**计算机网络编程**

**实验报告**

**班级：07111707**

**组长：1120171189 崔程远**

**成员：1120172149 吴沁璇**

**1120172153 张澈**

**1120172163 王晓媛**

**1120172736 张鉴昊**

**1120172765 曾煜瑾**

**1120173326 曾紫飞**

**北京理工大学**

**计算机学院**

**2020年5月**

**第六章 实验1 UDP协议服务器和客户**

**1. 实验目的**

学习UDP通信协议，掌握UDP socket通信方法。

1. **实验内容**

客户发送命令行文本给服务器，服务器转换大写后返回给客户并显示。

配置文件关键要点：

无，对方的IP地址、端口以及发送串以命令行参数的形式提供程序运行

1. **实验原理**

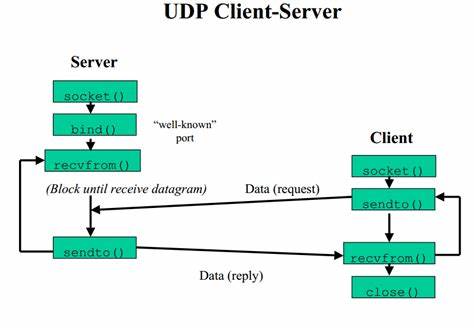
socket起源于Unix，而Unix/Linux基本哲学之一就是“一切皆文件”，都可以用“打开open –> 读写write/read –> 关闭close”模式来操作。我的理解就是Socket就是该模式的一个实现，socket即是一种特殊的文件，一些socket函数就是对其进行的操作（读/写IO、打开、关闭）。

UDP编程的服务器端一般步骤是：

1. 创建一个socket，用函数socket()；
2. 设置socket属性，用函数setsockopt();\* 可选
3. 绑定IP地址、端口等信息到socket上，用函数bind();
4. 循环接收数据，用函数recvfrom();
5. 关闭网络连接；

UDP编程的客户端一般步骤是：

1. 创建一个socket，用函数socket()；
2. 设置socket属性
3. 设置对方的IP地址和端口等属性;
4. 发送数据，用函数sendto();
5. 关闭网络连接；



图源网络

1. **实验环境**

|  |  |  |
| --- | --- | --- |
| 语言 | 集成开发环境 | 编译器 |
| C++ | Visual Studio 2017 | gcc version 4.8.1 |
| Java | Eclipse 2019 | java version "1.8.0\_65" |
| Python | Pycharm 2017 | Python 3.7.0 |

1. **实验步骤**

各个语言的实现都分为服务端和客户端。

·C语言版本

服务端首先判断接收参数数量是否符合要求，之后构建socket并绑定到本地相应端口；通过resvMsg(server\_fd, &str, &cli\_addr)函数接收来自客户端的字符串并获得客户端的信息；通过sendMsg(server\_fd, str, cli\_addr)函数和之前获得的客户端信息向客户端发送经过处理的字符串。

客户端首先判断接收参数数量是否符合要求，之后构建socket并绑定到本地相应端口（客户端实际可以不需要绑定），通过sendMsg(server\_fd, str, cli\_addr)函数向服务端发送待处理的字符串；通过resvMsg(server\_fd, &str, &cli\_addr)函数接收处理过的字符串。

由于刚接触socket编程这部分实现较为冗杂，后续Java和Python版本更简洁一些。

·Java语言版本和Python语言版本

服务端判断参数数量后创建Socket对应语言支持的socket对象，通过该对象的方法来接收和发送字符串。

Java中DatagramPacket的getSocketAddress()方法可用于获得包的来源地址，用于确定客户端地址。

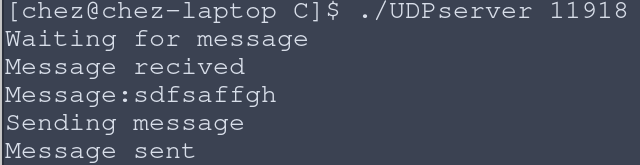
Python中socket的recvfrom()方法同时返回通信内容和来源地址信息，可据此确定客户端地址。

1. **运行结果**

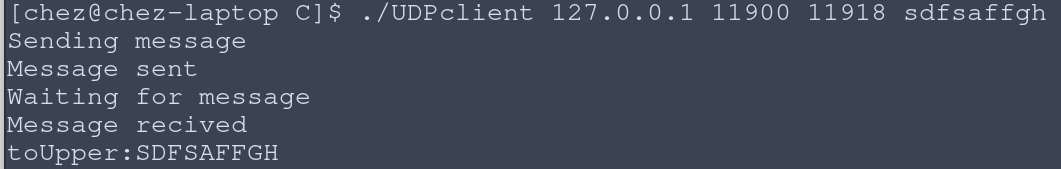
Send函数中将01100000000000111110000000000110作为待发送的数据信息。

**·C**

服务端：输入开放的端口号

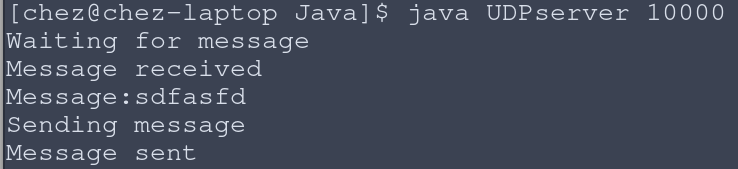


客户端：依次输入目标IP、本地开放端口、目标端口、待处理字符串



**·Java**

服务端：输入开放的端口号

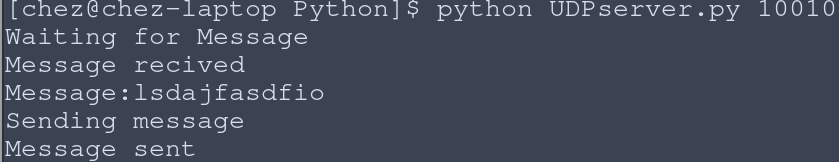


客户端：依次输入目标IP、本地开放端口、目标端口、待处理字符串

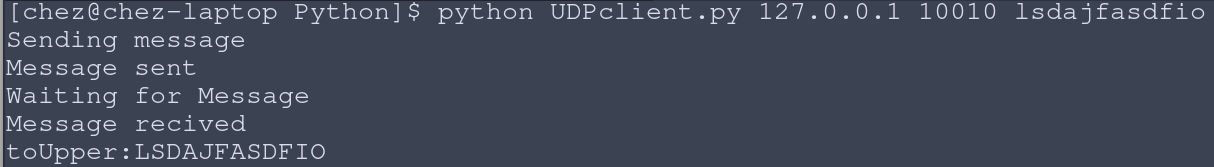


**·Python**

服务端：输入开放的端口号



客户端：依次输入目标IP、目标端口、待处理字符串



1. **实验总结**

实验内容相对简单，但由于对Socket的使用还不熟练，在第一个程序（C版本）的编写过程中把简单问题复杂化了，之后的编写就很轻松。由于各种语言对UDP socket的实现机制不同，不同系统对socket的支持也不同（POSIX标准和Winsocket），因此需要查阅相关资料。

总的来说，这个实验使我加深了对UDP通信机制的理解，提高了编程能力。