华南理工大学

《计算机网络》课程实验报告

实验题目： 实验五：网络Socket编程

姓名： 曾震 学号： 202030443356

班级： 计科2 组别：

合作者：

指导教师： 杜广龙

|  |
| --- |
| **实验概述** |
| 【实验目的及要求】  实验目的：  （1）掌握Socket网络编程的基本原理和方法。  （2）深刻理解Socket的底层运作原理。  （3）通过实践加深对计算机网络体系结构和运行机制的理解。  （4）提高编程和分析问题，解决问题的能力。  实验要求：   1. 基本要求：实现一个FTP协议的客户端和服务端，完成基本的文件传输功能。 2. 提交的内容：文档报告（包括设计文档，使用说明）， 源代码，可执行程序，以上内容分成三个文件夹存放（分别是Doc、Src、bin），再统一打包提交到教学在线。 3. 建议附加功能（可酌情获得加分，有附加功能的FTP可以分组进行，最多不超过3人）    * 多客户端访问。    * 在获取文件之前能够先得到文件列表。    * 有兴趣的同学可以提交其它的作品，但作品必须基于底层的Socket(具体见实验要求)，不能使用高层封装的Socket（如Java类库，MFC等）。   【实验环境】  实验平台：Windows  开发语言：C++  开发环境：Visual Studio |
| **实验内容** |
| 【实验过程】  一、实验步骤：  1、本次实验主要基于windows系统实现socket编程，因此先对socket库中的函数进行了解。本次实验主要使用到的函数如下：  WSAStartup(a, b)函数：该函数用于初始化socket库，两个参数代表其版本号和初始化对象。  WSACleanup()函数：用于解除与Socket库的绑定并且释放Socket库所占用的系统资源。  Socket函数，用来创建一个能够进行网络通信的套接字，有三个参数，第一个参数指定应用程序使用的通信协议的协议族，对于TCP/IP协议族，该参数置PF\_INET;第二个参数指定要创建的套接字类型，流套接字类型为SOCK\_STREAM、数据报套接字类型为SOCK\_DGRAM；第三个参数指定应用程序所使用的通信协议。主要用于创建服务端的监听套接字，和客户端的连接套接字。  Send函数，用来发送数据。  Recv函数，用来接收数据。  Bind函数，用于绑定socket的地址（IP+Port），一般用于服务端。  Listen函数，第一个参数是套接字序号，第二个参数是最多容纳连接请求，能使套接字处于监听状态。  Accept函数，服务程序调用accept函数从处于监听状态的流套接字s的客户连接请求队列中取出排在最前的一个客户请求，并且创建一个新的套接字来与客户套接字创建连接通道，如果连接成功，就返回新创建的套接字的描述符，以后与客户套接字交换数据的是新创建的套接字。  Connect函数：客户程序调用connect函数来使客户Socket s与监听于name所指定的计算机的特定端口上的服务Socket进行连接。  二、实验数据：  在实验结果中进行展示。  三、实验主要过程：   1. 首先是客户端和服务端共有的部分，编写函数对 socket库进行初始化和关闭操作。        1. 其次是建立连接过程。   服务端作为请求的接收方，需要先创建套接字，然后对套接字进行地址（IP+端口）的绑定，同时启用listen函数：表示对该IP的某端口进行监听，监听来自客户端的套接字connect请求，然后调用accept函数对客户端请求进行接收，accept函数执行成功就会返回一个新的套接字，在这里可以理解为“连接套接字”，而之前创建的为“监听套接字”，后续通过while循环就能对客户端发来的消息进行处理。  服务端的监听端口我这里选用8888，也就是SPORT，并监听所有网卡。  客户端的连接端口也是用8888，IP地址我选用本地：127.0.0.1，方便进行测试。      客户端的处理大同小异，但是因为客户端不用对套接字进行地址的绑定，系统会自动进行分配，代码如下：    建立好连接后，首先由客户端选择服务：上传/下载，其实现方式大同小异，这里重点讲解下载。首先服务器端会通过不断循环processMsg这个函数来对消息进行处理，客户端一旦send了一个消息，服务器端就进行recv操作，然后对文件中的字符数组进行强制类型转换，转换成自定义结构体：Header，其中定义了消息类型MsgTag，用来标记本次传输的信息是什么内容的。