



## 计算机网络

### FTP 实验报告

姓名：黄 翔

班级：软件 71

学号：2017013570

#### < 服务端 >

##### 【基本信息】

开发环境：Linux (Ubuntu 18.04)

编程语言：C

文件传输：二进制模式

##### 【实现命令】

命令	功能	返回
USER	发送用户名	331/501/504
PASS	发送密码	230/332/501
RETR	下载文件	150/226/425/426/450/451/500/501
STOR	上传文件	150/226/425/426/451/500/501
QUIT/ABOR	退出系统	221/501
SYST	获取系统类型	215/501
TYPE	设置传输类型	200/501/504
PORT	设置主动模式	200/501
PASV	设置被动模式	227/501
MKD	创建目录	257/450/501
CWD	切换目录	250/450/501
PWD	打印目录路径	257/501
LIST	获取目录文件列表	150/226/425/426/450/500/501
RMD	删除目录	250/450/501
RNFR	重命名目录	350/501
RNTD	重命名目录	250/450/501
REST	设置文件指针偏移量	350/501
其他未实现指令		500

##### 【多用户连接】



服务端采用了多线程结构，支持多用户连接。当新用户连接成功后，服务端将派生出一个新的线程处理用户请求。不同用户之间独立进行、互不干扰。关键代码如下：

```
// 持续创建连接 多线程
while (1) {
    // 进行连接
    if ((connfd = acceptConnection(listenfd)) < 0)
        continue;
    // 进行处理 派生线程
    pthread_t id;
    pthread_create(&id, NULL, handleConnection, (void*)&connfd);
    pthread_detach(id);
}
```

### 【断点续传】

服务端的断点续传功能主要针对下载文件（RETR）。服务端实现了 **REST 指令**，客户端在需要恢复下载时，可先检查已传输的字节，并用 REST 指令设定文件指针偏移，下次调用 RETR 指令下载时，服务端将自动设置传输文件的指针偏移，从之前文件的断点处继续传输。

### 【错误处理】

服务端进行了较为完善的错误捕获与处理，同时利用 <errno> 进行了错误输出。

部分已进行处理的错误如下：

```
#define S_SUCCESS      0      // 执行成功
#define S_ERROR_SOCKET -1     // 套接字创建失败 errno
#define S_ERROR_BIND   -2     // 绑定失败 errno
#define S_ERROR_LISTEN -3     // 监听失败 errno
#define S_ERROR_CONNECT -4    // 连接失败 errno
#define S_ERROR_ACCEPT -5     // 接受失败 errno
#define S_ERROR_READ   -6     // 读取失败 errno
#define S_ERROR_WRITE  -7     // 写入失败 errno
#define S_ERROR_FILE   -8     // 文件操作失败
#define S_CONNECT_BREAK -9    // 连接断开

#define F_SUCCESS      0      // 执行成功
#define F_ERROR        -1     // 执行失败 错误信息存于errno
#define F_OVERFLOW     -2     // 路径过长
#define F_NOTFOUND     -3     // 无效路径
#define F_NOENAME      -4     // 未处于重命名状态
#define F_ISROOT       -5     // 已处于根目录

/*
printf("Error socket(): %s(%d)\n", strerror(errno), errno);
printf("Error bind(): %s(%d)\n", strerror(errno), errno);
printf("Error listen(): %s(%d)\n", strerror(errno), errno);
printf("Error connect(): %s(%d)\n", strerror(errno), errno);
printf("Error accept(): %s(%d)\n", strerror(errno), errno);
*/
```

## < 客户端 >

### 【基本信息】

开发环境：Linux (Ubuntu 18.04)

编程语言：C/C++

IDE / GUI 框架：Qt



## 文件传输：二进制模式

### 【实现命令】

USER / PASS / RETR / STOR / QUIT / SYST / TYPE / PORT / PASV / MKD / CWD / PWD / LIST / RMD / RNFR / RNT0 / REST

### 【图形界面】



**【连接/登录/模式选择】**可进行连接、登录服务端，选择传输模式。在成功登录后，客户端将自动调用 SYST 指令与 TYPE 指令，获取系统信息并设置传输模式为二进制模式。

**【进度条】**在进行上传/下载任务时，进度条将可实时显示上传/下载进度。

**【信息框】**信息框将显示客户端的运行信息，在必要时会打印服务端的返回信息。成功信息用绿色标识，失败信息用红色标识，提示信息用黄色标识，服务端返回信息用蓝色标识。

**【文件列表】**文件列表将同步显示当前路径与当前目录文件信息。列表更新时，会自动调用 LIST 命令，并解析返回内容。对于标准服务端 vsftpd 及上述自主实验的 server，将能正确解析返回内容。对于其他服务端，由于返回格式的差，可能有部分内容无法解析。

### 【文件操作】

**切换** 双击文件列表中的路径即可进行路径切换。

**创建** 在创建文本框中输入目录名称，点击“创建”即可创建新目录；

**删除** 选中文件列表中的文件/目录（将会自动填入文本框），点击“删除”即可删除指定文件/目录；

**上传** 点击上传，在弹出的文件选择框中选中文件，即可自动进行文件上传。

**下载** 选中文件列表中的文件（将会自动填入文本框），点击“下载”，在弹出的文件保存框中选择好路径与文件名后即可自动进行文件下载。

**重命名** 选中文件列表中的文件/目录（将会自动填入左侧文本框），在右侧文本框中输入新名称，点击“重命名”即可进行路径重命名。

**断点续传** 选中文件列表中的文件（将会自动填入左侧文本框），点击“断点续传”，选择本地待续传的文件，即可进行断点续传。

**注意** 由于编码的问题，对于带中文的路径，处理时可能会遇到错误。请尽量在英文路径下测试。