厦門大學

信息学院软件工程系

《计算机网络》实验报告

题　　目 实验五　利用 Socket API 实现许可认证软件

班　　级 软件工程 2020 级卓越班

姓　　名 吴佳熙

学　　号 22920202204692

实验时间 2022 年 5 月 3 日

2022 年 5 月 3 日

填写说明

1、本文件为 Word 模板文件，建议使用 Microsoft Word 2019 打开，在可填写的区域中如实填写；

2、填表时勿破坏排版，勿修改字体字号，打印成 PDF 文件提交；

3、文件总大小尽量控制在 1MB 以下，最大勿超过 5MB；

4、应将材料清单上传在代码托管平台上；

5、在实验课结束 14 天内，按原文件发送至课程 FTP 指定位置。

1 实验目的

通过完成实验，掌握应用层文件传输的原理；了解传输过程中传输层协议选用、应用层协议设计和协议开发等概念。

2 实验环境

操作系统：Win10

编程语言：C++、python

3 实验结果

本实验使用 C++编写服务器和客户端的代码部分，使用 python 编写 CS_project，实现软件的对外接口，调用 g++命令编译和执行服务器和客户端的代码，实现许可认证软件。

C++编写的服务器和客户端部分：

由于本实验使用的是 windows 系统，部分 linux 系统的头文件无法使用，所以使用了以下替换文件：

Linux 头文件	Windows 头文件
#include <sys/socket.h>	#include <winsock2.h> #pragma comment(lib, “wsock32.lib”) #include<ws2tcpip.h>
#include<netinet.h>	
#include<netinet/in.h>	
#include<netdb.h>	
#include<arpa/inet.h>	#include <windows.h> #pragma comment(lib, “ws2_32.lib”)
#include<sys/types.h>	
#include<unistd.h>	
#include<unistd.h>	

当遇到套接字未正确创建或连接失败等情况时，程序会终止。

```
/* Convert host name to equivalent IP address and copy to sad. */
ptrh = gethostbyname(host);
if (((char*)ptrh) == NULL) {
    fprintf(stderr, "invalid host: %s\n", host);
    exit(1);
}
memcpy(&sad.sin_addr, ptrh->h_addr, ptrh->h_length);
/* Map TCP transport protocol name to protocol number. */
if (((unsigned long long)(ptrp = getprotobyname("tcp"))) == 0) {
    fprintf(stderr, "cannot map \"tcp\" to protocol number");
    exit(1);
}
/* Create a socket. */
sd = socket(PF_INET, SOCK_STREAM, ptrp->p_proto);
if (sd < 0) {
    fprintf(stderr, "socket creation failed\n");
    exit(1);
}

/* Connect the socket to the specified server. */
if (connect(sd, (struct sockaddr*)&sad, sizeof(sad)) < 0) {
    fprintf(stderr, "connect failed\n");
    exit(1);
}
```

```
if (port > 0) /* test for illegal value */
    sad.sin_port = htons((u_short)port);
else { /* print error message and exit */
    fprintf(stderr, "bad port number %s\n", argv[1]);
    exit(1);
}

/* Map TCP transport protocol name to protocol number */
if (((unsigned long long)(ptrp = getprotobyname("tcp"))) == 0) {
    fprintf(stderr, "cannot map \"tcp\" to protocol number");
    exit(1);
}

/* Create a socket */
sd = socket(PF_INET, SOCK_STREAM, ptrp->p_proto);
if (sd < 0) {
    fprintf(stderr, "socket creation failed\n");
    exit(1);
}

/* Bind a local address to the socket */
if (bind(sd, (struct sockaddr*)&sad, sizeof(sad)) < 0) {
    fprintf(stderr, "bind failed\n");
    exit(1);
}

/* Specify size of request queue */
if (listen(sd, QLEN) < 0) {
    fprintf(stderr, "listen failed\n");
    exit(1);
}
```

client.c 中设定了如果正确接收到套接字，则打印其中的数据内容和读取的字符数。

```
/* Repeatedly read data from socket and write to user's screen. */
n = recv(sd, buf, sizeof(buf), 0);
while (n > 0) {
    printf("1");
    printf("\n%d", buf);
    printf("\n%d", n);
    n = recv(sd, buf, sizeof(buf), 0);
}
```

使用 g++命令编译 server.c，生成 server.exe 可执行文件，用于启用服务器进行监听。

```
D:\大二下\计网\实验\实验五\CS_project\CS_project>g++ server.c -o server -lwsock32
server.c:3: warning: "WIN32" redefined
 #define WIN32

<built-in>: note: this is the location of the previous definition
server.c:37:5: warning: second argument of 'int main(int, char*)' should be 'char **' [-Wmain]
 int main(int argc, char* argv)
```

