

# 1. 引言

## 1.1 编写目的

本详细设计文档是为本小组后期的编码提供基础和依据，同时也为规范我们小组的开发。

## 1.2 定义

- Socket：套接字
- TFTP：简单文件传输协议
- IP：网络地址
- Port：端口号

## 1.4 参考资料

书名	作者	出版社	日期
《嵌入式 Linux》	孙琼	人民邮电出版社	2006 年 7 月
《C 语言程序设计》	谭浩强	清华大学出版社	2004 年 6 月

## 传输协议设计

**程序描述：**通信双方都需要遵循一定的数据结构，而协议就是定义这样的标准

**功能：**通过定义通信双方都需要遵循的数据结构，使对方能正常识别发送过过去的  
数据结构体。

**性能：**部分程序相对独立性较高，外部程序只需要在必要的时候调用该程序的接口就可  
以了，因此该程序容易集成，灵活性高；

**协议结构体：**

```
struct FilePackage
{
    char cmd;                //操作命令
    int  filesize;           //每次传输数据包大小
    int  ack;                //标志位
    char username[50];       //客服端用户名
    char filename[125];      //传输文件名
    char buf[1024];          //传输文件的元数据
};
```

**打包函数：**

```
struct FilePackage pack(char tCmd, char* tBuf, char* tFilename,
int tFilesize, int tAck,int count,char *uname)
{
    struct FilePackage tPackage;
    tPackage.cmd = tCmd;
```

```

memcpy(tPackage.buf,tBuf,count);
strcpy(tPackage.filename, tFilename);
strcpy(tPackage.username, uname);
tPackage.filesize = tFilesize;
tPackage.ack = tAck;
return tPackage;
}

```

**流程逻辑：**当接收到协议包的时候，调用解析函数解析出有效数据，当要发送数据的时候调用封包函数将要发送的数据封装到协议包。

**接口：**提供的接口：

```

struct FilePackage pack(char tCmd, char* tBuf, char* tFilename, int tFilesize, int
tAck,int count,char *uname) //打包函数

```

```

struct FilePackage unpack(SSL *,struct FilePackage ); //解包函数

```

**数据包cmd&ack具体设计：**

登陆：

**Cmd:L**

服务器发送数据包的 **ACK: 0** 用户名或密码错误

- 1 登陆成功
- 2 客户端最大连接数

客服端发送数据包的 **ACK: 9** 登陆服务器

下载：

**Cmd: D**

服务器发送数据包的 **ACK: 0** 接受下载，返回待下载文件大小

- 2 开始下载
- 4 下载完毕

客服端发送数据包的 **ACK: 9** 请求下载

- 3 接受完毕

上传：

**Cmd: U**

服务器发送数据包的 **ACK: 0** 接受上传请求

- 1 本地磁盘空间不足
- 3 接收完毕

客服端发送数据包的 **ACK: 9** 请求上传，上传文件大小,文件名

- 2 开始上传文件
- 4 上传完毕

显示文件列表：

**Cmd: S**

服务器发送数据包的 **ACK: 1** 第一个显示文件列表

客服端发送数据包的 **ACK: 9** 请求显示

## 服务器详细设计

**功能描述:** 实现连接用户的验证, 实现多用户连接, 日志管理, 配置文件, 管理员和用户账号申请, 完成和用户端命令的交互。

**流程介绍:** 首先管理员登陆服务器, 服务器验证成功后服务器进行相关初始化工作, 然后服务器产生 2 个线程, 一个菜单线程, 显示菜单操作列表, 另一个线程建立 **socket** 套接字, 监听网络, 等待客户端连接, 当有一个客户端发来连接请求时, 线程管理会自动为其分配一个线程处理此客户端的连接请求, 然后通过用户管理对其输入的用户名和密码进行验证, 验证通过后, 则接收客户端命令请求, 进行解析, 并处理相应的操作, 针对用户的每一种操作, 将其行为存放到日志文件中, 其中客户端与服务器之间通信时采用 **OpenSSL** 进行加密传输, 在服务器运行时可以设置客户端最大的连接, 增加管理员, 增加用户等配置信息, 申请新用户后, 服务器自动生成用户名的目录, 用户在自己的目录下进行相关操作。

### 相关接口:

#### 1) 服务器初始化

```
void InitAU();          /*将文件中管理员/用户的帐号信息读入数组*/
int InitMaxClientNum(); /*初始化客户端最大连接数*/
```

#### 2) 线程创建

```
pthread_create(&controlId, NULL, (void *)mainMenu, NULL) //菜单线程
pthread_create(&mainId, NULL, (void *)mainThread, NULL) //socket 线程
```

