**网络C/S文件传输工具说明文档**

目录

[**1简介 1**](#_Toc17141721)

[**2设计架构 1**](#_Toc17141722)

[2.1文件服务器探测 2](#_Toc17141723)

[2.2 TCP文件传输 2](#_Toc17141724)

[2.2.1 TCP文件传输流程 2](#_Toc17141725)

[2.2.2 TCP服务实现 3](#_Toc17141726)

[2.2.3 TCP并发服务 4](#_Toc17141727)

[2.3 UDP文件传输 4](#_Toc17141728)

[2.3.1 UDP文件传输流程 4](#_Toc17141729)

[2.3.2 UDP服务实现 4](#_Toc17141730)

[2.4文件协议 5](#_Toc17141731)

[2.5文件校验 5](#_Toc17141732)

[**3 演示文档 6**](#_Toc17141733)

[**4 测试文档 8**](#_Toc17141734)

[4.1测试方案 8](#_Toc17141735)

[4.2测试用例 8](#_Toc17141736)

[4.3测试结果报告 8](#_Toc17141737)

# 1简介

来源：校招新员工（嵌入式）编程训练

编程训练题目：网络C/S文件传输工具

功能要求：

1、文件服务器探测：实现服务器探测功能，客户端可以探测当前局域网中的可用服务器；探测结果在用户终端输出。

2、文件传输：服务器端支持TCP、UDP两种协议类型，客户端支持通过参数选择文件传输类型，包括TCP/UDP支持文件上传下载功能，同名文件覆盖。

3、文件校验：支持文件校验机制，确保我呢见内容正确完整持续文件续传。

4、开发工具要求：Linux环境，GCC工具链。

# 2设计架构

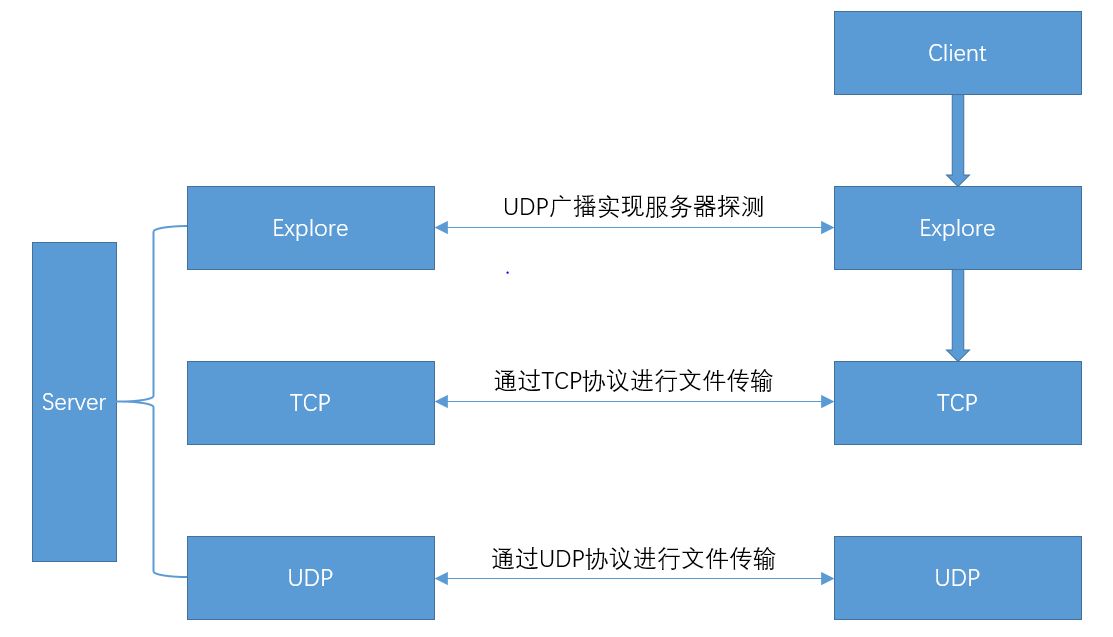


图1 文件传输设计架构

网络C/S文件传输设计架构如图1所示。服务器端同时开启三个服务线程，分别是服务器探测线程Explore、TCP协议服务线程和UDP协议服务线程。服务器探测部分基于UDP广播功能实现，客户端通过发送探测包探测出当前局域网中的所有可用的文件服务器，然后用户通过输入参数选择一个文件服务器，同时支持TCP和UDP两种文件传输协议，选定协议后，可实现与服务器的文件上传下载功能，文件传输同名文件直接覆盖，传输完成后利用sha256进行文件校验，保证文件传输内容准确无误。

