计算机网络 实验总结-3

袁欣怡 2018K8009929021

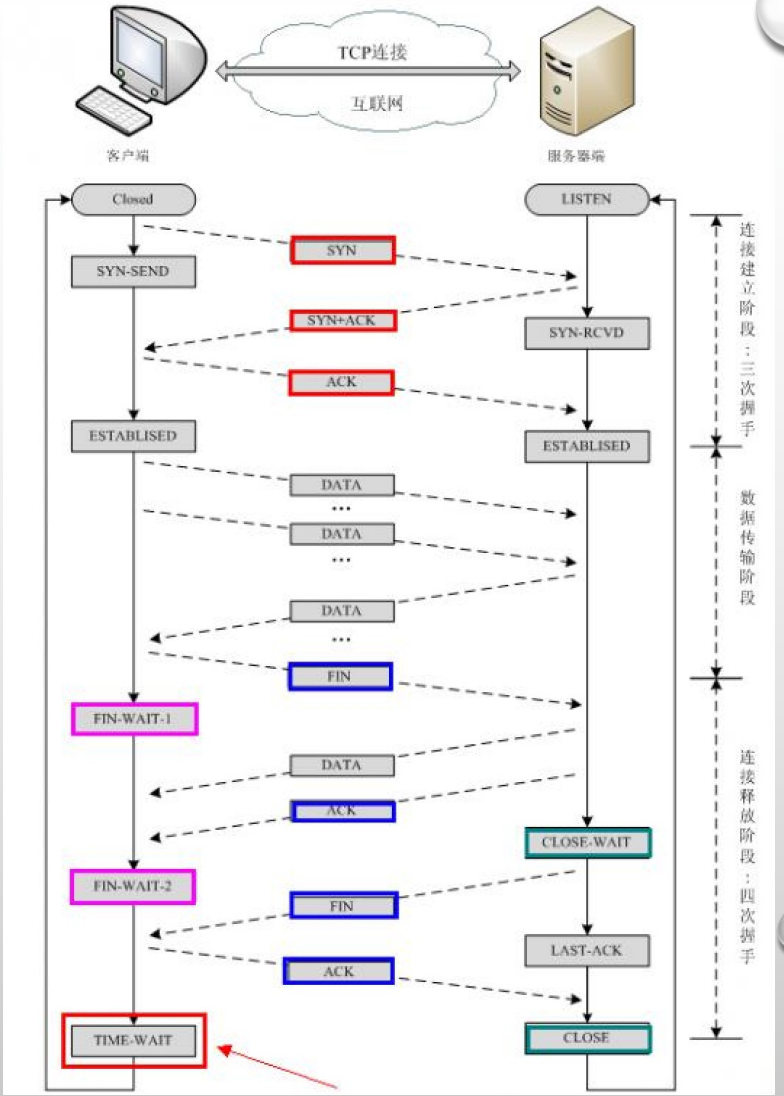
2021.7.13

TCP协议

- 传输控制协议TCP，是一种面向连接的、可靠的、基于字节流的传输层通信协议。在拥塞控制上，采用广受好评的TCP拥塞控制算法（也称AIMD算法）。
- 面向连接 – 13周：SOCKET的建立与释放，15周：字符串和文件的收发
- 可靠 – 16周：丢包重传
- 拥塞控制 – 17周：TCP拥塞控制

SOCKET的建立与释放

- TCP协议为SOCKET设计了一些状态，在建立连接和释放连接的时候，SOCKET会在状态之间切换。本次实验需要实现三次握手、四次挥手和状态之间的迁移函数。



实验心得 & 与理论课的联系

- 实现了SOCKET状态机，完成了SOCKET的连接和断开。
- 实际应用中，SOCKET在建立连接时会创建CHILD_SOCKET，并用LISTEN_QUEUE和ACCEPT_QUEUE来保存它们。这是为了区分管理完成三次握手的SOCKET和未完成三次握手的SOCKET。这一点在理论课上没有提到，但是在实际应用中会带来极大的便利。

字符串和文件的收发

- 连接建立（双方进入ESTABLISHED状态）后，可以进行数据包的收发。本实验中引入了环形缓存RECEIVING BUFFER来完成数据的发送和缓存。接收缓存大小为RECV_WINDOW，所以在本次实验中我们实现了最基础的流量控制。
- 在进行发送文件的时候，由于文件较大，所以需要划分成小的报文进行发送，接受端再将收到的数据拼起来，组成原来的文件。

实验心得 & 与理论课的联系

- 数据包的发送看似比之前实现状态机要简单，但是实现过程中有许多琐碎的细节：将文件内容切成小段填入报文，调整好报文的ACK和SEQ，修改SOCKET的SND_UNA和RCV_NXT等，实现时需要耐心检查调试。

超时重传

- 在之前的实验中，我们进行的是无丢包网络环境下的字符串和文件的传输。本次实验中，网络中可能存在丢包的情况，因此需要实现重传机制，进一步实现数据的可靠传输。
- 维护发送队列。
- 维护两个接收队列。
- 维护超时重传计时器。

实验心得 & 与理论课的联系

- 本次实验中，仿照之前的TIMER_LIST设计了超时重传寄存器RETRANS_TIMER_LIST，来计算等待的时间，判断何时进行重传。
- 由于收到的数据包中有些是重传而得，所以会出现无序的情况，为此设计了OUT-OF-ORDER队列来保存这些数据包。理论课没有强调这一点，实际上这是一个非常巧妙的设计。

拥塞控制

- 之前我们已经有了固定大小的发送窗口，为了提高数据传输的速度，同时尽量避免网络拥塞，本次实验中我们需要改变发送窗口大小CWND。
- 维护新增的变量：TCP拥塞控制状态和拥塞窗口大小CWND。
- 完成拥塞窗口增大的两个算法：慢启动和拥塞避免。
- 完成拥塞窗口减小的两个算法：快重传和超时重传。
- 完成拥塞窗口不变的一个算法：快恢复。

实验心得 & 与理论课的联系

- 实现的算法和理论课上讲的大致相同，但是理论课上的稳定时CWND每周期增量为1，实验中增量为1/CWND，比理论课讲的更慢，不易导致拥塞。
- 经过本次实验增加的拥塞控制后，丢包重传的次数与之前相比有下降，说明拥塞控制功能能很好地提升网络的传输效率，降低重传次数。

遇到的问题

- 1. 在第二个实验的文件收发部分，在SERVER和CLIENT均进入ESTABLISHED状态后，有一定几率会卡住，不进行数据包的发送。可能是因为锁的处理有问题，但是由于难复现，暂时还没解决（不过这个问题后续实验中没有再出现，因此也没有很大影响）。
- 2. 拥塞控制部分实验生成的效果图由于扫描频率不够高所以不够美观。

建议

- 17-拥塞控制部分，研讨课PPT上给出的状态迁移图和理论课不同，感觉理论课上更直观、好理解。