四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

学号：2022141460176 姓名：杨一舟 专业：计算机科学与技术 第 8 周

课程名称	计算机网络课程设计	实验课时	2 课时
实验项目	TCP 链接管理与 UDP 协议分析	实验时间	2024 年 10 月 24 日
实验目的	使用 wire shark 捕获 TCP 与 UDP 的数据包，分析 UDP 协议特点与原理，了解 TCP 连接建立与释放的机制		
实验环境	Windows 11		
实验内容（算法、程序、步骤和方法）	**一、UDP 协议分析** 第一步:UDP 数据包的捕获; 在 Windows 系统下,可以在命令提示符中输入 ipconfig/flushdns 命令来清空 DNS 缓存。 • 1)打开 Wireshark,启动分组捕获器; • 2)在命令行中输入:ping cs.scu.edu.cn,并回车; • 3)停止分组捕获; • 4)并在过滤器中输入“udp and dns” （为保证清晰将已将截图反色处理）		

```
C:\Users\MountainMist>ipconfig/flushdns

Windows IP 配置

已成功刷新 DNS 解析缓存。

C:\Users\MountainMist>ping cs.scu.edu.cn

正在 Ping cs.scu.edu.cn [202.115.32.43] 具有 32 字节的数据:
来自 202.115.32.43 的回复: 字节=32 时间=5ms TTL=61
来自 202.115.32.43 的回复: 字节=32 时间=1ms TTL=61
来自 202.115.32.43 的回复: 字节=32 时间=3ms TTL=61
来自 202.115.32.43 的回复: 字节=32 时间=1ms TTL=61

202.115.32.43 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 = 1ms, 最长 = 5ms, 平均 = 2ms

C:\Users\MountainMist>
```

第二步:UDP 数据包的分析。

1）从截获的数据包中找一个 UDP 数据报进行分析。从中可以看出 UDP 协议的头部包含几个字段?分别是什么?头部总共多少字节?

包含了四个字段，分别是源端口号、目的端口号、长度、校验和。

头部共有 8 个字节，一共四个字段，每个字段的长度都是 2 个字节。

（payload 则是指该报文传递的数据）

```
User Datagram Protocol, Src Port: 53, Dst Port: 59591
    Source Port: 53
    Destination Port: 59591
    Length: 148
    Checksum: 0xfcf0 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 1]
    [Stream Packet Number: 4]
  v [Timestamps]
      [Time since first frame: 0.309982000 seconds]
      [Time since previous frame: 0.002589000 seconds]
    UDP payload (140 bytes)
```

2）UDP 协议头部中的 Length 字段的含义是什么?

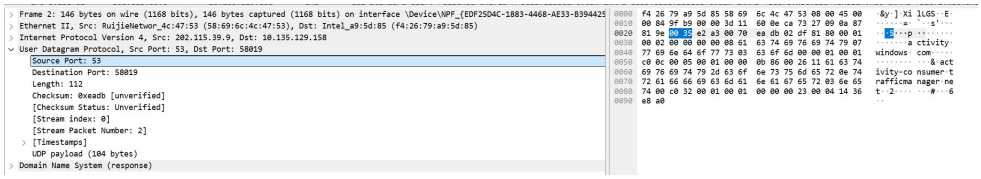
是指 UDP 头部和 UDP 数据的字节长度。因为 UDP 头部长度为 8 字节，所以该字段的最小值为 8。

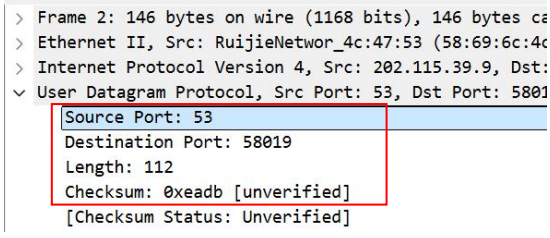
3）从 Wire shark 的数据区域,UDP 头部各个字段对应的 16 进制的编码分别是什么?

