Socket Programming Report

2016.10.24

Contents

1 作者信息

王永赫（软件 41 班，2014013406，jlmhkwang@gmail.com）

2 开发环境

- 操作系统：macOS Sierra 10.12
- make：GNU Make 3.81 built for i386-apple-darwin11.3.0
- CC：Apple LLVM version 8.0.0 (clang-800.0.38)
- PyQt：Python 3.5.2、Qt 5.7.0、PyQt 5.7

3 运行方式

3.1 server 端

请 cd 到 server 的 src 目录，在 bash 中运行：

```
make run
```

按提示输入密码即可完成，需要额外的配置可以更改 Makefile。

3.2 client 端

在 program 子目录中有可以在 macOS 系统中运行的程序，如果运行出错，请用 homebrew 安装 PyQt5：

```
brew install python3 pyqt5
```

后 cd 到对应目录运行

```
python3 client.py
```

4 架构设计

4.1 多线程

为了方便地解决响应多个客户端的需求，使用 pthread 模块在主线程 accept 每个客户端的登录请求后创建一个子线程完成该客户端的整个生命周期(监听-处理-响应)。

设计 thread_data 记录每个线程需要的信息，包括：

变量	说明
pthread_t tid	系统需要的线程标识符
int connfd	消息连接的文件描述符
unsigned long status	当前状态
char *user	用户名
char *type	当前通过 TYPE 设置的类型
struct sockaddr_in *remote	PORT 命令连接客户端信息
int listen	PORT 命令建立的 socket
char *temp	临时存储的字符串（用于重命名）
char *prefix	当前会话目录前缀

在每个线程（需要的）函数间传递上述结构体的指针作为会话信息。遇到核心错误或者客户端退出则关闭该线程，主线程退出前先 join 所有线程。

4.2 Check-Handle 模式

为了精简代码和便于增加新功能，将服务器行为概括为 check->handle->response，则要求所有 Handler 具备 check 和 handle 两个接口（response 可以为 handle 的返回值），于是线程生命函数 dispatcher 的消息循环只有几句话，即初始化，读消息，找个愿意接锅的函数处理一下，把返回值写过去。

如果 handle 函数有网络操作，例如 RETR 和 STOR，文档要求此时服务器暂时不处理任何其他请求，由于 handle 没有返回，则该线程具备这样的特性，但是没有实现 ABOR 的中断（如果有，仿佛需要再创建子线程?）。

例如向服务器添加 LIST 功能，需要做的是，在 handlers.c 文件中添加两个函数`int list_check(const char *st)`和

`int list_handle(thread_data *thd, char *st)`，然后在同一文件的 all_check_in_turn 和 all_handle_in_turn 中添加两个函数名，保证其位置对应，写完相应功能，即结束，很便于拓展。

遗憾的是一些进一步封装的操作（例如 handlers.c 中仍有重复代码）没有完成，代码还有重构的空间。

5 完成情况

5.1 server 端

实现了 19 个消息处理：USER, PASS, SYST, TYPE, QUIT, PORT, PASV, RETR, STOR, CWD, PWD, CDUP, DELE, MKD, RMD, RNFR, RNTO, LIST, ABOR；即要求的基本功能和目录功能。

接受 port 和 root 参数，并且建议 root 参数设置为/home/tmp 或其他，设置为/tmp 在我的电脑上引起了由于 macOS Sierra 10.12 系统的 SIP 功能导致的权限错误。

另外对于文档中提到的大文件传输如何不阻塞服务器的问题，可以限制传输占用的带宽给其他客户让路，可以 fork 出子进程专门处理该请求（没有实现），本质上服务器还是要服务好文件传输的，计算资源有必要使用，但是对于客户 connect 的响应不能被大文件传输阻塞。

5.2 client 端

使用 python3 和 PyQt5 编写图形界面程序便于测试，左侧为功能按钮可以直接测试服务器功能，右侧分别是命令行模拟器用于发送指令（例如没有作为按钮出现的 CWD 等）和文件树。

由于时间不够文件树每次只能列出一层目录的文件。

6 遇到的困难

多数是不细心导致的隐藏问题难以调试，例如

- 在运行 autograde.py 时长时间 0 分，后来查出回复消息的时候字符串的'\r\n' 之后多传出了一个'\0'；
- 在运行 LIST 功能的测试时发现第二次运行无法拿到结果，刚开始以为是 PyQt 的问题，后来发现是 popen 之后没有调用 pclose 把管道关闭掉；
- 在测试 PASV 模式的下载时发现文件明明传输结束却卡死在传输上，后来发现是缺少 socket 的 close 环节导致；
- 在编写 check_path 函数时发现如果第一次调用返回错误，则永远返回错误，调试后发现逻辑有问题（因为 C 语言的 errno 是出错了设置一下，平时不动的）。

7 遗憾

由于错误的时间规划，大作业未能如期完成，在此做严重检讨，但还是希望写到能基本通过测试的程度再提交，由此对助教团队造成的不便深感抱歉……