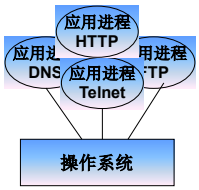
实验内容

- 1. 网络进程间通信
- 2. Socket编程接口(API)
  - 2.1 基于UDP协议通信编程
  - 2.2 基于TCP协议通信编程

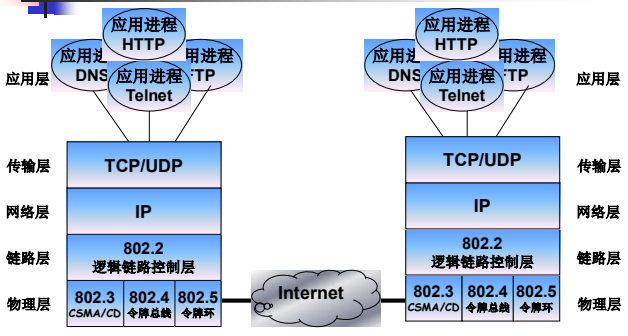
教材实验内容二:
（P259，P162-194）

1. 网络间进程通信



- 进程标识的问题
  - 在同一台主机上，不同进程可用进程号（Process ID）来标识。

1. 网络间进程通信



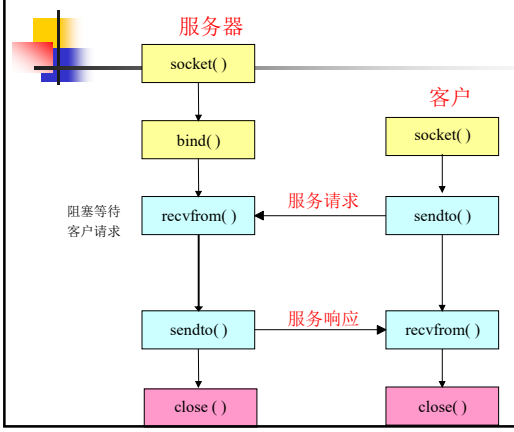
1. 网络间进程通信

- 网络上进程标识的问题
  - 进程ID本地有意义
  - 进程ID是动态的
- 网络上进程通信的解决方法
  - 唯一标识一个进程需要用一个二元组：
    （主机IP地址，端口号）⇨ 套接字（socket）
- 网络上两个进程之间进行通信需要用一个五元组来标识：
  - 进程间采用什么协议进行通信。
  - （本地主机IP地址，本地端口号，协议号，远程主机IP地址，远程端口号）
  - 获知五元组后才能通信

2. Socket编程接口

- 为了便于网络进程间使用TCP/UDP进行通信，传输层向应用层提供了一套编程接口—套接字（Socket）编程接口。
- 套接字编程接口包括以下主要API：
  - 创建socket：socket( )
  - 绑定本地地址：bind( ) – 主要用于服务器端
  - 建立连接：connect( )
  - 接收连接请求：listen( ) 、accept( )
  - 发送数据：send( ) , sendto( )
  - 接收数据：recv( ) , recvfrom( )

2.1 C/S模型流程图(UDP)



（1） 创建套接字—socket( )

- 创建套接字socket，其调用格式如下：
  - **sockid=socket（af，type，protocol）**
  - af(Address Family)—网络地址类型，一般为AF_INET，表示在Internet中使用；
  - type—传输层通信协议类型，SOCK_STREAM表示面向连接的字节流通信方式，SOCK_DGRAM表示无连接的数据报通信方式；
  - protocol— 网络通信协议指定为IPPROTO_IP ；
  - 返回值sockid是一个整数(句柄)，即socket号；
  - 双方都知道五元组信息，双方就可以发送和接收数据了。
  - 一般情况下：客户进程首先知道五元组信息；在TCP协议下，只有三次握手后，服务器进程才知道五元组信息；在UDP协议下，只有服务器进程收到客户进程服务请求后，才知道五元组信息。
- 用法：

```
SOCKET sockid = socket(AF_INET,SOCK_DGRAM,IPPROTO_IP);
if(sockid==INVALID_SOCKET)
{
   //错误处理
}
```

（2）绑定本地地址bind( )

- bind( ) – 用于UDP/TCP服务器端
  - 含义：将服务器进程标识（IP地址+端口号）与所创建的socket绑定，bind( )的调用格式为：
    bind（sockid, Servaddr, addrlen ）
  - sockid，已获得的socket号。
  - Servaddr，本地socketaddr_in地址结构变量：包括本地主机IP地址+端口号。
  - addrlen，表示以字节为单位本地socket地址结构的长度。
  - 用法：

```
sockaddr_in Servaddr;
Servaddr. sin_family=AF_INET;
Servaddr. sin_port= htons(5050); //保证字节顺序
Servaddr. sin_addr.s_addr= inet_addr("20.22.68.5")
int nResult=bind(socketid, (sockaddr*) &Servaddr, sizeof(Servaddr));
if(nResult==SOCKET_ERROR)
 {
    //错误处理
 }
```

（3）发送/接收数据

- 采用UDP协议进行数据发送调用必须明确指定接收方socket地址：
  - sendto（sockid, buf, buflen, flags, destadd, addrlen）
  - 返回值为发送数据字节数

