq=209 Ack=50 Win=65536 Len=0

可以看到 tcp 的三次握手和四次挥手分别是前三个数据包和后四个数据包。

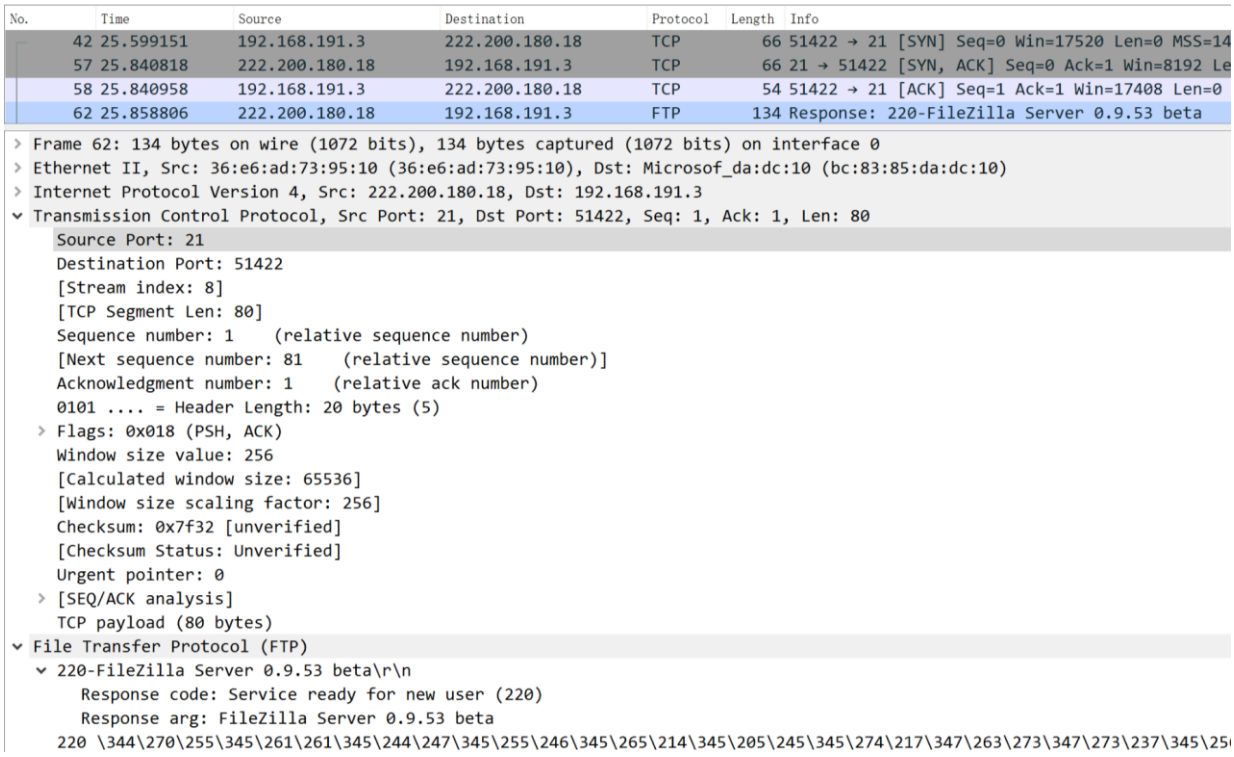
(2) 浏览器

FTP-WEB



追踪 TCP 流

1） 控制连接创建



前面三帧是客户端与主机的三次握手。

第四帧是服务器端的响应报文，并以 ASCII 码方式明文传输数据，这一帧表示控制连接完成。

2） 数据连接创建

PWD 表示客户端确认并请求访问文件目录，请求主机现实用户所在的路径。

下一个数据包表示服务器响应请求，数据连接建立

No.	Time	Source	Destination	Protocol	Length	Info
200	39.553616	192.168.191.3	222.200.180.18	FTP	60	Request: SYST
201	39.590104	222.200.180.18	192.168.191.3	FTP	86	Response: 215 UNIX emulated by FileZilla
202	39.590418	192.168.191.3	222.200.180.18	FTP	59	Request: PWD
203	39.598271	222.200.180.18	192.168.191.3	FTP	85	Response: 257 "/" is current directory.

```
Source Port: 21
Destination Port: 51446
[Stream index: 33]
[TCP Segment Len: 31]
Sequence number: 163    (relative sequence number)
[Next sequence number: 194    (relative sequence number)]
Acknowledgment number: 40    (relative ack number)
0101 .... = Header Length: 20 bytes (5)
> Flags: 0x018 (PSH, ACK)
Window size value: 256
[Calculated window size: 65536]
[Window size scaling factor: 256]
Checksum: 0x90c2 [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
> [SEQ/ACK analysis]
TCP payload (31 bytes)
v File Transfer Protocol (FTP)
