

计算机网络实验报告-15

阮星程

2015K8009929047

一、 实验题目

网络传输机制实验一。

二、 实验内容

理解 TCP 协议的基本流程，完成建立和结束 TCP 连接的代码结构，实现最基本的 TCP 连接功能。

三、 实验过程

本次实验代码主要在两个文件中，tcp_in.c 和 tcp_sock.c，为了更加熟悉地掌握 tcp sock 部分的内容，我是从 tcp_sock.c 开始写起的。

在理解和思考 tcp 连接的建立流程之后，我试图开始将 tcp 连接的过程与代码框架结合起来，就拿三次握手来说，在第一次 connect 时，需要创建一个 sock，将其 bind 到一个端口，然后放入 bind_hash_table，由于四元组已经齐全，再将其放入 established_table 即可。说实话，这个操作有些不自然，不过也无可厚非。值得一提的是我不太清楚为什么在 alloc 的时候不对下面两个 list 进行初始化，经过初始化之后可以方便地通过 list_empty 判断是否放入了对应的 list 中去，这样在异常退出时可以利用这个判断来进行方便且一致的操作。只不过在 delete_entry 时需要将这个 list_head 重新初始化一下罢了。之后 wait_connect，等待唤醒。

```
init_list_head(&tsk->hash_list);  
init_list_head(&tsk->bind_hash_list);
```

Accept 时，首先创建一个子节点，放入 listen_queue，然后将它从 listen_table 中 unhash，再 hash 到 established_table 中即可，为了方便，我直接调用了 unhash 函数，由于 unhash 函数中进行了 free，为了与设计相一致，在实现时我把它注释掉了。这里的 move 到 established_table 同样不太自然，花了一些时间才能接受这个流程。

收到 syn 和 ack 包的回复就暂且略过，收到 ack 包时将 sock 放到 parent 节点的 accept queue 中，等待被接受。同样不大自然。

熟悉了基本流程之后，便一帆风顺了，只是有时对下面这几个值的理解容易混淆，犯了些小错误，也许直接换成 ack 和 seq 会方便不少，完全可以避免这些不必要的错误。

```
■ u32 snd_una;      // 对端连续确认的最大序列号  
■ u32 snd_nxt;      // 本端发送的最大序列号  
■ u32 rcv_nxt;      // 本端连续接收的最大序列号
```

接下来就可以回到 `tcp_in.c` 去了，绝大部分的代码老师已经用注释叙述得蛮清楚了，其实没太多可说，只是我觉得在 `tcp_process` 中将 `RST` 的处理放在第一位会相对合适些，以及在查找的时候，如果没有找到需要进行相应的处理，这点在原有框架中有缺漏，我将它不上了。

四、实验结果

可见上图，tcp 连接正确地建立以及结束，并在 timeout 后被成功移除，结果符合预期。

本次实验中涉及的 tcp 的协议部分其实很简单，麻烦的是将协议与实际代码框架结合起来，说实话，在第一次看到 tcp_sock 里面的五个 list_head 的时候，简直觉得眼花缭乱无从下手，尤其是 hash_list 那个部分，略微有些 trick 的设计，使代码显得有些 unnatural，没有看到详细的叙述，就觉得有些晕，但当逐渐熟悉起来过后，由于各个步骤老师都写得比较详细了，所以没花太多心思，只是单纯的翻译下代码便是。书写完成后，debug 也仅仅是梳理了下 snd_una, sun_nxt, rcv_nxt 与 ack 和 seq 之间的关系，生成子节点时将 parent 项添加上，便完成了实验。还是蛮轻松的。