**海南大学计算机与网络空间安全学院**

**实 验 报 告**

|  |  |
| --- | --- |
| **评定成绩** | **指导教师** |
|  |  |

**实验课程：** 网络协议与分析

**实验名称：**UDP广播与接受-电子点名系统

1. 实验目的
   1. 掌握网络应用程序的开发方法
   2. 掌握Client/ Server结构软件的设计与开发方法
   3. 掌握Socket机制的工作原理
   4. 掌握UDP协议原理与广播机制
2. 实验任务

在教师机（服务器）上发送UDP广播，要求所有学生机（客户端）接受点名，学生在收到点名信息后回复自己的学号、姓名以及所在的机器名。

1. 实验环境

Win10+visual studio2019

1. 实验内容

服务器端代码实现：

#include "Winsock2.h""

#include "stdio.h"

#include "string.h"'

#pragma comment(lib, "wsock32.lib")

#define RECV\_PORT 3000

#define SEND\_PORT 3001

SOCKET sock;

SOCKET ServerSock;

sockaddr\_in ServerAddr;

sockaddr\_in broadAddr;

**void** main()

{

    WSADATA WSAData;

**if** (WSAStartup(MAKEWORD(2, 2), &WSAData) != 0)

    {

        printf("套接字初始化失败!\n");

**return**;

    }

    sock = socket(AF\_INET, SOCK\_DGRAM, 0);

**if** (sock == SOCKET\_ERROR)

    {

        printf("套接字创建失败!\n");

        WSACleanup();

**return**;

    }

    ServerSock = socket(AF\_INET, SOCK\_DGRAM, 0);

**if** (ServerSock == SOCKET\_ERROR)

    {

        printf("套接字创建失败!\n");

        WSACleanup();

**return**;

    }

    memset(&ServerAddr, 0, **sizeof**(ServerAddr));

    ServerAddr.sin\_family = AF\_INET;

    ServerAddr.sin\_addr.s\_addr = inet\_addr("10.210.225.191");

    ServerAddr.sin\_port = htons(RECV\_PORT);

    memset(&broadAddr,0,**sizeof**(broadAddr));

    broadAddr.sin\_family = AF\_INET;

    broadAddr.sin\_addr.s\_addr = inet\_addr("255.255.255.255");

    printf("广播地址: %s \n", inet\_ntoa(broadAddr.sin\_addr));

    broadAddr.sin\_port = htons(SEND\_PORT);

**bool** bBroadcast = **true**;

**int** n = setsockopt(sock, SOL\_SOCKET, SO\_BROADCAST, (**const** **char**\*)&bBroadcast, **sizeof**(bBroadcast));

**int** opt = **sizeof**(bBroadcast);

    getsockopt(sock, SOL\_SOCKET, SO\_BROADCAST, (**char**\*)&bBroadcast, &opt);

**if** (bBroadcast == **true**)

    {

        printf("\n测试是否可以广播: 是\n");

    }

**else**

    {

        printf("\n测试是否可以广播: 否 \n");

**return**;

    }

**char** buf[1024] = "/name:开始点名！\0";

    n = sendto(sock, buf, strlen(buf) + 1, 0, (**const** sockaddr\*)&broadAddr, **sizeof**(broadAddr));

**if** (n == SOCKET\_ERROR)

    {

        printf("广播数据错误!\n");

        printf("错误代码:%d\n", WSAGetLastError());

    }

**else**

    {

        printf("\n发送广播数据:\"%s\" 成功! \n\n", buf);

    }

**if** (bind(ServerSock, (**struct** sockaddr\*)&ServerAddr, **sizeof**(ServerAddr)) == SOCKET\_ERROR)

    {

        printf("绑定失败!\n");

        printf("错误代码:%d\n", WSAGetLastError());

        closesocket(ServerSock);

        getchar();

**return**;

    }

**else**

    {

        printf("绑定成功！开始接受来自客户端的数据...\n");

    }

**while** (1)

    {

        sockaddr\_in ClientAddr;

        memset(&buf, '\0', **sizeof**(buf));

        memset(&ClientAddr, 0, **sizeof**(ClientAddr));

        n = **sizeof**(ClientAddr);

        n = recvfrom(ServerSock, buf, **sizeof**(buf), 0, (**struct** sockaddr\*)&ClientAddr, &n);

**if** (n != SOCKET\_ERROR)

        {

            printf("\n 接受远程主机 %s 传回的数据：%s \n", inet\_ntoa(ClientAddr.sin\_addr), buf);

        }

    }

    closesocket(sock);

    closesocket(ServerSock);

    WSACleanup();

}

客户端代码实现：

#include "Winsock2.h""

#include "stdio.h"

#include "string.h"

#pragma comment(lib, "wsock32.lib")

#define RECV\_PORT 3001

#define SEND\_PORT 3000

SOCKET sock;

sockaddr\_in ServerAddr;

sockaddr\_in ClientAddr;

**void** main()

{

**int** n;

    WSADATA WSAData;

**if** (WSAStartup(MAKEWORD(2, 2), &WSAData) != 0)

    {

        printf("套接字初始化失败!\n");

**return**;

    }

    sock = socket(AF\_INET, SOCK\_DGRAM, 0);