  > 257 "/" is current directory.\r\n
```

3） FTP 数据传送

PASV 模式，是客户端通过 PASV 命令告诉服务器端，向用被动方式传输数据，服务器收到命令后在服务器端建立数据端口的 TCP/IP 监听，并把端口号返回客户端，客户端通过连接此端口进行数据传送。

No.	Time	Source	Destination	Protocol	Length	Info
209	39.624688	222.200.180.18	192.168.191.3	FTP	101	Response: 250 CWD successful. "/" is current directory.
210	39.624937	192.168.191.3	222.200.180.18	FTP	60	Request: PASV
211	39.629804	222.200.180.18	192.168.191.3	FTP	104	Response: 227 Entering Passive Mode (222,200,180,18,211,3)

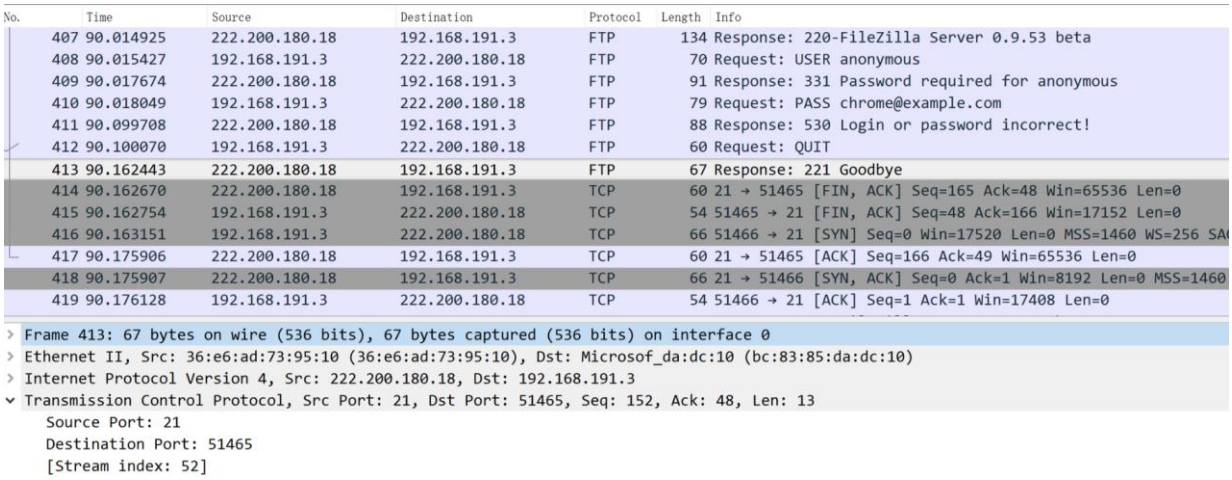
```
v Transmission Control Protocol, Src Port: 21, Dst Port: 51446, Seq: 280, Ack: 69, Len: 50
  Source Port: 21
  Destination Port: 51446
  [Stream index: 33]
  [TCP Segment Len: 50]
  Sequence number: 280    (relative sequence number)
  [Next sequence number: 330    (relative sequence number)]
  Acknowledgment number: 69    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
  Window size value: 256
  [Calculated window size: 65536]
  [Window size scaling factor: 256]
  Checksum: 0xa4a3 [unverified]
  [Checksum Status: Unverified]
  Urgent pointer: 0
  > [SEQ/ACK analysis]
  TCP payload (50 bytes)
v File Transfer Protocol (FTP)
  v 227 Entering Passive Mode (222,200,180,18,211,3)\r\n
```