使用 g++命令编译 client.c，生成 client.exe 可执行文件，用于生成客户端，可用于发送请求。

```
D:\大二下\计网\实验\实验五\CS_project\CS_project>g++ client.c -o client -lwsock32
client.c:3: warning: "WIN32" redefined
 #define WIN32

<built-in>: note: this is the location of the previous definition
client.c:39:5: warning: second argument of 'int main(int, char*)' should be 'char **' [-Wmain]
 int main(int argc, char* argv)
```

可以通过 server 执行 server.exe 可执行程序。

```
D:\大二下\计网\实验\实验五\CS_project\CS_project>server
```

可以通过 client 执行 client.exe 可执行程序。

```
D:\大二下\计网\实验\实验五\CS_project\CS_project>client
```

python 编写的对外接口部分：

CS_project.py 通过 os.system 调用命令行，编译和执行 server 和 client 程序。

```
#编译服务器和客户端
compile_server="g++ .\CS_project\CS_project\server.c -o .\CS_project\CS_project\server -lwsock32"
compile_client="g++ .\CS_project\CS_project\client.c -o .\CS_project\CS_project\client -lwsock32"
os.system(compile_server)
os.system(compile_client)
```

编译结果与直接在终端中编译返回的信息和结果相同：

```
(base) D:\大二下\计网\实验\实验五>C:/ProgramData/Miniconda3/python.exe d:/大二下/计网/实验/实验五/CS_project.py
.\CS_project\CS_project\server.c:3: warning: "WIN32" redefined
 #define WIN32

<built-in>: note: this is the location of the previous definition
.\CS_project\CS_project\server.c:37:5: warning: second argument of 'int main(int, char*)' should be 'char **' [-Wmain]
 int main(int argc, char* argv)
     ^~~~
.\CS_project\CS_project\client.c:3: warning: "WIN32" redefined
 #define WIN32

<built-in>: note: this is the location of the previous definition
.\CS_project\CS_project\client.c:39:5: warning: second argument of 'int main(int, char*)' should be 'char **' [-Wmain]
 int main(int argc, char* argv)
     ^~~~
```

执行结果与直接在终端中执行返回的信息和结果也相同：

```
#打开服务器
open_server=".\CS_project\CS_project\server"
os.system(open_server)
```

```
create_client=".\CS_project\CS_project\server" #用户运行软件，向许可证服务器发送验证
os.system(create_client)
```

CS_project 实现了输入用户名、口令和许可证类型，发放许可证，记录已发放许可证的数量和使用人数等功能。序列号通过随机数生成，并判断之前是否已经使用过该序列号。

```
num=random.randint(1000000000,9999999999) #随机生成十位数序列号

for i in range(len(used_num)): #判断该序列号是否使用过
    if(used_num[i]==num):
        num=random.randint(1000000000,9999999999)
        i=0

used_num.append(num)
print("您的序列号是："+str(num)+'\n') #输出序列号
```

```
是否需要购买许可证？（Y/N）Y
请输入用户名：wujx
请输入口令：123456
请输入许可证类型：TCP
您的序列号是：4760218981
```

同时可根据使用人数判断是否继续允许连接。

```
if(connect_people<MAX_NUM):
```

```
else:
    #达到最大连接人数时退出程序
    print("目前许可证使用已达到最大人数，无法连入")
```

```
是否需要使用远程桌面连接软件（Y/N）：
Y
请输入您的序列号：4760218981
```

实现了定期（程序设定是 1 分钟）向服务器报告其状态，显示许可证发放数量和连接人数。

```
end_time=time.time()
if(end_time-start_time>60):
    start_time=time.time()
    print("当前许可证发放"+str(license_num)+"张")
    print("目前有"+str(len(used_num))+"人在使用该软件")
```

```
当前许可证发放1张
目前有1人在使用该软件
```

使用了 while(True)循环，可以实现非用户主动退出，无法退出程序。即可以在许可证服务器崩溃后，重新启动并恢复。

```
elif(choice_buy=='Y'):
    while(True):
        if(license_num!=0):
            choice_buy=input("是否需要购买许可证？（Y/N）")
        if(choice_buy=='N'):
            print("欢迎下次使用！再见！")
            break #用户主动退出程序
        username=input("请输入用户名：")
        password=input("请输入口令：")
        type=input("请输入许可证类型：")
```

4 实验代码

本次实验的代码已上传于以下代码仓库：https://gitee.com/wujx0206/CNI-Exp/tree/master/E5_4692

5 实验总结

通过本次实验，我进一步学习了应用层文件传输的原理，了解了传输过程中传输层协议选用、应用层协议设计和协议开发等概念。

本次实验使用 python 和 C++编程，实现了利用 Socket API 实现了许可认证软件。混合编程可以充分利用各语言的优势，python 通过命令行调用 C++程序，可以提高程序效率。