## 2.1文件服务器探测

广播是一种允许一台主机发送单一[数据包](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%8C%85)到局域网内所有主机的[网络技术](https://baike.baidu.com/item/IP%E7%BD%91%E7%BB%9C%E6%8A%80%E6%9C%AF)。广播作为一点对多点的通信，是节省[网络带宽](https://baike.baidu.com/item/%E7%BD%91%E7%BB%9C%E5%B8%A6%E5%AE%BD)的有效方法之一。广播地址(Broadcast Address)是专门用于同时向网络中所有工作站进行发送的一个地址。在使用TCP/IP 协议的网络中，主机标识段host ID 为全1 的IP 地址为广播地址，广播的分组传送给host ID段所涉及的所有计算机。例如，对于10.1.1.0 （255.255.255.0 ）网段，其广播地址为10.1.1.255 （255 即为2 进制的11111111 ），当发出一个目的地址为10.1.1.255 的分组（封包）时，它将被分发给该网段上的所有计算机。。255.255.255.255这一特殊的广播地址可以向全世界进行广播，但一般路由器都会屏蔽。广播发送的目的MAC地址为FF:FF:FF:FF:FF:FF。

服务器探测实现步骤：

1. 服务器开启探测服务，监听客户端的探测
2. 客户端由于不知道可用文件服务器IP地址，利用广播发送探测包，其中广播地址设为：“255.255.255.255”，服务器的端口号设为：8848
3. 服务器接收到广播探测包之后，向发送探测包的客户端发送确认包
4. 客户端接收到确认包后，即知道文件服务器的IP地址，客户端输出所有探测到的可用文件服务器
5. 用户选择相应的服务器。

## 2.2 TCP文件传输

### 2.2.1 TCP文件传输流程

TCP提供基于IP环境下的数据可靠性传输，事先需要进行三次握手来确保数据传输的可靠性。TCP协议socket通信流程如图2所示。

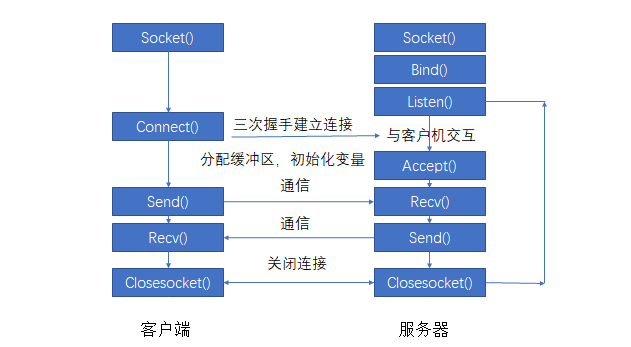


图2 TCP协议通信流程

### 2.2.2 TCP服务实现

TCP上传服务步骤：

1、客户端发送网络文件包，主要包含上传命令，文件路径名。

2、服务器接收文件信息。

3、服务器根据文件路径解析文件名，创建文件，同名文件直接覆盖。然后开启循环等待客户端发送文件内容。

4、客户端通过循环发送文件内容，发送完毕等待服务器发送文件校验码。

5、服务器文件接收完成，利用select机制跳出循环，进行文件校验，然后将校验信息发送给客户端。

6、客户端计算本地文件的校验码，然后对比服务器发送的校验码，进行文件校验。

TCP下载服务步骤：

1、客户端发送网络文件包，主要包含上传命令，文件路径名。然后开启循环等待接收服务器发送文件内容。

2、服务器检测文件路径是否合法，错误输出错误信息，文件路径合法，开启循环向客户端发送文件内容。发送完毕后等待客户端的接收完成通知，如果此时发送校验码会被客户端的文件接收循环解析。

3、客户端循环接收文件，文件接收完成，利用select机制跳出循环，客户端发送文件接收完成消息给服务器端

4、服务器发送文件校验码

5、客户端校对校验码。

### 2.2.3 TCP并发服务

服务器程序采用多线程实现客户端并发连接。server程序会开启socket监听每一个用户的连接，当客户端连入时，server程序开启新的线程负责连入客户端的通信。通过多线程可支持简单的并发，为了防止过多的连接，server队列设置上线20个，超出的客户端连接会处于等待状态。

## 2.3 UDP文件传输

### 2.3.1 UDP文件传输流程

用户数据报协议(UDP, User Datagram Protocol)是一种无连接的传输层协议，它不必等待连接建立后才能进行通信。该服务对数据的传输提供不可靠、尽力的传输。UDP属于报文流传输，UDP对应用层报文封装、传输，但不会拆分，也不合并。UDP协议通信流程如图3所示。