4） FTP 指令传送和响应

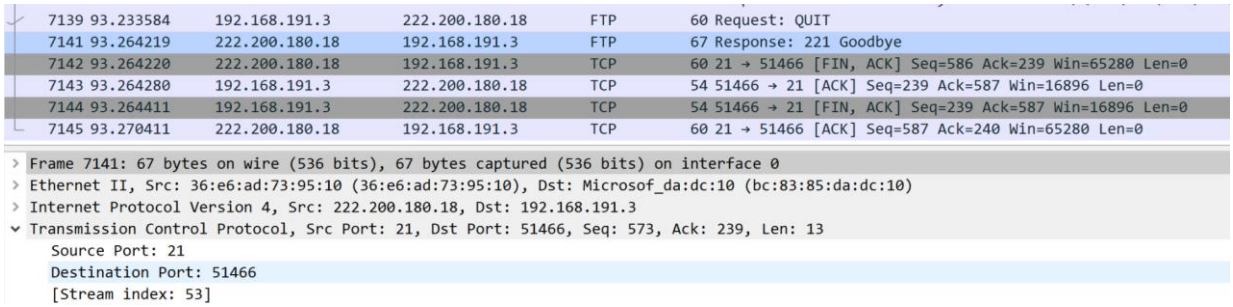
435	90.230074	192.168.191.3	222.200.180.18	FTP	60	Request: PASV
436	90.235278	222.200.180.18	192.168.191.3	FTP	105	Response: 227 Entering Passive Mo

```
> Frame 436: 105 bytes on wire (840 bits), 105 bytes captured (840 bits) on interface 0
> Ethernet II, Src: 36:e6:ad:73:95:10 (36:e6:ad:73:95:10), Dst: Microsof_da:dc:10 (bc:83:85:da:dc:10)
> Internet Protocol Version 4, Src: 222.200.180.18, Dst: 192.168.191.3
v Transmission Control Protocol, Src Port: 21, Dst Port: 51466, Seq: 321, Ack: 173, Len: 51
    Source Port: 21
    Destination Port: 51466
    [Stream index: 53]
```

5）数据连接的释放



6）控制连接的释放



2.2 实验问题

(1) 对 FTP-DOS 报文进行分析，找到 TCP 三次握手后的第一个 FTP 报文，分析并填写下表

FTP报文格式分析

源IP地址	222.200.180.18	源端口	21
目的IP地址	192.168.191.3	目的端口	51185
FTP字段	字段值	字段所表达的信息	
Response Code	Service ready for new user(220)	准备好为新用户提供服务	
Response Arg	FileZilla Server 0.9.32 beta	ftp服务已经就绪	

(2) 在 FTP-DOS 中找出 FTP 指令传送和响应的报文，分析并填写下表

FTP指令和响应过程分析

过程	指令/响应	报文号	报文信息
User	Request	15	连接用户
	Response	145	请求用户密码
Password	Request	29	传送密码
	Response	180	用户登录
Quit	Request	43	请求断开连接
	Response	195	连接断开

(3) 对 FTP-WEB 捕获的报文进行综合分析，观察 FTP 协议的工作过程。特别观察两种连接的建立过程和释放过程，以及这两种连接建立和释放的先后顺序，将结果填入下表。