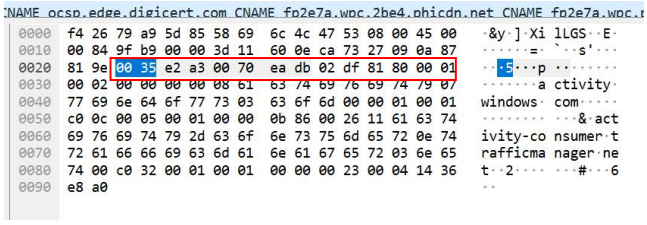
各个字段对应的编码依次分别是它们十进制数的 16 进制表示

源端口号 $53_{10\text{进制}}$为 $0035_{16\text{进制}}$、目的端口号 $58019_{10\text{进制}}$为 $E2A3_{16\text{进制}}$

报文长度 $112_{10\text{进制}}$为 $0070_{16\text{进制}}$、校验和本身即为 16 进制 EADB







4）还可以通过什么方式获取 UDP 协议的数据包?

还可以使用 Python 的 scapy 库编写代码来获取

```python
from scapy.all import *

def print_pkt(pkt):
    if UDP in pkt:
        print("Source IP: ", pkt[IP].src)
        print("Destination IP: ", pkt[IP].dst)
        print("Source Port: ", pkt[UDP].sport)
        print("Destination Port: ", pkt[UDP].dport)
        print("Length of Data: ", len(pkt[UDP].payload))
        print("Data: ", pkt[UDP].payload.load.decode('utf-8', errors='ignore'))
        print()

# 设置监听接口
interface = "eth0"

# 开始捕获UDP数据包
print("Sniffing for UDP packets...")
sniff(iface=interface, filter="udp", prn=print_pkt, store=0)
```

（接上）实验内容（算法、程序、步骤和方法）

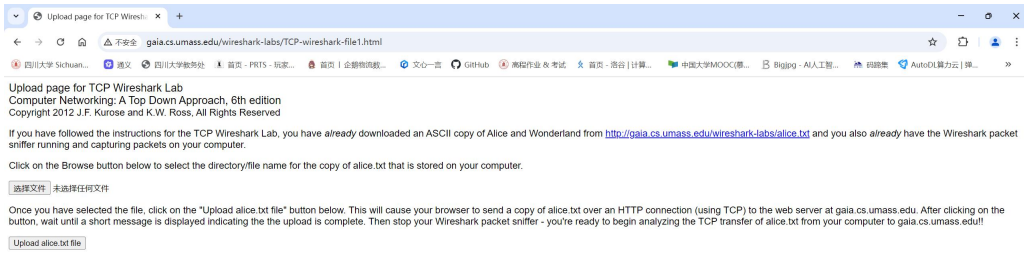
二、TCP 连接管理分析

1、TCP 会话过程数据包的捕获

1)打开 Wireshark,启动 Wireshark 分组俘获器;

2)在 WEB 浏览器地址栏中输入 http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html 并回车

3)待获取完整页面以后,停止分组捕获;