同时，还有结构体fileinfo，用来封装文件的属性信息；fileData：用来封装具体文件的内容，同时fileData中还有定义对大文件的拆包开始位置，和每次传递包的内容大小，后续会在传递文件中提及。  以下是Header结构体部分：    服务器recv到消息后，根据msg中的标记来进行路由选择不同的处理函数，代码如下：    客户端在这方面大致内容相同，只是在下载功能的实现上，客户端和服务端使用的标记会不相同。这里就不对服务端的recv循环进行代码展示。   * 接下来就是客户端与服务端下载的信息传递过程：  1. 首先是客户端向服务端发起请求，调用getFileName函数，输入文件的绝对路径，表示要从服务端下载的文件路径是什么。然后将消息头标记为MSG\_FILENAME，然后send。（该函数较为简单，不进行展示） 2. 服务端接收到客户端的MSG\_FILENAME 的消息头标记后，通过switch路由，调用readFile() 函数，首先客户端用fopen对文件进行打开，并用FILE\*类型数据pread进行接收。如果打开失败，就返回错误信息；成功打开的话，就通过fseek()函数，遍历文件大小，并将文件大小放入回复消息体msg中的fileinfo中，然后使用\_splitpath函数对绝对路径进行分解，将文件名和文件类型进行拼装后，再放入msg的fileinfo中。其实就是将该文件的名称，和文件大小send回给客户端。本部分代码如下：     在send之后，还要将该文件读入到一个全局变量中，该变量是一个动态数组，之前已经获取到了该文件的大小，现在可以通过calloc函数，分配内存空间。     1. 这时消息传递回客户端，客户端接收到文件大小后调用readyRea函数，同理使用calloc函数，动态开辟一块内存空间。并返回一个标记位为MSG\_READY\_READ的消息头，代码如下：      1. 服务端在接收到MSG\_READY\_READ请求头标记后，就可以开始发送数据了，因为在上一步已经将文件内容都放进全局的动态数组中，因此这里可以进行发送。但是这里需要注意一个关键的问题，在这个程序中，我使用的数组大小都是1024，也就是说每次传送消息头之前，都是把消息头强制类型转换成(char\*)类型，然后接收消息时(processMsg函数中)，在强制类型转换回自定义类型Header，因此就需要规定每次消息头大小最大为1024。因此在自定义类型中有细节处理，就是每个fileData不能太大，已经固定了最大字节数PACKET\_SIZE，这个在之前结构体的代码展示中已经有了。   因此在后续发送文件过程中，要把文件拆分成一个一个PACKET\_SIZE大小的块，放进Header中进行传输，每次传输都会带有标记nstart，表示这是第几个块，还有nsize: 表示这个块有多少有效数据。循环直到最后一个块可能小于等于PACKET\_SIZE，就发送最后一个包。代码如下：     1. 服务端会不断接收数据，每次接收都会进行判断是不是最后一个块，这个可以通过传来的消息体中的nsize，nstart，和已经保存的全局变量filesize文件大小，进行比较，看看是不是最后一个。同时每次接收到一个块，都复制到对应开辟的内存块位置上。代码如下：     上述就是代码的实现原理和过程。文件的上传和下载函数是基本一致的，只是在switch路由的时候会有不同，因此不过多赘述。  四、实验结果展示：  先将两个可执行文件分别放在两个文件夹中，一个代表客户端，一个代表服务器端。    1、先启动服务端程序，再启动客户端程序，界面如下：    2、如果选择1，下载文件：    3、输入想下载的文件名（绝对路径，可以直接拖拽进命令行）：    回车后，传输完成，下载的文件就在客户端文件夹中了    并且可以正常打开。  4、选择上传功能，将client中的文件上传，最终结果会在服务器端的文件夹中。      实验完毕。 |
| **小结** |
| **本次实验通过使用传输层的Socket库，来实现服务器的连接和文件的传输，通过对Socket库的学习，让我更加深入的了解了如何使用套接字接口来实现网络通讯，也实现了一套文件传输系统，也对C/S模型做出了相应的实现，让我结合实际了解到了更多知识，受益良多。** |
| **指导教师评语及成绩** |
| 评语：  成绩：           指导教师签名：杜广龙                                                 批阅日期： |