#### 3) 处理客户端线程

```
pthread_create(&id, NULL, (void *)process, ssl);
```

#### 4) 服务器接受/发送数据包

```
SSL_read(NewFd, &buff, sizeof(struct FilePackage));
SSL_write(NewFd, &sendPackage, sizeof(struct FilePackage));
```

#### 5) 服务器打包/解包函数

```
struct FilePackage unpack(SSL *NewFd, struct FilePackage tpack);
struct FilePackage pack(char tCmd, char* tBuf, char*
tFilename, int tFilesize, int tAck, int count, char *uname);
```

#### 6) 服务器加密传输相关函数

```
SSL_library_init(); /* SSL 库初始化 */
OpenSSL_add_all_algorithms(); /* 载入所有 SSL 算法 */
SSL_load_error_strings(); /* 载入所有 SSL 错误消息 */
SSL_CTX_use_certificate_file(ctx, temp=strcmp(pwd, "/cacert.pem"), SSL_FILETYPE_PEM) /* 载入用户的数字证书 */
SSL_CTX_use_PrivateKey_file(ctx,
temp=strcmp(pwd, "/privkey.pem"), SSL_FILETYPE_PEM) /* 载入用户私钥 */
```

## 客户端详细设计

**功能描述:** 实现客户端与服务器的文件下载和上传并且 openssl 传输加密, 客服端实现多线程, 允许一个客服端同时从服务器下载或上传多个文件。

**流程介绍:** 首先使用服务器新建一个账号, 然后即可以利用客户端进行登陆, 进行本地目录操作与上传下载。

**相关接口:**

### 1) 连接函数

```
int connectto(int argc,char *args[]);           //与服务器建立连接
```

### 2) 用户登陆函数

```
int login(char username[],char userpasswd[]);
```

### 3) 用户主菜单函数

```
void mainMenu();//主菜单
```

### 4) 显示客户端, 服务器目录与切换客户端目录操作

```
void Show(char temp[100]);
```

### 5) 上传函数

```
int UpdateF();
```

### 6) 下载函数

```
int DownloadF();
```

### 7) 上传函数

```
int UpdateF();
```

### 8) SSL 相关初始化函数

```
/* SSL 库初始化 */
```

```
SSL_library_init();
```

```
/* 载入所有 SSL 算法 */
```

```
OpenSSL_add_all_algorithms();
```

```
/* 载入所有 SSL 错误消息 */
```

```
SSL_load_error_strings();
```

```
/* 以 SSL V2 和 V3 标准兼容方式产生一个 SSL_CTX , 即  
SSL Content Text */
```

```
ctx = SSL_CTX_new(SSLv23_server_method());
```

```
/* 也可以用 SSLv2_server_method() 或
```

```
SSLv3_server_method() 单独表示 V2 或 V3 标准 */
```

## 上传/下载详细设计

### 上传

**功能介绍:** 客服端上传文件至服务器端

**流程介绍:** 首先客服端向服务器发送cmd为U,ack为9的数据包请求上传, 服务器接受到客服端的命令后, 进行磁盘检查和上传文件是否已存在, 若磁盘空间不足, 则向客服端返回cmd为U, ack为1的数据包, 如果上传文件已经存在则覆盖原文件, 然后客

服务端开始上传文件，客户端上传完毕后向服务器发送cmd为U，ack为3的数据包，服务器接受到该数据包后停止接受上传，向服务器返回一个cmd为U，ack为4的数据包，表示接受完毕。

#### 相关接口：

```
int UpdateF(); //客户端上传文件
struct FilePackage unpack(SSL *NewFd, struct FilePackage tpack); //服务器解析 cmd 为 U 的数据包
```

#### 下载

**功能介绍：** 客户端从服务器上下载文件

**流程介绍：** 首先客户端向服务器发送cmd为D，ack为9的数据包请求下载，服务器接受到命令后返回D，ack为0的数据包，服务器发送cmd为D，ack为2的数据包开始下载文件，下载完毕后服务器向客户端发送cmd为D，ack为4的数据包，客户端接受此数据包后返回cmd为D，ack为3的数据包表示接受完毕，如果下载到重名的文件，客户端退出此次下载

#### 相关接口：

```
int DownloadF() //客户端上传文件
struct FilePackage unpack(SSL *NewFd, struct FilePackage tpack); //服务器解析 cmd 为 D 的数据包
```

## OPENSSL加密步骤

服务器	客户端
SSL 库初始化	SSL 库初始化
载入所有 SSL 算法	载入所有 SSL 算法
载入所有 SSL 错误消息	载入所有 SSL 错误消息
创建本次会话连接所使用的协议	创建本次会话连接所使用的协议
申请 SSL 会话的环境 CTX	申请 SSL 会话的环境 CTX
载入用户的数字证书	创建一个 socket 用于 tcp 通信
载入用户私钥	初始化服务器端的地址和端口信息
检查用户私钥是否正确	
开启一个 socket 监听	
等待客户端连上来	连接服务器
基于 ctx 产生一个新的 SSL	基于 ctx 产生一个新的 SSL
将连接用户的 socket 加入到 SSL	
建立 SSL 连接	建立 SSL 连接
接收数据并保存到newfile	发送数据
关闭 SSL 连接	关闭 SSL 连接
释放 SSL	释放 SSL
关闭 socket	关闭 socket
关闭监听的 socket	
释放 CTX	释放 CTX

