FTP实验报告

1. 实验环境

- FTP Server: Ubuntu 22.04 LTS(VMWare) + VSCode + GNU C
- FTP Client: Ubuntu 22.04 LTS(VMWare) + VSCode + Python 3.10.12

2. 功能实现

2.1 FTP Server

2.1.1 项目结构和模块划分

该项目位于`/server/src`目录下，`common.h`用于引入必要头文件，声明函数，结构体和全局变量，`main.c`用于解析命令行参数，`server.c`用于启动客户端，`handles.c`用于处理客户端连接和各类命令，`utils.c`定义了与主要逻辑无关的一些工具函数，`server`为可执行文件。

2.1.2 命令实现与错误处理

本FTP服务端实现的命令功能说明及错误处理如下表所示：

FTP命令	功能说明	错误处理
USER <username>	接受用户名，这里只支持“anonymous”一个用户，其他任何用户均将被拒绝，用户名输入正确返回状态码331	当输入用户不为“anonymous”时返回状态码530
PASS <passwode>	接受密码，这里当用户输入任意密码后均可成功登录，返回状态码230	当没有参数时返回状态码501
PORT <h1, h2, h3, h4, p1, p2>	进入PORT传输模式，接受并记录用户传来的IP地址和端口号，等待后续数据传输时连接客户端的相应端口，成功后返回状态码200	当没有参数或参数格式错误时返回状态码501
PASV	进入PASV传输模式，创建socket，将服务端IP地址和socket所绑定的端口号以一定格式传递给客户端，等待客户端进行连接，成功后返回状态码227	

FTP命令	功能说明	错误处理
SIZE <filename>	根据用户提供的文件名，返回该文件的字节数，正确状态码213	当没有参数时返回状态码501，当文件无法访问时返回状态码550
REST <n>	将文件传输设置为从n字节开始，用于断点续传，正确状态码350	
RETR <filename>	根据客户端之前选择的数据传输模式，将服务端的文件传输到客户端，在PORT模式下，会先连接PORT命令给出的IP地址和端口号，在PASV模式下，会让之前创建的socket先接受客户端的连接，然后在该连接上进行数据传输，首先返回状态码150确认连接成功，准备开始进行数据传输，当数据传输结束后返回状态码226	若没有参数返回状态码501，若无法正确获取指定文件返回状态码550，若数据传输连接错误返回状态码425或426
STOR <filename>	与RETR命令基本相同，客户端将本地文件上传到服务端	同RETR
SYST	固定返回“215 UNIX Type: L8”	
TYPE <type>	只支持将数据类型设为byte，即“TYPE I”，成功返回状态码200	参数错误返回状态码501
QUIT	关闭连接，返回状态码221	
ABOR	同QUIT	
MKD <path>	创建指定文件夹，成功返回状态码250	若没有参数返回状态码501，若目录已存在或目录创建失败，返回状态码550
CWD <path>	进入指定文件夹，成功返回状态码250	若没有参数返回状态码501，若进入失败或尝试进入根目录之外的目录，返回状态码550
PWD	获取当前文件夹路径，成功返回状态码257	
LIST [path]	过程基本同RETR和STOR，使用“ls -l”命令并使用管道读取来获得当前目录信息	若管道打开失败，返回状态码550，其他同RETR和STOR

FTP命令	功能说明	错误处理
RMD <path>	删除指定文件夹，成功返回状态码250	若没有参数返回状态码501，若删除失败返回状态码550
RNFR <filename>	指定一个文件，后面讲使用RNTO命令来修改这个文件的名字，成功返回状态码330	若没有参数返回状态码501，若无法找到指定文件返回状态码550
RNTO <filename>	重命名之前用RNFR指定的文件，成功返回状态码250	若没有参数返回状态码501，若重命名失败返回状态码550

2.1.3 可选功能实现

使用REST命令可以设置需要重传的位置，此后使用RETR或STOR命令时就会从该位置开始传输，实现了断点续传。

在RETR和STOR命令中使用fork()命令创建子进程来进行文件传输，不会阻塞主进程进行命令处理。

2.2 FTP Client

在客户端实现了与服务端完全相同的命令，与服务端相反的是，使用PASV命令后，客户端记录服务端发来的IP地址和端口，在数据传输前进行连接，使用PORT命令后，创建一个新的端口，在数据传输前接收服务端的连接。

除此之外，客户端还实现了一个size命令，该命令不会传入服务端，只是用来获取指定本地文件的大小，方便用户在使用REST命令时正确地设置文件传输的开始位置。

因此，客户端也同样实现了断点续传的功能。

3. 实验感想

本次实验从0开始实现FTP文件传输协议对我来说是一个巨大的挑战，由于对socket网络编程毫无了解一开始无从下手，但通过查阅资料了解了主要流程后也算是成功完成了任务。这次实验让我对socket网络编程有了一定的认识，可以从更底层的角度了解网络传输是怎样实现的，同时我也了解了Linux的不少系统调用，这对我以后进行基于Linux系统的编程也十分有帮助。