哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

计算机网络之网尽其用

主讲人：李全龙

本讲主题

Socket编程-客户端软件设计

解析服务器IP地址

- 客户端可能使用域名（如:study.163.com）或IP地址（如：123.58.180.121）标识服务器
- IP协议需要使用32位二进制IP地址
- 需要将域名或IP地址转换为32位IP地址
  - 函数*inet_addr( )* 实现点分十进制IP地址到32位IP地址转换
  - 函数*gethostbyname( )* 实现域名到32位IP地址转换
    - 返回一个指向结构*hostent* 的指针

```
struct hostent {
    char FAR*       h_name;         /*official host name    */
    char FAR* FAR*  h_aliases;      /*other aliases         */
    short           h_addrtype;     /*address type          */
    short           h_lengty;       /*address length */
    char FAR* FAR*  h_addr_list;    /*list of address   */
};
#define h_addr h_addr_list[0]
```

解析服务器（熟知）端口号

- 客户端还可能使用服务名（如HTTP）标识服务器端口
- 需要将服务名转换为熟知端口号
  - 函数*getservbyname( )*
    - 返回一个指向结构*servent*的指针

```
struct servent {
    char FAR*       s_name;     /*official service name  */
    char FAR* FAR*  s_aliases;  /*other aliases          */
    short           s_port;     /*port for this service  */
    char FAR*       s_proto;    /*protocol to use        */
};
```

解析协议号

- 客户端可能使用协议名（如:TCP）指定协议
- 需要将协议名转换为协议号（如：6）
  - 函数*getprotobyname ( )* 实现协议名到协议号的转换
    - 返回一个指向结构*protoent*的指针

```
struct protoent {
    char FAR*       p_name;      /*official protocol name */
    char FAR* FAR*  p_aliases;   /*list of aliases  allowed*/
    short           p_proto;     /*official protocol number*/
    };
```

TCP客户端软件流程

1. 确定服务器IP地址与端口号
2. 创建套接字
3. 分配本地端点地址（IP地址+端口号）
4. 连接服务器（套接字）
5. 遵循应用层协议进行通信
6. 关闭/释放连接

UDP客户端软件流程

1. 确定服务器IP地址与端口号
2. 创建套接字
3. 分配本地端点地址（IP地址+端口号）
4. 指定服务器端点地址，构造UDP数据报
5. 遵循应用层协议进行通信
6. 关闭/释放套接字

客户端软件的实现- connectsock()

- 设计一个*connectsock*过程封装底层代码

```
/* consock.cpp - connectsock */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <winsock.h>

#ifndef INADDR_NONE
#define INADDR_NONE 0xffffffff
#endif  /* INADDR_NONE */

void    errexit(const char *, ...);
/*------------------------------------------------------------------------
 * connectsock - allocate & connect a socket using TCP or UDP
 *------------------------------------------------------------------------
 */
```

客户端软件的实现- connectsock()

```
SOCKET connectsock(const char *host, const char *service, const char
                   *transport )
{
    struct hostent  *phe;  /* pointer to host information entry     */
    struct servent  *pse;  /* pointer to service information entry  */
    struct protoent *ppe;  /* pointer to protocol information entry */
    struct sockaddr_in sin;/* an Internet endpoint address          */
    int   s, type;         /* socket descriptor and socket type     */


    memset(&sin, 0, sizeof(sin));
    sin.sin_family = AF_INET;
```

指向与哪一个服务器通讯,
服务器的端点地址
服务器对应的服务名
传输层协议

客户端软件的实现- connectsock()

```
/* Map service name to port number */
    if ( pse = getservbyname(service, transport) )
            sin.sin_port = pse->s_port;
    else if ( (sin.sin_port = htons((u_short)atoi(service))) == 0 )
            errexit("can't get \"%s\" service entry\n", service);

     /* Map host name to IP address, allowing for dotted decimal */
    if ( phe = gethostbyname(host) )
            memcpy(&sin.sin_addr, phe->h_addr, phe->h_length);
    else if ( (sin.sin_addr.s_addr = inet_addr(host))==INADDR_NONE)
            errexit("can't get \"%s\" host entry\n", host);

     /* Map protocol name to protocol number */
    if ( (ppe = getprotobyname(transport)) == 0)
            errexit("can't get \"%s\" protocol entry\n", transport);
```

服务名到服务号的解析

转化成网络字节顺序

域名到IP地址的解析

将点分十进制转化成32位的IP地址

协议名到协议号的解析

客户端软件的实现- connectsock()

```
/* Use protocol to choose a socket type */
    if (strcmp(transport, "udp") == 0)
          type = SOCK_DGRAM;
    else                    根据用户输入的传输类型的不同，创建不同类型的套接字
          type = SOCK_STREAM;
     /* Allocate a socket */
    s = socket(PF_INET, type, ppe->p_proto);   创建套接字
    if (s == INVALID_SOCKET)
          errexit("can't create socket: %d\n", GetLastError());
     /* Connect the socket */
    if (connect(s, (struct sockaddr *)&sin, sizeof(sin))==SOCKET_ERROR)
          errexit("can't connect to %s.%s: %d\n", host, service,
                GetLastError());
    return s;
}
```

客户端软件的实现-UDP客户端

- 设计*connectUDP*过程用于创建连接模式客户端UDP套接字

```
/* conUDP.cpp - connectUDP */
#include <winsock.h>
SOCKET connectsock(const char *, const char *, const char *);
/*------------------------------------------------------------
  * connectUDP - connect to a specified UDP service
  * on a specified host
  *------------------------------------------------------------
  */
SOCKET connectUDP(const char *host, const char *service )
{
    return connectsock(host, service, "udp");
}
```

