计算机网络实验四 TCP 协议的实现

彭程　2020011075
清华大学　自动化系　自 02 班

日期：2023 年 1 月 15 日

摘　　要

本文为 2022 秋《计算机网络及应用》实验四的实验报告。本次作业 python 编程实现了 TCP 协议。

关键词： 计算机网络，TCP

1　功能实现

本次作业实现了以下功能：

1. 三次握手建立连接（Fig.1）
2. 双向发送小规模数据，curl www.baidu.com，获得 html 报文
3. TCP 四次挥手，正确关闭（Fig.1）
4. 接受中等规模的数据: wget 和 curl 均可
5. 通过浏览器访问真实网页: 百度、知乎、bilibili 均可（Fig.2）
6. 超时重传机制（Fig.3）
7. 缓存乱序报文等待正确报文

2　功能实现思路

2.1　构建 TCP 报文

本次我使用了一个 TCPPacket 类，参考了这里, 可以实现校验和计算和构建合适的报文等功能。

2.2　三次握手和四次挥手

三次握手即我方发送 SYN，对方回复 ACK+SYN, 我方回复 ACK。此处 app_connnect 函数第一次握手由 main 调用，第三次握手由 tcp_rx 调用，第三次握手后 tcp_rx 调用 app_connnected 通知应用层。

四次挥手通常是我方接受完消息后发出 FIN，对方回应 ACK，对方发送 FIN，我方回应 ACK，之后经过等待终止连接。

2.3　状态维护

我维护一个全局的状态机 state_machine 字典，字典的 key 为 conn 的哈希值（因此有个 hash_conn 函数用来进行哈希），字典每个 value 为一个 state 类，包含 seq_num,ack_num 等必要的状态数据。这样实现了对于不同 conn 维护不同的状态，这对最终在浏览器打开网页有很大的帮助。

2.4 超时重传

我使用了提供的 tick 函数作为超时重传的载体，逻辑是如果是报文传递模式，如果一定时间内待发送报文的 ack 或者接收到报文的 ack 均未发生改变，则认为超时，进行重传。

2.5 乱序报文处理

在实验中发现报文会有乱序情况，于是对乱序报文进行处理是十分必要的。我采取的策略是，如果接受报文比需要的报文序号大，就进行缓存，当接收到需要报文时，在缓存中进行查找，若有后续报文则依次发送给应用层。

3 收获与总结

这次作业难度十分大，既没有往届的参考，网络上相关参考也很少。在我的不懈努力下终于较为圆满的完成了。在完成过程中当然遇到了一系列的问题：刚开始不会用 wireshark 对比，走了很多弯路；之后校验和算不对，又研究了半天；之后下载任务中，开始是 https 无法连接，之后又是传到一半会中断，最终在我使用了超时重传、乱序处理和缓存之后，终于可以保持一个较为稳定的下载（本地目前每次都可以成功）。在上传过程中一直收不到对面发来的 ack，还好该任务取消了。在浏览器打开网页时一直有奇怪的连接，在我重构了状态存储机制，允许对于不同连接进行不同状态维护后，终于可以成功的用浏览器打开各种网页。

本次实验收获主要在于：对 TCP 的细节认识更清楚了，学会了 wireshark 这一重要工具，对网络中可能存在的各种问题有了更多的认识（重传、丢包等）。超时重传和缓存机制是我认为本是实验中的亮点。

4 建议

以下是一些小小的建议：

1. 时间有些紧张。本次考完试后突然通知于一周内完成改实验，我从周二开始做，到周六下午基本完成，每天都做 5-6h，总共应该超过 20h。同期还有数图大作业、科研等任务，导致这一周十分紧张，打乱了一些原有计划。建议以后在布置该项作业时，告知大家预计耗时并适当的延长完成周期。
2. 不可知错误太多。这次作业最大的感受就是“卡”，经常在一个错误卡主，然后几个小时毫无进展，让同学们试错和提升自主解决问题的能力是可以理解的，毕竟在科研中没有人能一直帮你解决问题，但试错成本可以略微降低一点（尤其对于一个在总评中占比并不算高的作业，这个任务量和其他课的一些大作业不相上下），希望在文档中对容易出问题的地方进行更多的提示，对共性问题提供一些总结。

同时完全可以感受到本实验开发和答疑工作量真的很大，感谢助教提供了丰富的接口、详细的文档和耐心的答疑指导。希望以上意见对助教有所帮助。

附图

```
23.066373728  192.168.31.141  14.215.177.38    TCP      54 64241 → 80 [SYN] Seq=0 Win=65530 Len=0
23.104108726  14.215.177.38   192.168.31.141   TCP      60 80 → 64241 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=53
23.104896681  192.168.31.141  14.215.177.38    TCP      54 64241 → 80 [ACK] Seq=1 Ack=1 Win=65530 Len=0
23.119014618  192.168.31.141  14.215.177.38    HTTP    131 GET / HTTP/1.1
23.156955999  14.215.177.38   192.168.31.141   TCP      60 80 → 64241 [ACK] Seq=1 Ack=78 Win=29040 Len=0
23.156956570  14.215.177.38   192.168.31.141   HTTP   2835 HTTP/1.1 200 OK  (text/html)
23.159301048  192.168.31.141  14.215.177.38    TCP      54 64241 → 80 [ACK] Seq=78 Ack=2782 Win=65530 Len=0
23.197967027  192.168.31.141  14.215.177.38    TCP      54 64241 → 80 [FIN, ACK] Seq=78 Ack=2782 Win=65530 Len=0
23.250163522  14.215.177.38   192.168.31.141   TCP      60 80 → 64241 [ACK] Seq=2782 Ack=79 Win=29040 Len=0
23.250163963  14.215.177.38   192.168.31.141   TCP      60 80 → 64241 [FIN, ACK] Seq=2782 Ack=79 Win=29040 Len=0
23.261175683  192.168.31.141  14.215.177.38    TCP      54 [TCP Dup ACK 193#1] 64241 → 80 [ACK] Seq=79 Ack=2782 Wi
```

图 1: 三次握手和四次挥手

```
39.155.141.16    192.168.31.141   TCP       590 [TCP Retransmission] 443 → 63516 [ACK] Seq=1033965 Ack=922 Wi
39.155.141.16    192.168.31.141   TLSv1.2   590 Ignored Unknown Record
192.168.31.141   39.155.141.16    TCP        54 63516 → 443 [ACK] Seq=922 Ack=1034501 Win=65530 Len=0
39.155.141.16    192.168.31.141   TCP       590 [TCP Out-Of-Order] 443 → 63516 [ACK] Seq=1034501 Ack=922 Win=
39.155.141.16    192.168.31.141   TLSv1.2   590 Ignored Unknown Record
192.168.31.141   39.155.141.16    TCP        54 63516 → 443 [ACK] Seq=922 Ack=1035037 Win=65530 Len=0
192.168.31.141   39.155.141.16    TCP        54 [TCP Dup ACK 2062#1] 63516 → 443 [ACK] Seq=922 Ack=1035037 Wi
192.168.31.141   39.155.141.16    TCP        54 [TCP Dup ACK 2062#2] 63516 → 443 [ACK] Seq=922 Ack=1035037 Wi
192.168.31.141   39.155.141.16    TCP        54 [TCP Dup ACK 2062#3] 63516 → 443 [ACK] Seq=922 Ack=1035037 Wi
39.155.141.16    192.168.31.141   TCP       590 [TCP Retransmission] 443 → 63516 [ACK] Seq=1035037 Ack=922 Wi
```

图 2: 超时重传

图 3: 浏览器打开网页