FTP 实验报告

2013013340 陈子剑

一. 总体实现：

服务器建立 socket 并监听特定的端口，然后进入死循环等待客户端的链接。在循环中采用不阻塞的 Accept 方式与客户端连接，连接后将这个新的链接加入到一个集合中，这个集合维护了每个客户端的各种状态信息，然后遍历这个集合判断是否有客户端发送命令。如果有，则新建一个线程处理该命令。
客户端建立 socket 并连接特定的 ip 和端口， 然后进入死循环。每次循环先以阻塞方式监听键盘事件，解析命令，如果需要则执行相应的命令，否则直接发送给服务器命令然后等待服务器回传消息。

二. 数据结构：

```
typedef struct {
    unsigned int count;
    int fdArray[FD_SETSIZE];
    int fdState[FD_SETSIZE];
    char username[FD_SETSIZE][10];
    struct sockaddr_in addrs[FD_SETSIZE];
    int pasvFdArray[FD_SETSIZE];
    int rename;
    char renamefile[FD_SETSIZE][257];
} socketSet;
```

这个数据结构是用来维护与服务器建立连接客户端的信息的，其中 fdArray 存储客户端的套接字描述符，fdState 存储客户端的状态，username 是用户名，addrs 存放用户发送 port 命令时的 ip 地址和端口，pasvFdArray 存放客户端发送 pasv 命令建立的监听数据传输的套接字描述符，rename 为 1 时表示有文件正在被重命名可以使用 rnto 指令，renamefile 为正在被重命名的文件。
客户端的状态有 5 种：

```
#define GUEST 0 //未登录也未输入用户名的状态
#define PASS 1  //输入用户名未输入密码的状态
#define LOGIN 2 // 已经登录的状态
#define PORT 3 // 已经登录并且已经被服务器接受PORT请求的状态
#define PASV 4 // 已经登录并且已经被服务器接受 PASV 请求的状态
```

三．命令实现：

除了 QUIT 和 USER 命令，其他命令均不许在未登录的状态下使用，若使用会返回 503 提示用户先登录。

1. USER：
   服务器接收到这个命令则设置对应客户端的用户名并且改变它的状态为输入用户名的状态，并返回 331。如果用户已经登录则返回 501 提示用户断开连接后重新连接登录。
2. PASS:
   先检测状态是否为已经输入用户名的状态，否则返回 503 提示用户先输入用户名，是则验证。如果验证成功返回 230 并改变状态为已登录，否则返回 530。在这个服务器里只允许匿名用户，匿名用户的密码任意。
3. QUIT：
   服务器向客户端发送 221 并断开连接，从集合中删除相应的连接。
4. SYST:
   服务器向客户端发送 215 系统信息。
5. TYPE：
   服务器判断参数是否为 I 或 i，是则返回 200，否则返回 504 参数错误。（后期为了支持 mac os 上的 transmit app 连接此服务器又增加了参数 A/a 的支持。）
6. PORT:
   服务器解析 ip 和端口，若解析成功则修改状态为 PORT，为后来收到 RETR 和 STOR 命令做准备，并返回客户端 200。否则如果解析错误返回 501。
   客户端接受服务器返回的 code，如果为 200 则新建 socket 并监听先前声明的端口并设置状态为 PORT。
7. PASV:
   服务器修改状态为 PASV，然后新建一个 socket 并监听随机 20000-65535 的端口，并且将自己的 ip 和这个随机端口返回给客户端。成功则返回给客户端 227，否则返回 504.
   客户端接受服务器返回的条件码，如果为 227 则存储返回的 ip 和端口并改变状态为 PASV，以供后面传送文件时需要。
8. RETR：
   服务器判断状态是为 PORT 或 PASV，是 PORT 则向客户端发起连接，是 PASV 则接受客户端的连接请求。然后获取相应的文件，并发送给客户端。如果获取文件不存在或没有权限则返回相应的错误条件码。
   客户端判断状态时 PORT 或 PASV，是 PORT 则接受服务器的请求，是 PASV 则主动向服务器发送连接请求。
9. STOR：
   STOR 和 RETR 是对称的关系。
10. CWD:
    linux 的 cd 命令，支持绝对路径和相对路径，在根目录运行 cwd ..不会再向上一级（如：把根目录设置为/tmp，在这个目录下 cwd ..不会移动，而且客户端看到的假象是/tmp 就是/）。

11. CDUP：同 CWD ..
12. DELE: 删除文件，支持绝对路径和相对路径下的文件。
13. LIST：同 linux 命令 ls –l，不接受参数。
14. MKD: 创建一个文件夹，支持绝对路径和相对路径。
15. RMD：删除一个文件夹，支持绝对路径和相对路径。
16. RNFR 和 RNTO：重命名一个文件，RNFR 选择要重命名的文件，RNTO 修改文件名。支持绝对路径和相对路径，如果 RNTO 带路径则文件会被移动带相应的路径并修改文件名。

四． 困难：

1. 用 C 写代码缺少了很多库（其实其他库我不在乎关键是字符串的库也很弱，如果是 C++的话有 string 也足够了）导致了字符串处理的复杂，全部需要手动编写。
2. 传大文件的阻塞问题：
   一开始只是防止了服务器被 Accept 函数阻塞掉，执行其他用户命令时还是会被阻塞，但是因为命令都是简单地，所以基本不会有影响，但是传大文件的时候服务器就会因为在执行传送命令被阻塞。考虑很久只好对每个客户端的命令新开一个线程执行，这样在传送大文件的时候就不会阻塞了。

五． 特色：

1. 大文件传输不会阻塞 server：
   前面提到，客户端的每条命令都会对应服务器的一个线程，采用多线程的方式不会阻塞 server；
2. 支持多种 server 的回传 code：
   写 ftp 时不断 telnet ftp.ssast.org 21，模仿了它的回传命令，使得错误提示也很丰富。
3. 命令兼容小写字母：
   解析命令时无视大小写，测试时候也省去了按 Caps lock。
4. server 可以用 Mac OS 上的 APP：Transmit 连接登录，支持多种命令。
5. client 可以进行域名解析，用户输入的不是 ip 是 hostname 时，client 会获得 hostname 的所有 ip 遍历，使用第一个可以建立连接的 ip，每次 connect 的 timeout 为 5 秒。