**if** (sock == SOCKET\_ERROR)

    {

        printf("套接字创建成功!\n");

        WSACleanup();

**return**;

    }

    memset(&ClientAddr, 0, **sizeof**(ClientAddr));

    ClientAddr.sin\_family = AF\_INET;

    ClientAddr.sin\_addr.s\_addr = inet\_addr("10.210.225.191");

    ClientAddr.sin\_port = htons(RECV\_PORT);

    n = bind(sock, (**struct** sockaddr\*)&ClientAddr, **sizeof**(ClientAddr));

**if** (n == SOCKET\_ERROR)

    {

        printf("绑定失败!\n");

**return**;

    }

**else**

    {

        printf("绑定成功!\n");

    }

**char** buf[1024] = "\0";

**int** len;

    printf("开始接受数据...\n");

**while** (1)

    {

        memset(&buf, '\0', **sizeof**(buf));

        memset(&ServerAddr, 0, **sizeof**(ServerAddr));

        len = **sizeof**(ServerAddr);

        printf("开始接受数据...\n");

        n = recvfrom(sock, buf, **sizeof**(buf), 0, (**struct** sockaddr \*)&ServerAddr, (**int** \*)&len);

**if** (n == SOCKET\_ERROR)

        {

            printf("\n 获取广播数据失败! \n'");

            printf("\n 错误代码: %d\n", WSAGetLastError());

        }

**else**

        {

            printf("\n 获取广播数据: \" % s\" 来自 %s \n", buf, inet\_ntoa(ServerAddr.sin\_addr));

**if** (strcmp(buf, "/nane") >= 0)

            {

                ServerAddr.sin\_family = AF\_INET;

                ServerAddr.sin\_addr.s\_addr = inet\_addr("10.210.225.191");

                ServerAddr.sin\_port = htons(SEND\_PORT);

                memset(&buf, '\0', **sizeof**(buf));

                strcpy(buf, "20181685310031 张迈");

                n = sendto(sock, buf, strlen(buf) + 1, 0, (**struct** sockaddr\*)&ServerAddr, **sizeof**(ServerAddr));

**if** (n == SOCKET\_ERROR)

                {

                    printf("发送失败!\n");

                }

**else** **if** (n = 0)

                {

                    printf("发送失败!\0");

                }

**else** **if** (n != SOCKET\_ERROR)

                {

                    printf("发送成功!\n");

                }

            }

        }

        Sleep(1000);

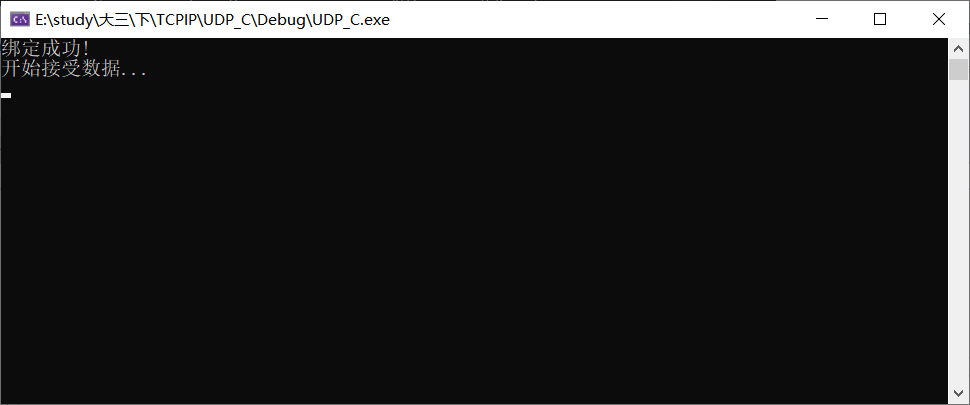
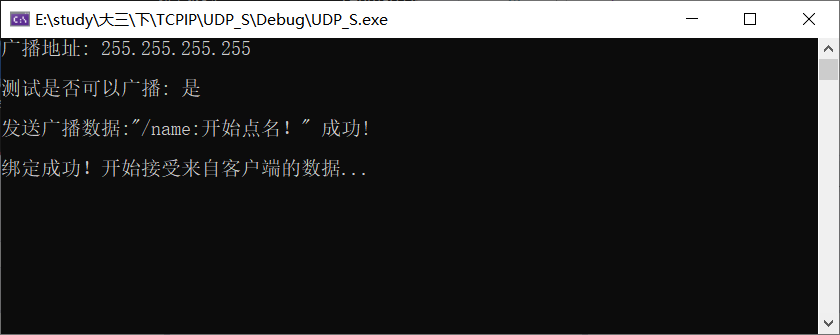
    }

    getchar();

    closesocket(sock);

    WSACleanup();

}



**五、实验总结**

通过本次实验，我掌握了基于UDP广播服务器端客户端收发消息的机制，掌握了UDP协议原理与广播机制。UDP 为用户数据包协议，它的头为8byte数据；分别为16bit源端口号，16bit目的端口号，16bit长度，16bit校验和；它为无连接，主要用于视频传输相关的应用； 它不需像TCP 那样，在发送数据前需建立连接。