计算机网络

实验报告

实验名称：tcp 抓包

实验地点：233

实验日期：2023.12.19

学生姓名：李泽祥

学生学号：21020007048

一、实验目的

（1）通过实验熟悉 Wireshark 抓包软件的使用方法，理解 TCP 传输过程，以及慢启动、拥塞避免等相关技术。

（2）通过实验熟悉 Wireshark 抓包软件的使用方法，理解 TCP 传输过程，以及慢启动、拥塞避免等相关技术。熟悉在两台计算机之间进行 TCP 传输的过程。

……

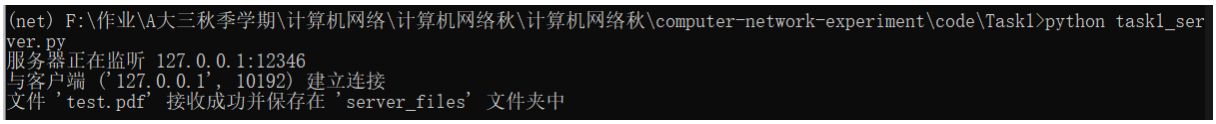
二、实验内容

（一）实验一

说明你的实验步骤并提供截图证明

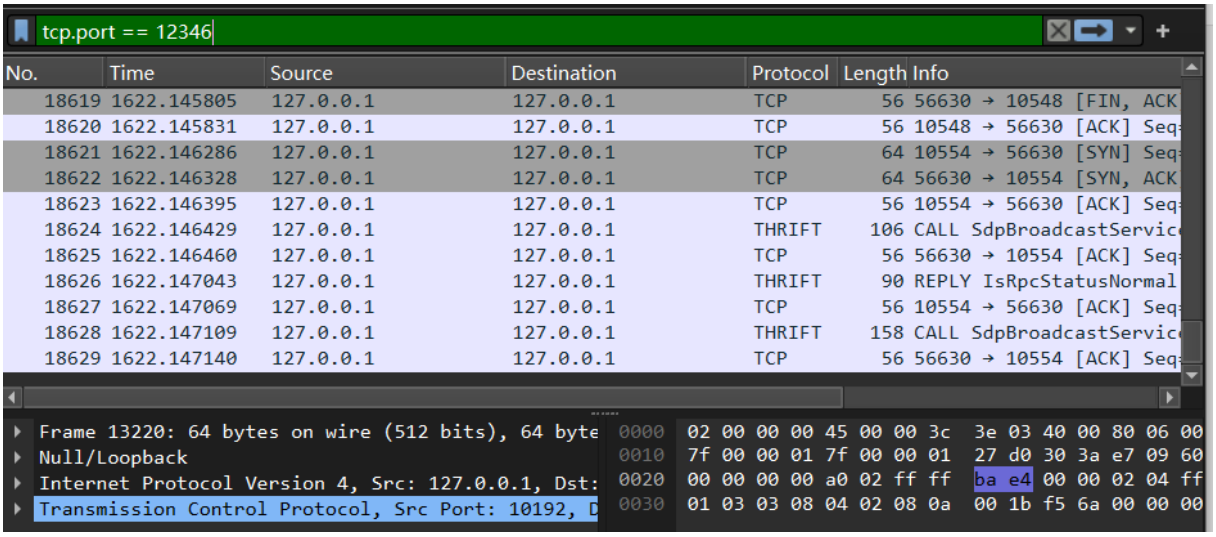
1. 第一步：安装 wireshark，并配置 python 环境

2. 第二步：运行 task1_server.py 监听 12346 端口，接着运行 task1_client.py 实现建立连接



成功发送 test.pdf

3. 第三步：用 wireshark 查看 12346 端口的 tcp 抓包情况



……

（二）实验二

说明你的实验步骤并提供截图证明

组队：李泽祥、李迅

1. 第一步：查看校园网下的 ip 地址为 10.149.3.229

```
无线局域网适配器 WLAN:

   连接特定的 DNS 后缀 . . . . . . . :
   本地链接 IPv6 地址. . . . . . . . : fe80::b3d:956c:2126:c5c5%16
   IPv4 地址 . . . . . . . . . . . . : 10.149.3.229
   子网掩码 . . . . . . . . . . . . : 255.255.252.0
   默认网关. . . . . . . . . . . . . : 10.149.0.1
```

2. 第二步：先修改 client 处的 ip 为同学的 ip 地址。然后我作为客户端，成功发送 test.pdf

task2_client.py

```python
import socket

def main():
    # 创建一个TCP socket
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # 定义服务器的IP地址和端口号
    server_ip = "10.149.0.134"
    server_port = 12346
```

```
(net) F:\作业\A大三秋季学期\计算机网络\计算机网络秋\计算机网络秋\computer-network-experiment\code\Task2>python task2_client.py
请输入要发送的文件名: test.pdf
文件 'test.pdf' 发送成功
```