客户端软件的实现-TCP客户端

- 设计*connectTCP*过程，用于创建客户端TCP套接字

```
/* conTCP.cpp - connectTCP */
#include <winsock.h>
SOCKET connectsock(const char *, const char *, const char *);
/*------------------------------------------------------------
  * connectTCP - connect to a specified TCP service
  * on a specified host
  *------------------------------------------------------------
  */
SOCKET connectTCP(const char *host, const char *service )
{
    return connectsock( host, service, "tcp");
}
```

客户端软件的实现-异常处理

```
/* errexit.cpp - errexit */
#include <stdarg.h>
#include <stdio.h>
#include <stdlib.h>
#include <winsock.h>
/*------------------------------------------------------------
 * errexit - print an error message and exit
 *------------------------------------------------------------
 */
/*VARARGS1*/
void errexit(const char *format, ...)
{   va_list args;
    va_start(args, format);
    vfprintf(stderr, format, args);
    va_end(args);
    WSACleanup();
    exit(1);}
```

例1：访问DAYTIME服务的客户端（TCP）

- DAYTIME服务
  - 获取日期和时间
  - 双协议服务（TCP、 UDP），端口号13
  - TCP版利用TCP连接请求触发服务
  - UDP版需要客户端发送一个请求

例1：访问DAYTIME服务的客户端（TCP）

```
/* TCPdtc.cpp - main, TCPdaytime */
#include <stdlib.h>
#include <stdio.h>
#include <winsock.h>

void      TCPdaytime(const char *, const char *);
void      errexit(const char *, ...);
SOCKET connectTCP(const char *, const char *);

#define LINELEN 128
#define WSVERS MAKEWORD(2, 0)
/*------------------------------------------------------------
 * main - TCP client for DAYTIME service
 *------------------------------------------------------------
 */
```

例1：访问DAYTIME服务的客户端（TCP）

```
int main(int argc, char *argv[])
{
    char *host = "localhost";      /* host to use if none supplied  */
    char *service = "daytime";     /* default service port          */
    WSADATA wsadata;
    switch (argc) {
    case 1:
          host = "localhost";
          break;
    case 3:
          service = argv[2];
          /* FALL THROUGH */
    case 2:
          host = argv[1];
          break;
```

例1：访问DAYTIME服务的客户端（TCP）

```
default:
        fprintf(stderr, "usage: TCPdaytime [host [port]]\n");
        exit(1);
    }
if (WSAStartup(WSVERS, &wsadata) != 0)
        errexit("WSAStartup failed\n");
    TCPdaytime(host, service);
    WSACleanup();
    return 0;     /* exit */
}

/*------------------------------------------------------
 * TCPdaytime - invoke Daytime on specified host and print results
 *------------------------------------------------------
 */
```

例1：访问DAYTIME服务的客户端（TCP）

```
void TCPdaytime(const char *host, const char *service)
{
    char buf[LINELEN+1];        /* buffer for one line of text   */
    SOCKET s;                   /* socket descriptor             */
    int cc;                     /* recv character count          */
    s = connectTCP(host, service);

    cc = recv(s, buf, LINELEN, 0);
    while( cc != SOCKET_ERROR && cc > 0)
    {
          buf[cc] = '\0';       /* ensure null-termination       */
          (void) fputs(buf, stdout);
         cc = recv(s, buf, LINELEN, 0);
    }
    closesocket(s);
}
```

例2：访问DAYTIME服务的客户端（UDP）

```
/* UDPdtc.cpp - main, UDPdaytime */
#include <stdlib.h>
#include <stdio.h>
#include <winsock.h>

void      UDPdaytime(const char *, const char *);
void      errexit(const char *, ...);
SOCKET connectUDP(const char *, const char *);

#define LINELEN 128
#define WSVERS MAKEWORD(2, 0)
#define MSG  "what daytime is it?\n"

/*------------------------------------------------------------------------
 * main - UDP client for DAYTIME service
 *------------------------------------------------------------------------
 */
```

例2：访问DAYTIME服务的客户端（UDP）

```
int main(int argc, char *argv[])
{
    char *host = "localhost";      /* host to use if none supplied  */
    char *service = "daytime";     /* default service port          */
    WSADATA wsadata;
    switch (argc) {
    case 1:
          host = "localhost";
          break;
    case 3:
          service = argv[2];
          /* FALL THROUGH */
    case 2:
          host = argv[1];
          break;
```

例2：访问DAYTIME服务的客户端（UDP）

```
default:
        fprintf(stderr, "usage: UDPdaytime [host [port]]\n");
        exit(1);
    }
if (WSAStartup(WSVERS, &wsadata) != 0)
        errexit("WSAStartup failed\n");
    UDPdaytime(host, service);
    WSACleanup();
    return 0;     /* exit */
}

/*------------------------------------------------
 * UDPdaytime - invoke Daytime on specified host and print results
 *------------------------------------------------
 */
```

例2：访问DAYTIME服务的客户端（UDP）

```
void UDPdaytime(const char *host, const char *service)
{
    char   buf[LINELEN+1];    /* buffer for one line of text */
    SOCKET  s;                /* socket descriptor           */
    int  n;                   /* recv character count        */

    s = connectUDP(host, service);
    (void) send(s, MSG, strlen(MSG), 0);

    /* Read the daytime */
    n = recv(s, buf, LINELEN, 0);
    if (n == SOCKET_ERROR)
        errexit("recv failed: recv() error %d\n", GetLastError());
    else
    {
        buf[cc] = '\0';       /* ensure null-termination     */
        (void) fputs(buf, stdout);
    }
    closesocket(s);
    return 0;                 /* exit                         */
}
```

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

谢 谢！