FTP传送过程中的报文

报文类型	所包括的报文序号	客户端口	服务器端口
控制连接的建立	8	51422	21
数据连接的建立	163	51446	21
FTP数据传送	33	51446	21
FTP指令传送和响应	53	51466	21
数据连接的释放	52	51465	21
控制连接的释放	53	51466	21

(4)从协议层面分析 FTP-DOS 与 FTP-WEB 的异同。

服务器端口都是 21，

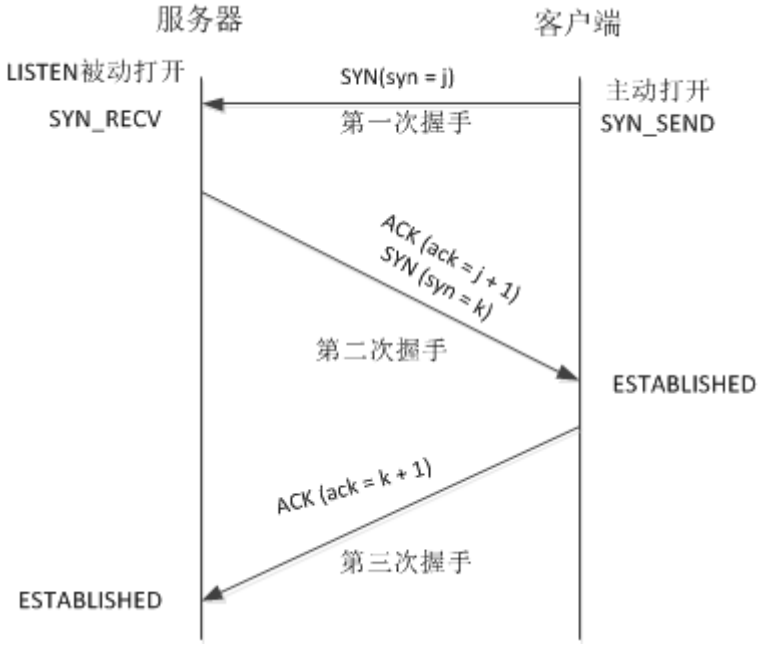
WEB 模式对应 PASV 方式进行数据连接，DOS 模式对应 PORT 方式进行数据连接。

(5) 在步骤 5 中，FTP 中的匿名账户是什么？

匿名账户：Anonynous

(6) 叙述 TCP 连接建立的三次握手的过程，四次挥手终止连接的过程。

i) 三次握手



第一次握手：客户端向服务器端发送连接请求包 SYN（syn=j），等待服务器回应；

第二次握手：服务器端收到客户端连接请求包 SYN（syn=j）后，将客户端的请求包 SYN（syn=j）放入到自己的未连接队列，此时服务器需要发送两个包给客户端；

（1）向客户端发送确认自己收到其连接请求的确认包 ACK（ack=j+1），向客户端表明已知道了其连接请求

（2）向客户端发送连接询问请求包 SYN（syn=k），询问客户端是否已经准备好建立连接，进行数据通信；

即在第二次握手时服务器向客户端发送 ACK（ack=j+1）和 SYN（syn=k）包，此时服务器进入 SYN_RECV 状态。

第三次握手：客户端收到服务器的 ACK（ack=j+1）和 SYN（syn=k）包后，知道了服务器同意建立连接，此时需要发送连接已建立的消息给服务器；

向服务器发送连接建立的确认包 ACK（ack=k+1），回应服务器的 SYN（syn=k）告

诉服务器，我们之间已经建立了连接，可以进行数据通信。

ACK(ack=k+1)包发送完毕,服务器收到后,此时服务器与客户端进入 ESTABLISHED 状态，开始进行数据传送。

ii) 四次挥手

第一次挥手：

Client 发送一个 FIN，用来关闭 Client 到 Server 的数据传送，Client 进入 FIN_WAIT_1 状态。

第二次挥手：

Server 收到 FIN 后，发送一个 ACK 给 Client，确认序号为收到序号+1（与 SYN 相同，一个 FIN 占用一个序号），Server 进入 CLOSE_WAIT 状态。