3. 第三步：先修改 server.py 处的 ip 为我校园网下的 ip 地址：我作为服务端，成功接收 test.pdf

```python
def main():
    # 创建一个TCP socket
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # 定义服务器的IP地址和端口号
    server_ip = "10.149.3.229"
    server_port = 12346
```

```
(net) F:\作业\A大三秋季学期\计算机网络\计算机网络秋\计算机网络秋\computer-network-experiment\code\Task2>python task2_server.py
服务器正在监听 10.149.3.229:12346
与客户端 ('10.149.0.134', 63351) 建立连接
文件 'test.pdf' 接收成功并保存在 'server_files' 文件夹中
```

回答下列问题（提供截图证明）

1. 在第一个实验中的客户端程序发送数据的端口是多少，这个端口是谁来决定的？

task1_client.py

```python
import socket

def main():
    # 创建一个TCP socket
    client_socket = socket.soc

    # 定义服务器的IP地址和端口号
    server_ip = "127.0.0.1"
    server_port = 12346
```

12346；我在 python 程序中决定的

2. 在第二个实验中服务端程序和客户端程序的 ip 地址和端口号是什么？

task2_client.py

```python
import socket

def main():
    # 创建一个TCP socket
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # 定义服务器的IP地址和端口号
    server_ip = "10.149.0.134"
    server_port = 12346
```

客户端如图

```python
def main():
    # 创建一个TCP socket
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # 定义服务器的IP地址和端口号
    server_ip = "10.149.3.229"
    server_port = 12346
```

服务端如图

3. 为什么在第一个实验中不需要设置端口的防火墙的通行权限？

   这两个程序之间的通信是基于本地主机（127.0.0.1）进行的，也就是在同一台计算机上运行的客户端和服务器程序之间的通信。由于它们位于同一台计算机上，所以不需要通过防火墙进行网络通信。防火墙主要用于控制网络流量和保护计算机免受未经授权的访问。当计算机上运行的程序尝试与外部网络进行通信时，防火墙通常会起到限制或允许该通信的作用。但是，在同一台计算

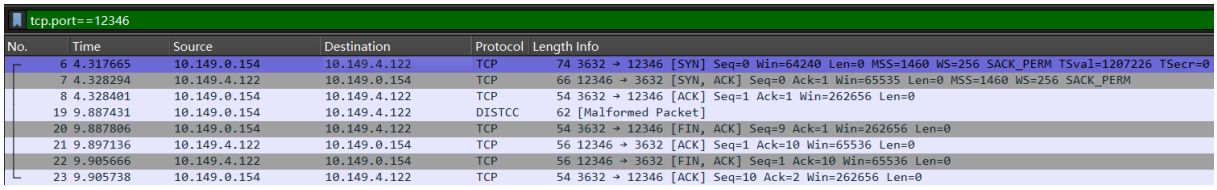
机上运行的程序之间进行通信时，数据传输是在本地进行的，不需要通过网络接口进行传输。

4. 在第二个实验中，用来初始化客户端和服务端的 TCP 连接的 TCP SYN 报文段的序号是什么？在报文段中，哪个地方表明这是一个 SYN 报文段？
（以下的 ip 和上面的不一样是因为又重新做了一遍，传了一个自己新建的 temp.txt 文件【无内容】）

```
No.     Time           Source                Destination           Protocol Length Info
      6 4.317665       10.149.0.154          10.149.4.122          TCP      74     3632 → 12346 [SYN] Seq=0 Win=64240 Len=0
MSS=1460 WS=256 SACK_PERM TSval=1207226 TSecr=0
Frame 6: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{A7A05F72-3692-4D3F-B6E1-50268E914716},
id 0
Ethernet II, Src: Intel_8a:84:2d (84:1b:77:8a:84:2d), Dst: IETF-VRRP-VRID_01 (00:00:5e:00:01:01)
Internet Protocol Version 4, Src: 10.149.0.154, Dst: 10.149.4.122
Transmission Control Protocol, Src Port: 3632, Dst Port: 12346, Seq: 0, Len: 0
No.     Time           Source                Destination           Protocol Length Info
      7 4.328294       10.149.4.122          10.149.0.154          TCP      66     12346 → 3632 [SYN, ACK] Seq=0 Ack=1 Win=65535
Len=0 MSS=1460 WS=256 SACK_PERM
Frame 7: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{A7A05F72-3692-4D3F-B6E1-50268E914716},
id 0
Ethernet II, Src: IETF-VRRP-VRID_01 (00:00:5e:00:01:01), Dst: Intel_8a:84:2d (84:1b:77:8a:84:2d)
Internet Protocol Version 4, Src: 10.149.4.122, Dst: 10.149.0.154
Transmission Control Protocol, Src Port: 12346, Dst Port: 3632, Seq: 0, Ack: 1, Len: 0
```