目的IP+目的PORT

- 采用UDP协议数据接收
  - recvfrom（sockid, buf, buflen, flags, souradd, addrlen）
  - 返回值为接收数据字节数

此时，获得源IP+源PORT

（4）关闭套接字

- 释放所占有的资源，服务器端和客户端使用。
- int closesocket(SOCKET socketid) // socketid为欲关闭的套接字；
- 用法：

```
int nResult=closesocket(socketid);
if(nResult==SOCKET_ERROR)
{
        //错误处理
}
```

助教检查点（UDP）

- 在客户端，从键盘输入任意字符串发送给服务器，服务器收到后，在屏幕上显示该字符串，并将客户端IP地址和端口号显示在屏幕上；
- 服务器将客户端发送来的字符串发送给客户端，客户端显示该字符串、服务器IP地址及其端口号。

助教记录分数

- 两个同学为一组；
- 当完成一个检查点时，主动要求助教检查，助教对检查间完成情况进行记录，并记录好完成时间。

C/S模型流程图（TCP）

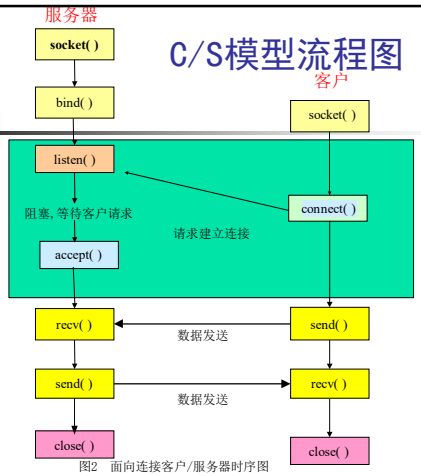


图2 面向连接客户/服务器时序图

服务器端的侦听、接收连接请求—listen( )和accept( )

- 采用TCP协议的服务器进程一般在某个周知的端口上等待客户进程的连接请求（connect()）。
- 服务器进程平时处于侦听状态*listen( )*，一旦有连接请求来到时，服务器进程被唤醒并处理客户进程的连接请求。
- 服务器进程通过*accept( )*系统调用来接收并处理客户进程的连接建立请求。
  - listen( )：将服务器进程设置为侦听(服务等待)状态；
  - accept( )：服务器进程受理客户进程的连接请求。

listen函数调用

- 如果有客户端连接请求，则把请求放入到等待队列中排队等待处理，其调用格式为：
  - listen（sockid, quelen）
  - Sockid：本地socket号，服务器进程在此socket地址上接收连接请求。
  - Quelen：服务端连接请求队列长度；此参数限制连接请求的排队长度，通常允许的连接请求排队长度最大值。
- 用法：

```
int nResult=listen(sockid,5) //最多5个连接
if(nResult==SOCKET_ERROR)
{  //错误处理 }
```

accept函数调用

- accept( )：服务器进程处理客户进程的连接请求；
- 当等待队列没有连接请求时，服务进程处于侦听状态，否则，响应并处理连接请求，其调用格式如下：
  - newsock = accept (sockid, clientaddr, addrlen)
  - Sockid：本地socket号。
  - clientaddr，指向客户端socket地址结构的指针，初始值为空，当accept调用返回后，客户进程的socket地址被填入该地址结构中.
  - addrlen，初值为0，当accept调用返回后存放客户socket地址长度。
  - 用法：

```
sockaddr_in clientaddr;
SOCKET newsockid =
accept(sockid,(sockaddr*)&clientaddr,sizeof(sockaddr))
if(newsockid==INVALID_SOCKET)
{ //错误处理 }
```

客户端建立连接请求 connect函数

- 采用TCP协议通信的客户进程通过调用connect( ) 主动请求与服务器建立连接；
- connect() 的调用格式为：
  - connect (sockid, Seraddr, Serveraddrlen)
  - Sockid：本地socket号。
  - Seraddr：是一个指向服务器的socket地址结构（包括IP地址、端口等信息）的指针。
  - Seraddrlen：服务器socket地址结构长度。

建立连接请求connect函数用法

- 用法：用于TCP客户端。

```
sockaddr_in Seraddr;
Seraddr. sin_family = AF_INET;
Seraddr. sin_port=htons(5050); //保证字节顺序
Seraddr. sin_addr.s_addr= inet_addr("20.22.68.5") //保证字节顺序
int nResult=connect(sockid,(sockaddr*)&Seraddr,sizeof(sockaddr));
if(nResult==SOCKET_ERROR)
{
      //错误处理
}
```

与UDP发送数据差异

- 采用TCP协议进行数据发送，任意一方可以向对方发送数据；不必指定对方地址，因为sockid中已包含通信的5元组：
  - send (sockid, buf, buflen, flags)
  - 返回值为发送数据字节数。

紧急数据可在这里设定.

与UDP接收数据差异

- 接收数据调用与发送数据调用是一一对应的。
- 采用TCP协议数据接收，不必指定对方地址
  - recv (sockid, buf, buflen, flags)
  - 返回值为接收数据字符数。

助教检查点（UDP）

- 在客户端，从键盘输入任意字符串发送给服务器，服务器收到后，在屏幕上显示该字符串，并将客户端IP地址和端口号显示在屏幕上；
- 服务器将客户端发送来的字符串发送给客户端，客户端显示该字符串、服务器IP地址及其端口号。