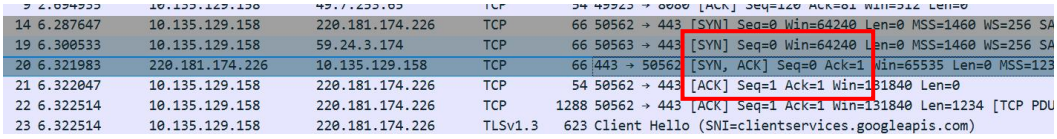


2、TCP 数据包的分析

1)从捕获的数据包中,找出三次握手建立连接的数据包

三次握手连接的数据包分别是

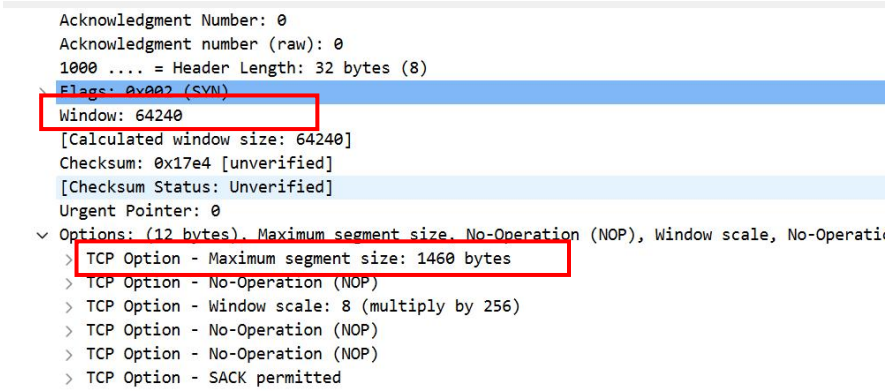
“[SYN]Seq=0” “[SYN,ACK]Seq=1 Ack=1” “[ACK]Seq=1”



2) 从找到的三次握手数据包中观察,客户端协商的 MSS 为多少?客户端接收窗口大小?

客户端协商 MSS(Maximum Segment Size)为 1460 bytes

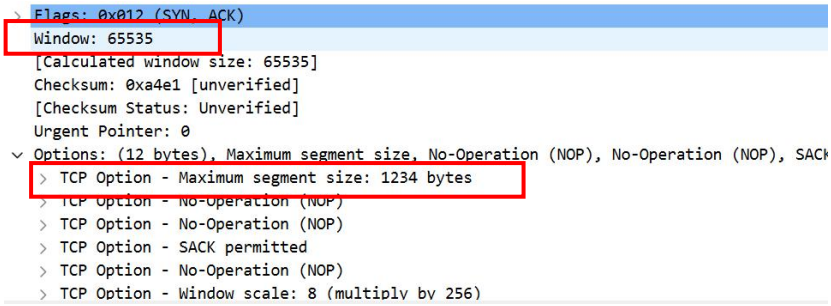
客户端接收窗口(Window)大小为 64240



3) 服务器协商的 MSS 为多少?服务器端接收窗口大小为多少?

服务器协商 MSS(Maximum Segment Size)为 1234 bytes

服务器接收窗口(Window)大小为 65535



4) 在传输过程中,客户端和服务器传输数据时的 MSS 为多少?

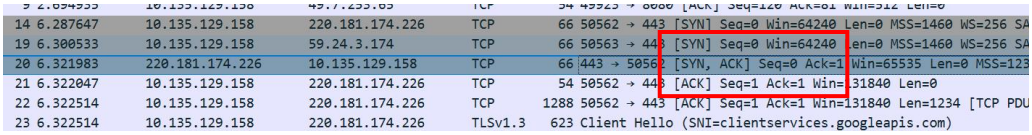
传输过程中通常会使用二者中较小的 MSS，即 1234 bytes

5）说明在三次握手过程中，数据包的序号，确认号，SYN 标志位，ACK 标志位的变化？

对于[SYN]，序号 Seq=0，确认号无，SYN 标志置位

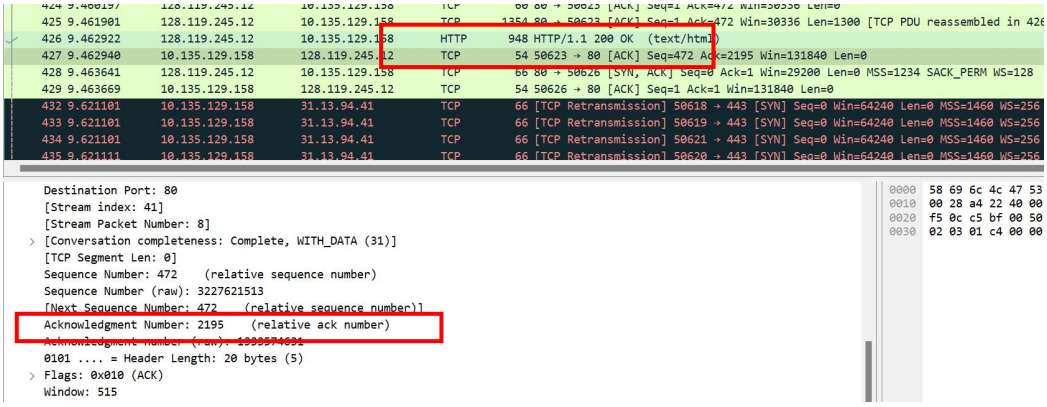
对于[SYN,ACK]，序号 Seq=0，确认 Ack=1，SYN 与 ACK 标志置位