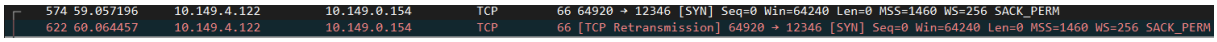
‘[SYN]’表示 SYN 报文段，序号是 Seq=0

5. 开始的 6 个 TCP 报文段的长度各自是多少？

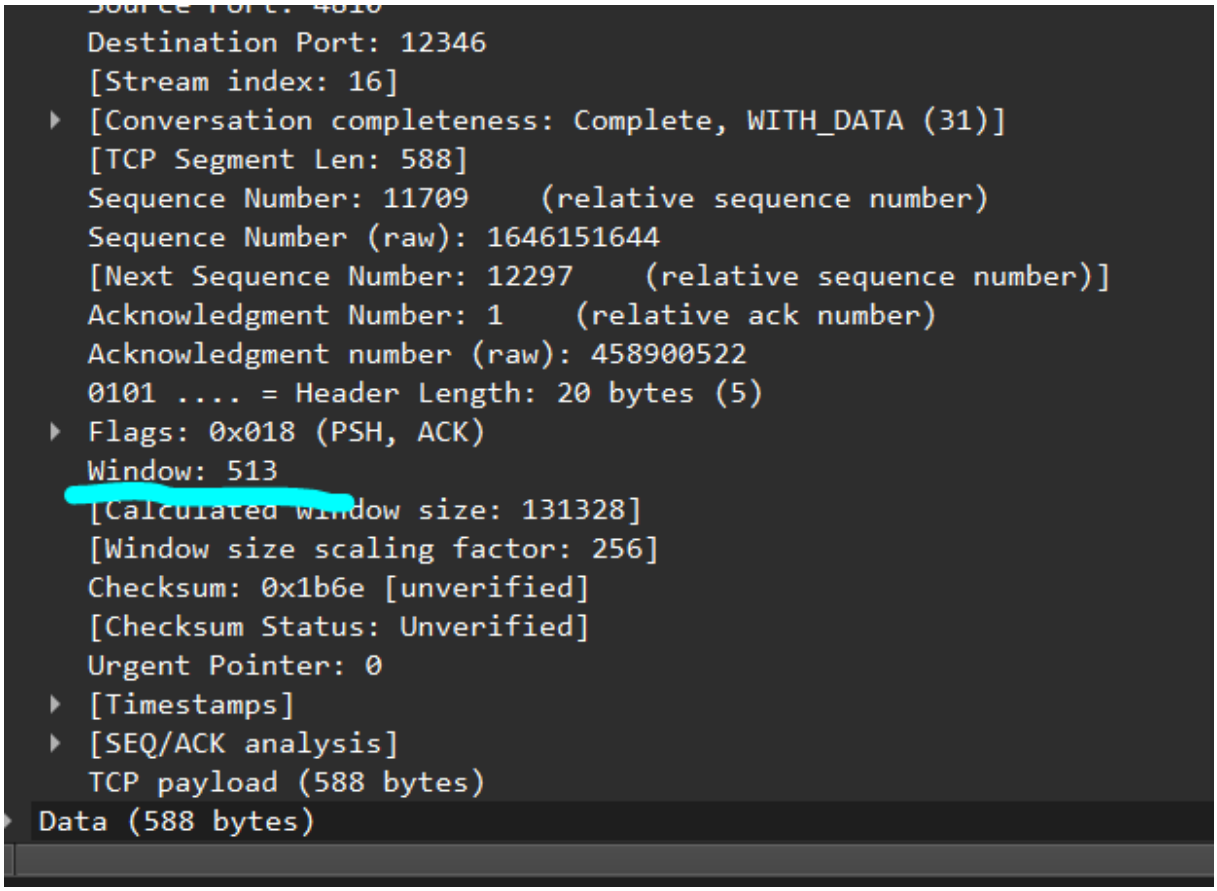


74、66、54、62、54、56、56、54

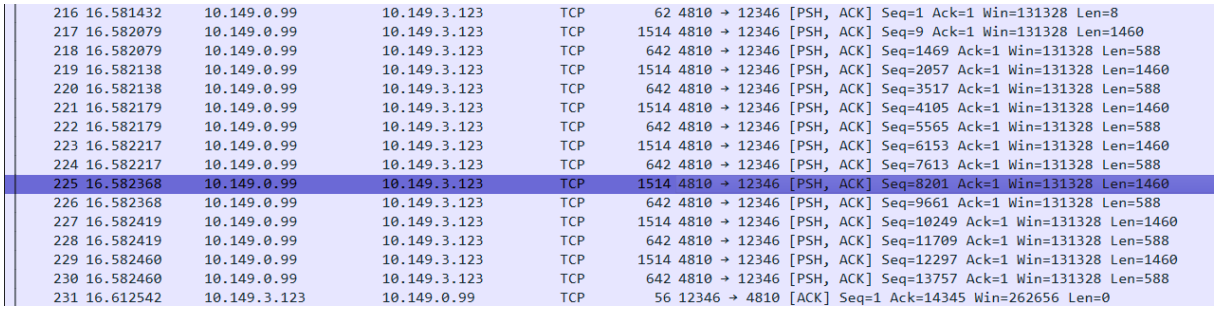
6. 在跟踪文件中，有重传的报文段么？回答这个问题，你需要检查哪个地方？
有重传；需要检查的是 TCP 报文段的 INFO 是否是 RETRANSMISSION（或者说在 wireshark 中是否是红色的条）



7. 接收方在一个 ACK 中，通常确认多少数据？你能辨别出这样一种情形吗：即接收方对收到的报文段，每隔一个确认一次？



确认 513bit



采用延迟确认，将多个 ACK 合并为一个

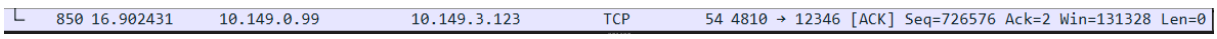
接收方发送的 ACK 报文中 Acknowledgement Number 连续增加，但确认号并不是连续的。

接收方发送的 ACK 报文中，ACK 标志位被设置为 1，表示确认收到数据。

接收方发送的 ACK 报文中，确认号指示了已成功接收的连续数据块的结束位置。

8. 这个 TCP 连接的吞吐量（每单位时间传输的字节数）是多少？解释你是如何计算这个数值的？

开始时间：16.581432



结束时间：16.902431

Seq=726576

所以吞吐量：726576/(16.902431-16.581432)= 2,263,483.686865068

三、实验体会

可根据“实验思考”部分作答，也可以根据个人具体体会作答。

这次实验让我对上课讲过的 tcp 相关内容，有了更深的认识，也顺便巩固了一下上课学过的内容。对于滑动窗口、慢启动、拥塞避免等技术更明白了一些。