## 测设计划

### 服务器:

序号	测试对象	测试计划
1	登录验证	测试管理员是否能够正常登录
2	添加账号	测试能否成功添加管理员和用户账号。
3	日志管理	测试能否记录操作日志
4	连接验证	测试能否与客户端进行连接
5	多线程	测试能否同时处理多客户请求
6	配置管理	测试能否改变客户最大连接数。
7	浏览	测试能否收到客户请求的包及发给客户文件列表的包
8	上传	测试能否收到从客户端发过来的包
9	下载	测试能否收到客户的请求包及发送客户文件内容的包
10	多线程上传	测试能否同时收到从客户端发来的不同请求的包
11	多线程下载	测试能否同时发送不同文件的包到客户端

### 客户端:

序号	测试对象	测试计划
1	登录验证	测试管理员是否能够正常登录
2	浏览	测试能否浏览本地目录以及接受服务器发送的文件列表数据包并显示
3	上传	测试能否发送数据包至服务器
4	下载	测试能否收到从服务器发过来的包
5	多线程上传	测试能否同时上传多个文件
6	多线程下载	测试能否同时下载多个文件

## 客户端的登录界面

```
root@localhost:/home/liyu23/client
[root@localhost client]# ./client 192.168.0.7
ID: user
PASSWD: 111
```

客户端主窗口程序-显示服务器文件列表与客户端文件列表

```
root@localhost:/home/liyu23/client

Client Files List
client  client.c.bak  client.h.bak  test  xx.jpg
client.c  client.h      server1.c    test.c  xxx
Server Files List
.      ..      admin.txt   client.h  log.txt  maxclientnum.txt
erver  test   putty.exe   server.h  server.h
erver1.c.bak  user.txt   xx.jpg

Client console
1.Update Files  2.Download Files  3.Exit
Use ls or cd to display and change dir
Please input the Client command:
```

客户端当前目录

服务器当前用户目录

操作命令菜单  
也可以直接输入ls,cd  
等此类命令进行客户  
端目录切换和显示,  
如rm想删除个文件当  
然也是可以的

```
root@localhost:/home/liyu23/client

Client Files List
bin dev home lib misc opt root tftpbboot usr
boot etc initrd lost+found mnt proc sbin tmp var

Server Files List
. .. admin.txt client.h log.txt maxclientnum.txt s
erver test putty.exe server.h server.h.bak server1.c s
erver1.c.bak user.txt xx.jpg

Client console
1.Update Files 2.Download Files 3.Exit
Use ls or cd to display and change dir
Please input the Client command:cd /boot 从/根目录切换到boot目录
```

```
root@localhost:/home/liyu23/client

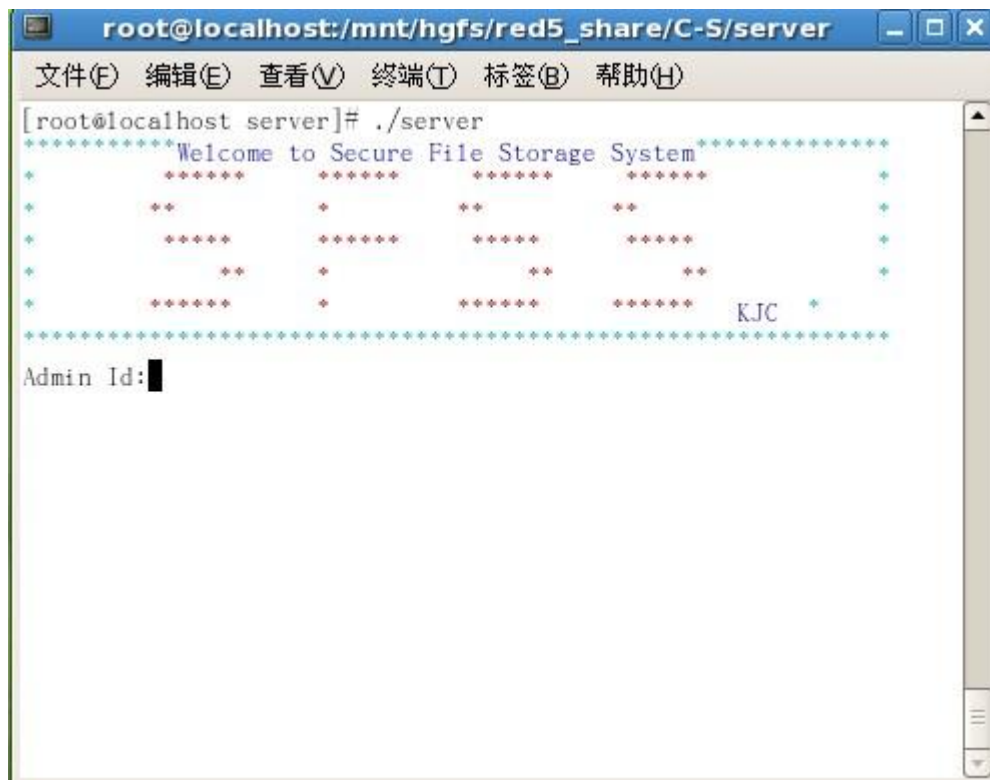
Client Files List
boot.b kernel.h module-info-2.4.20-8 vmlinuz
chain.b lost+found os2_d.b vmlinuz-2.4.20-8
config-2.4.20-8 message System.map
grub message.ja System.map-2.4.20-8
initrd-2.4.20-8.img module-info vmlinux-2.4.20-8

Server Files List
. .. admin.txt client.h log.txt maxclientnum.txt s
erver test putty.exe server.h server.h.bak server1.c s
erver1.c.bak user.txt xx.jpg

Client console
1.Update Files 2.Download Files 3.Exit
Use ls or cd to display and change dir
Please input the Client command:
```