对于[ACK]，序号 Seq=1，确认 Ack=1，ACK 标志置位



6）当客户端发送了 HTTP 请求报文以后，客户端收到服务器的 ACK 为多少？

收到的 ACK 是 2195（relative ack number）



7）在捕获的数据包中是否有窗口更新报文，如果有，请问在什么情况下会产生窗口更新报文？

没有窗口更新报文，通常在接收缓冲区变得可用时才会产生

8）从捕获的数据包中，找到挥手释放连接的数据包。

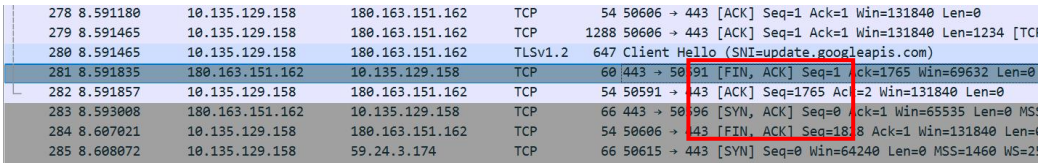
TCP 连接的关闭需要四个步骤来完成：

第一次挥手：一方发送一个 FIN（终止）标志的数据包给另一方，表示数据发送完毕，希望关闭连接。

第二次挥手：接收方发送一个 ACK 数据包确认收到 FIN，但此时接收方可能还有未处理完的数据，所以不会立即关闭连接。

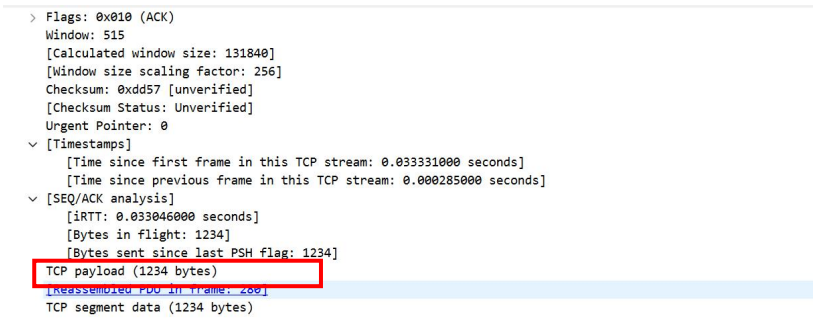
第三次挥手：接收方处理完所有数据后，也会发送一个 FIN 标志的数据包给原发送方，表示同意关闭连接。

第四次挥手：原发送方收到 FIN 后，发送一个 ACK 数据包确认，然后双方进入关闭状态。



9）在这个 TCP 的会话过程中，服务器一共给客户端传送了多少应用层数据？

共传递了 1234 bytes 的数据，因为 payload 是指该报文传递的数据量



数据记录和计算	所捕获的数据包均如上图所示

结论（结果）	本次实验中，我们观察到了三次握手和四次挥手的过程。实验结果表明，TCP 协议的设计能够有效应对复杂的网络环境，保证数据传输的质量，而 UDP 不需要经历 TCP 那样的复杂握手过程即可直接发送数据报文。UDP 协议虽然简单且不保证数据的可靠送达，但它在特定的应用场景下有着不可替代的优势，尤其是在那些对延迟敏感而不那么关心数据完整性的应用中。
小结	本次实验让我对 TCP 与 UDP 的实际运作有了更为直观的感受。特别是通过观察 TCP 的三次握手和四次挥手过程，我深刻体会到 TCP 协议为了确保数据可靠传输的精巧设计，每一个细节都有其存在的必要性。与此同时，我也认识到 UDP 协议虽然看似简单，却在特定的应用场景中展现出其独特的优势。这次实验不仅巩固了我的理论知识，也使我能够在今后的学习中更好地理解和运用网络协议。
指导老师评议	成绩评定：　　指导教师签名：