第三次挥手：

Server 发送一个 FIN，用来关闭 Server 到 Client 的数据传送，Server 进入 LAST_ACK 状态。

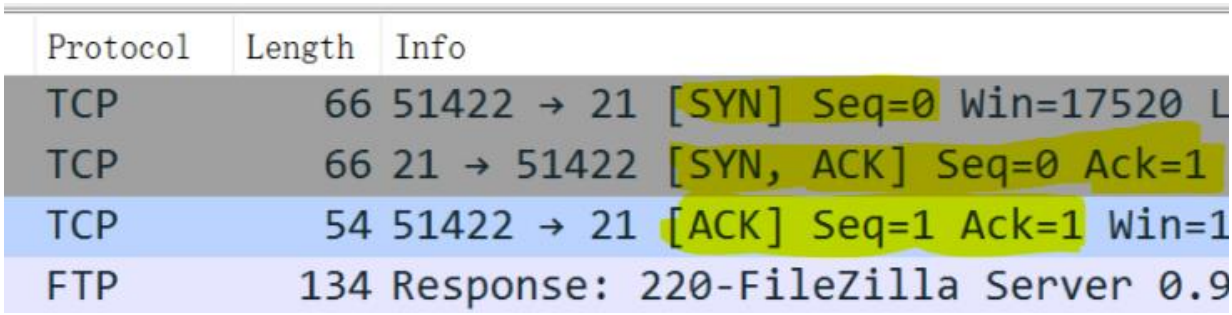
第四次挥手：

Client 收到 FIN 后,Client 进入 TIME_WAIT 状态,接着发送一个 ACK 给 Server，确认序号为收到序号+1，Server 进入 CLOSED 状态，完成四次挥手。

(7) 从捕获的数据包分析三次握手的过程，四次挥手终止连接的过程

i) 三次握手

Time	Source	Destination	Protocol	Length	Info
42 25.599151	192.168.191.3	222.200.180.18	TCP	66	51422 → 21 [SYN] Seq=0 Win=17520 Len=0 MSS=1460 WS=256 SACK_PERM=1
57 25.840818	222.200.180.18	192.168.191.3	TCP	66	21 → 51422 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
58 25.840958	192.168.191.3	222.200.180.18	TCP	54	51422 → 21 [ACK] Seq=1 Ack=1 Win=17408 Len=0
62 25.858806	222.200.180.18	192.168.191.3	FTP	134	Response: 220-FileZilla Server 0.9.53 beta



第一次握手：建立连接时，客户端 A 发送 SYN 包(seq=0)到服务器 B，并进入 SYN_SEND 状态，等待服务器 B 确认。

第二次握手：服务器 B 收到 SYN 包，必须确认客户 A 的 SYN(ACK=seq+1)，同时自己也发送一个 SYN 包(seq=0)，即 SYN+ACK 包，此时服务器 B 进入 SYN_RECV 状态。

第三次握手：客户端 A 收到服务器 B 的 SYN＋ACK 包(seq=1)，向服务器 B 发送确认包 ACK(ACK=1)，此包发送完毕，客户端 A 和服务器 B 进入 ESTABLISHED 状态，完成三次握手。

ii) 四次挥手

Source	Destination	Protocol	Length	Info
222.200.180.18	192.168.191.3	FTP	67	Response: 221 Goodbye
222.200.180.18	192.168.191.3	TCP	60	21 → 51466 [FIN, ACK] Seq=586 Ack=239 Win=65280 Len=0
192.168.191.3	222.200.180.18	TCP	54	51466 → 21 [ACK] Seq=239 Ack=587 Win=16896 Len=0
192.168.191.3	222.200.180.18	TCP	54	51466 → 21 [FIN, ACK] Seq=239 Ack=587 Win=16896 Len=0
222.200.180.18	192.168.191.3	TCP	60	21 → 51466 [ACK] Seq=587 Ack=240 Win=65280 Len=0