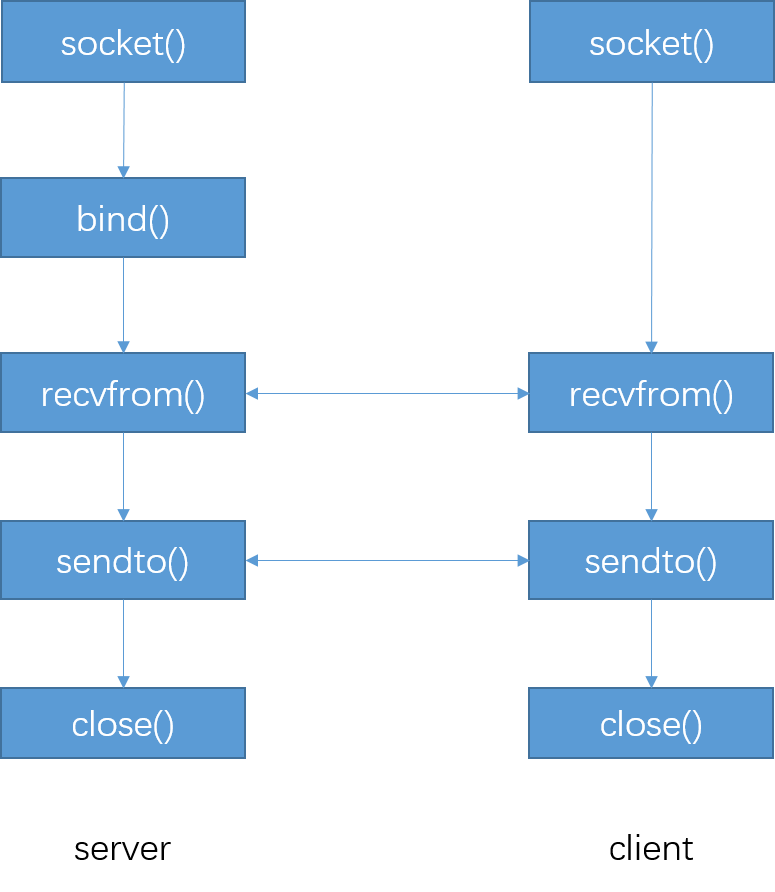


图3 UDP协议通信流程

### 2.3.2 UDP服务实现

UDP上传服务步骤：

1、客户端发送网络文件包，主要包含上传命令，文件路径名。

2、服务器接收文件信息。

3、服务器根据文件路径解析文件名，创建文件，同名文件直接覆盖。然后开启循环等待客户端发送文件内容。

4、客户端通过循环发送文件内容，每一次发送文件都要在循环内等待服务器接收确认通知，因为UDP传输包不会合并。最后文件发送结束后，向服务器发送0字节数据帧表明数据发送完毕。客户端等待服务器发送文件校验码。

5、服务器文件循环接收，每一次接收都会向客户端发送确认接收消息直至完成。利用客户端发送的0字节数据帧跳出循环，进行文件校验，然后将校验信息发送给客户端。

6、客户端计算本地文件的校验码，对比服务器发送的校验码，进行文件校验。

UDP下载服务步骤：

1、客户端发送网络文件包，主要包含上传命令，文件路径名。然后开启循环等待接收服务器发送文件内容。

2、服务器检测文件路径是否合法，错误输出错误信息，文件路径合法，开启循环向客户端发送文件内容，每一次发送文件都要在循环内等待服务器接收确认通知，因为UDP传输包不会合并。最后文件发送结束后，向客户端发送0字节数据帧表明数据发送完毕。等待客户端发送请求验证码。

3、客户端循环接收文件，每一次接收都会向客户端发送确认接收消息直至完成。文件接收完成，利用发送的0字节数据帧跳出循环，客户端发送文件接收完成消息给服务器端，请求发送验证码。

4、服务器发送文件校验码。

5、客户端校对校验码。

## 2.4文件协议

为保证通信双方能可靠有效通信规定的一系列文件协议，文件协议保存在procotol.h文件中，主要包括命令包交互格式、网络协议包格式、文件缓冲区大小、延迟、重试次数等。