服务器的登录界面



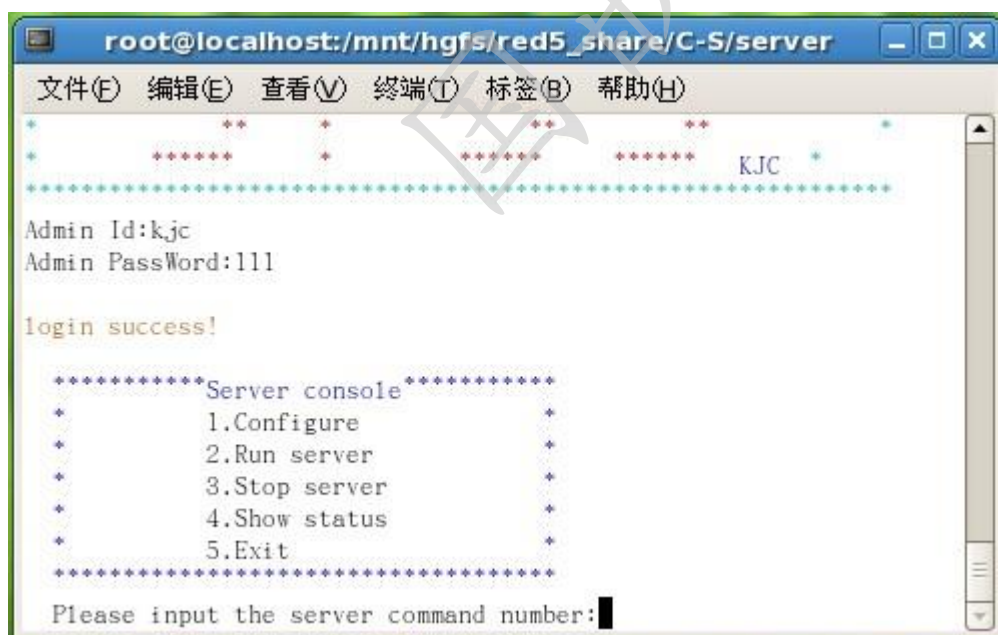


```
root@localhost:/mnt/hgfs/red5_share/C-S/server
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)

[root@localhost server]# ./server
*****Welcome to Secure File Storage System*****
*               *               *               *               *
*      **      *               **      **      *
*      ***** *****      *****      *****
*      **      *               **      **      *
*      ***** *               ***** ***** KJC
*****

Admin Id:█
```

成功登陆



```
root@localhost:/mnt/hgfs/red5_share/C-S/server
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)

*               *               *               *               *
*      **      *               **      **      *
*      ***** *               ***** ***** KJC
*****

Admin Id:kjc
Admin PassWord:111

login success!

*****Server console*****
*               *               *               *               *
*      1.Configure      *
*      2.Run server      *
*      3.Stop server      *
*      4.Show status      *
*      5.Exit      *
*****

Please input the server command number:█
```

### Configure配置服务器

- 1, 设置服务器最大连接数
- 2, 添加服务器管理账号
- 3, 添加客户端账号

```
root@localhost:/home/liyu23/server
*****Configure*****
*      1.Set maximum client      *
*      2.Add admin account       *
*      3.Add client account      *
*      4.Go back                 *
*****
Please input the configuration command number: █
```