通过实验一，我安装了 Wireshark，并配置了 Python 环境。然后我运行了 task1_server.py 和 task1_client.py 两个程序，成功建立了连接并发送了 test.pdf 文件。通过 Wireshark 查看 12346 端口的 TCP 抓包情况，我能够清楚地看到数据包的传输过程。
在实验二中，我与我的队友一起进行了实验。我们首先查看了校园网下的 IP 地址，并进行了相应的修改。我作为客户端成功发送了 test.pdf 文件，而我的队友作为服务端成功接收了文件。
回答问题方面，第一个实验中客户端程序发送数据的端口是 12346，这个端口是我在 Python 程序中决定的。在第二个实验中，服务端程序的 IP 地址是 10.149.3.229，端口号是 12346；客户端程序的 IP 地址是我的同学的 IP 地址，端口号未提供。
在第一个实验中不需要设置端口的防火墙通行权限，是因为客户端和服务器程序之间的通信是基于本地主机（127.0.0.1）进行的，即在同一台计算机上运行的程序之间的通信，不需要通过网络接口进行传输。
在第二个实验中，用来初始化客户端和服务端的 TCP 连接的 TCP SYN 报文段的序号是 0，并且在报文段中有一个标志位显示这是一个 SYN 报文段。
开始的 6 个 TCP 报文段的长度分别是 74、66、54、62、54 和 56。而在跟踪文件中，存在重传的报文段，可以通过检查 TCP 报文段的 INFO 是否是 RETRANSMISSION 来确定。
接收方在一个 ACK 中通常确认 513bit 的数据。同时，我也能辨别出接收方对收到的报文段每隔一个确认一次的情形，这是采用了延迟确认的策略，将多个 ACK 合并为一个。
最后，根据实验数据，我计算了这个 TCP 连接的吞吐量为 2,263,483.686865068（字节/秒），通过计算传输的数据量与传输时间的比值得出。