## 2.5文件校验

sha256是SHA-2下细分出的一种算法，SHA-2，名称来自于安全散列算法2（英语：Secure Hash Algorithm 2）的缩写，一种密码散列函数算法标准，由属于SHA算法之一，是SHA-1的后继者。

sha256是一个哈希函数，是一种从任何一种数据中创建小的数字“指纹”的方法。散列函数把消息或数据压缩成摘要，使得数据量变小，将数据的格式固定下来。该函数将数据打乱混合，重新创建一个叫做散列值（或哈希值）的指纹。散列值通常用一个短的随机字母和数字组成的字符串来代表。对于任意长度的消息，sha256都会产生一个256bit长的哈希值，称作消息摘要。这个摘要是个长度为32个字节的数组，用一个长度为64的十六进制字符串来表示。

本项目使用sha256算法进行文件校验。在文件传输完成后，调用sha256算法计算文件内容产生一个256bit长的哈希值。服务器程序会将哈希值以字符串的形式发送给客户端，客户端同样使用sha256计算本地上传/下载的文件内容产生哈希值，然后对比两个哈希值，如果值相同说明文件传输准确无误。

# 3 演示文档

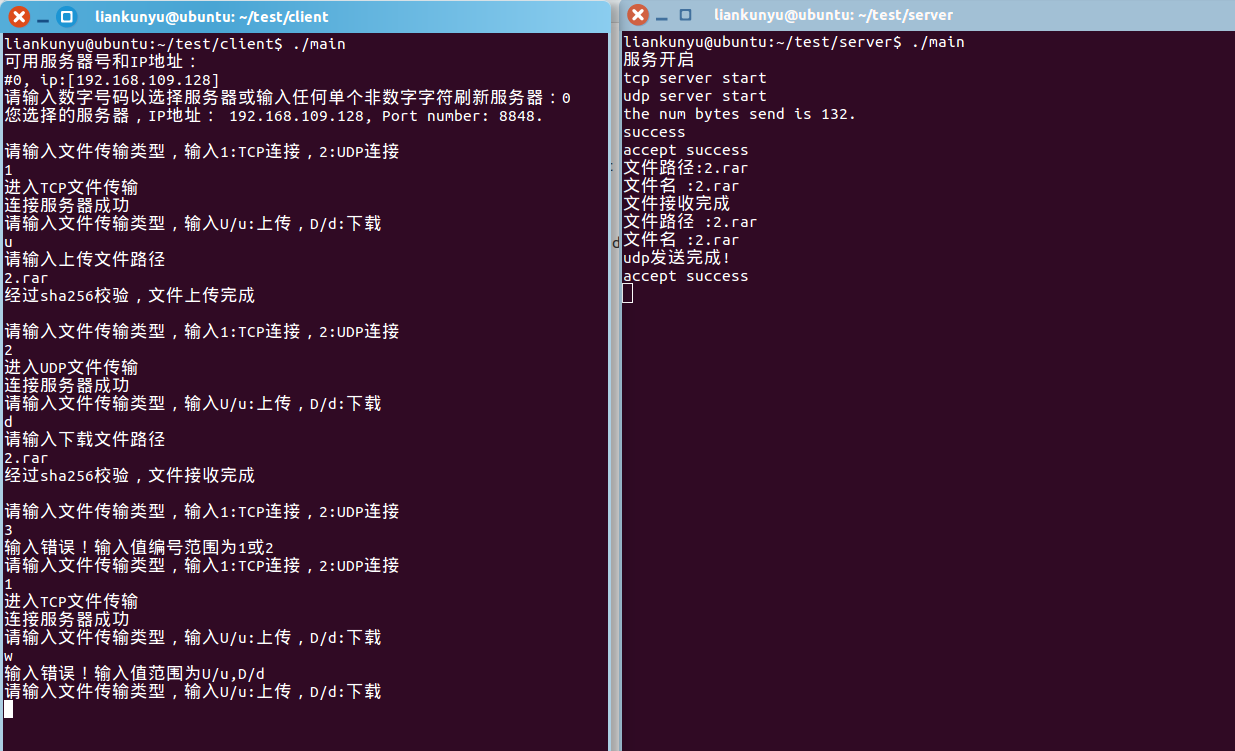


图4 网络C/S文件传输用户界面

网络C/S文件传输的用户界面如图4所示。具体操作步骤如下：

1. 在shell界面运行服务器端程序server，程序运行之后服务器端开启服务器探测支持、TCP协议支持和UDP协议支持服务。
2. 客户端程序client开启之后会发送探测包查询局域网中可用的服务器，查询完毕会将可用服务器号和IP地址打印在用户界面，用户通过选择的服务器号选择相应的文件服务器，或者输入任何非数字单字符进行刷新可用服务器。
3. 用户选择相应服务器号，例如输入0之后，界面会打印出相应IP和port，然后程序提示用户输入文件传输类型TCP/UDP。用户输入1：TCP服务，程序提示用户选择可用功能，U/u上传或D/d下载。
4. 用户输入u，程序提示用户输入文件路径，路径输入完成，文件开始上传，完成提示文件传输完成，文件经过sha256校验文件，内容准确无误。