第一次挥手：

客户端给服务器发送 TCP 包，用来关闭客户端到服务器的数据传送。

Seq = 586,Ack = 239

第二次挥手：
服务器收到 FIN 后，发回一个 ACK。
Seq = 239,Ack = 587
第三次挥手：
服务器关闭与客户端的连接，发送一个 FIN。
Seq = 239,Ack = 587
第四次挥手：
客户端收到服务器发送的 FIN 之后，发回 ACK 确认。
Seq = 587,Ack = 240

2.3 实验思考

(1) 分析 FTP 使用的两个 TCP 连接，具体指出哪些情况下使用数据连接，哪些情况下使用控制连接。

连接 FTP 下载或上传数据时使用控制连接，显示路径等控制功能都由控制连接实现。

(2) 比较 FTP 协议和 HTTP 协议（部分资料来源于网络）

TCP/IP 协议是传输层协议，主要解决数据如何在网络中传输，而 HTTP 是应用层协议，主要解决如何包装数据。关于 TCP/IP 和 HTTP 协议的关系，网络有一段比较容易理解的介绍：“我们在传输数据时，可以只使用（传输层）TCP/IP 协议，但是那样的话，如果没有应用层，便无法识别数据内容，如果想要使传输的数据有意义，则必须使用到应用层协议，应用层协议有很多，比如 HTTP、FTP、TELNET 等，也可以自己定义应用层协议。WEB 使用 HTTP 协议作应用层协议，以封装 HTTP 文本信息，然后使用 TCP/IP 做传输层协议将它发到网络上。”

术语 TCP/IP 代表传输控制协议/网际协议，指的是一系列协议。“IP”代表网际协议，TCP 和 UDP 使用该协议从一个网络传送数据包到另一个网络。把 IP 想像成一种高速公路，它允许其它协议在上面行驶并找到到其它电脑的出口。TCP 和 UDP 是高速公路上的“卡车”，它们携带的货物就是像 HTTP，文件传输协议 FTP 这样的协议等。

TCP 和 UDP 是 FTP，HTTP 和 SMTP 之类使用的传输层协议。虽然 TCP 和 UDP 都是用来传输其他协议的，它们却有一个显著的不同：TCP 提供有保证的数据传输，而 UDP 不提供。这意味着 TCP 有一个特殊的机制来确保数据安全的不出错的从一个端点传到另一个端点，而 UDP 不提供任何这样的保证。

HTTP(超文本传输协议)是利用 TCP 在两台电脑(通常是 Web 服务器和客户端)之间传输信息的协议。客户端使用 Web 浏览器发起 HTTP 请求给 Web 服务器，Web 服务器发送被请求的信息给客户端。

1) HTTP 协议是用来浏览网站的，而 FTP 是用来访问和传输文件的，FTP 文件传输有点批量上传和维护网站的意思，而 HTTP 文件传输更多的是为终端用户提供文件传输，比如电影、图片、音乐之类。

2) HTTP 和 FTP 客户端：通常的 HTTP 客户端就是浏览器，而 FTP 服务可以通过命令行或者用户自有的图形界面客户端。

3) HTTP 头：HTTP 头包含了 metadata ，比如说最后更改的日期、编码方式、服务器名称版本还有其他的一些信息，而这些在 FTP 中是不存在的。

4) 数据格式：FTP 能传输 ACSII 数据或者二进制格式的数据，而 HTTP 只用二进制格式。

5) HTTP 中的流水线：HTTP 支持流水线，这就意味着客户端可以在上一个请求处理完之前，发出下一个请求，其结果就是多次请求数据之前省掉了部分服务器客户端往返时延。而 FTP 并没有这项支持。