错误提示：

1. 如图5所示，当服务器没有开启或者没有可用服务器时，客户端程序会重试发送探测包，发送一定次数后确认局域网中没有可用的服务器，

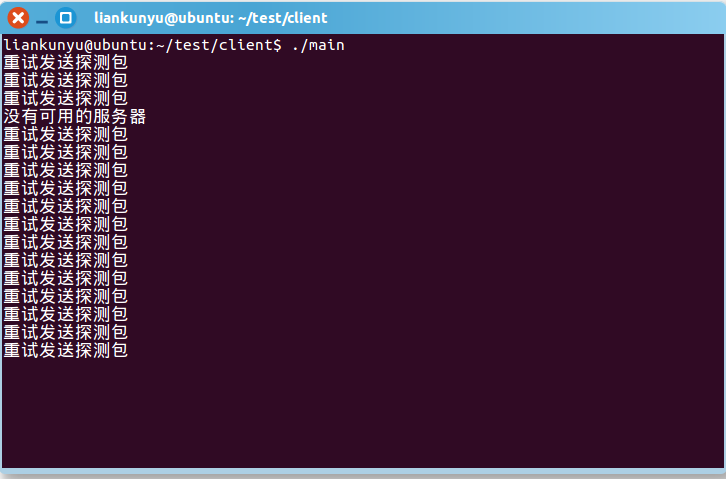


图5 客户探测包发送界面

2、如图6所示，在用户文件传输类型选择错误时，程序提示错误和相应输入值范围，直至用户输入正确值。

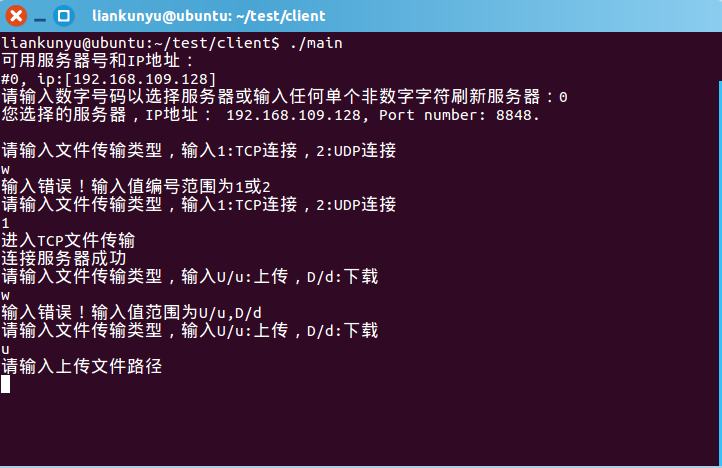


图6 参数选择错误提示

# 4 测试文档

## 4.1测试方案

程序采用白盒测试方法，选用三个不同大小的文件进行测试，三个文件分别是1.sh、2.rar和3.rar。测试三个文件在TCP/UDP协议下，上传/下载成功率，以及每个文件传输用时。

## 4.2测试用例

三个文件：1.sh、2.rar、3.rar

程序：client、server

环境：Ubuntu、局域网

## 4.3测试结果报告

测试结果如表1所示，在客户端程序中使用gettimeofday()函数对文件传输进行计时，结果输出用户界面。结果显示在小文件下UDP耗时很短，但是在较大文件时UDP传输却比TCP耗时长，文件的上传下载用时相差不大。

表1 测试结果

|  |  |  |  |
| --- | --- | --- | --- |
| 文档名称 | 1.sh | 2.rar | 3.rar |
| 文档大小 | 170byte | 1.4MB | 261.3MB |
| TCP协议上传次数/成功次数 | 10/10 | 10/10 | 5/5 |
| TCP协议下载次数/成功次数 | 10/10 | 10/10 | 5/5 |
| UDP协议上传次数/成功次数 | 10/10 | 10/10 | 5/5 |
| UDP协议下载次数/成功次数 | 10/10 | 10/10 | 5/5 |
| TCP上传平均耗时（ms） | 503 | 510 | 1424 |
| TCP下载平均耗时（ms） | 501 | 503 | 1276 |
| UDP上传平均耗时（ms） | 1.449 | 13.7 | 2614 |
| UDP下载平均耗时（ms） | 1.444 | 12.7 | 2393 |