6) HTTP 中的动态端口：FTP 一个最大的问题就是它使用两个连接，第一个连接用来

发送控制指令，当接受或者发送数据的时候，又打开第二个 TCP 连接。而 HTTP 在双向传输中使用动态端口。

7) HTTP 中的持久连接：对一个 HTTP 会话来讲，客户端可以维护一个单个的连接并使用它进行任意数量的数据传输。FTP 每次有数据的需要时都创建一个新的连接。重复的创建新的连接带来的体验并不好，因为每次创建连接都必须让双方握手验证，这消耗了很多时间。

8) HTTP 中的压缩算法：HTTP 提供了一个在一些压缩算法中客户端和服务器共同协商选择的办法。其中 gzip 可以说是最有影响力的一种，而 FTP 中并不存在这种复杂的算法。

9) HTTP 支持代理：HTTP 一个很大的特点就是支持代理，这种功能是构建在协议里的，而 FTP 并不支持。

10) 而 FTP 也能脱颖而出的一点是这个协议是直接面向文件级别的。这以为着 FTP 有例如可以通过命令列出远程服务器上的目录列表，而 HTTP 没有这个概念。

11) 速度：可能最通常的问题了：哪一个传输更快？

FTP 服务更快:

1、没有在发出的数据中加入 meta-data，仅传输原二进制文件。

2、没有过度的分块编码

HTTP 服务更快:

1、重用已存在的持久连接，从而有更好的 TCP 表现。

2、流水线的支持使得从同一个服务器上请求多个文件更快。

3、自动的压缩机制使得传输的数据更少。

4、没有命令/应答机制最大限度的减少了往返时延。

(3)讨论 FTP 协议的安全问题

1) 明文口令

由于 TCP/IP 协议族的设计在相互信任和安全的基础上的,FTP 的设计也没有采用加密传送，FTP 客户与服务器之前所有的数据传送都是通过明文的方式，当然也包括了口令。

自从有了交换环境下的数据监听技术之后，这种明文传送就变得十分危险，因为别人可能从传输过程过捕获一些敏感的信息，如用户名和口令等。像 HTTPS 和 SSH 都采用加密解决了这一问题。而 FTP 仍然是明文传送，而像 UINX 和 LINUX 这类系统的 ftp 账号通常就是系统帐号。这样黑客就可以通过捕获 FTP 的用户名和口令来取得系统的帐号,如果该帐号可以远程登录的话，通常采用本地溢出来获得 root 权限。这样该 FTP 服务器就被黑客控制了。

2) 易遭受穷举攻击

破解FTP口令必须首先获得对方的用户名,而获得对方的用户名的方法有很多种，例如可以使用社会工程学来骗取用户名，或者使用 Finger 命令来得到用户名，甚至可以猜测对方的用户名。

3) 在获得对方的用户名后，就可以开始对密码进行破解了。不管你通过什么途径得到了系统的用户名，破解密码的过程都是穷举密码的过程，而穷举密码的过程说通俗了就是使用不同的密码进行登陆，直到试出正确的密码。 3、任意用户均可登陆 FTP 服务器

由于 FTP 协议的特点，互联网上的任意用户均可登陆 FTP 服务器，某个非法用户在获得 FTP 服务器的用户名和密码后，如果用户能够向 FTP 服务器传送文件，那么非法用户可以向 FTP 服务器发送大量的没有的文件，造成 FTP 服务器工作的瘫痪。

(4) 启用 FileZilla 创建的 FTP Server 的口令安全，通过捕获数据包分析它能否保证用户名和口令的安全。

不能

(5) 同一台主机，既作为 FTP 服务器，又充当客户端（非虚拟机形式），如何捕获 FTP 数据包？

使用当前较为流行的一种计算机网络调试和数据包嗅探软件 ethereal。

当一个端口为 21 的 tcp 数据流来到时。首先由 tcp 协议注册的解析模块处理，处理完之后通过查找协议树找到自己协议下面的子协议，判断应该由那个子协议来执行，找到正确的子协议后，就转交给 ftp 注册的解析模块处理。这样由根节点开始一层层解析下去。