1 TCP\IP的四层模型，每一层指出两个协议。

链路层：以太网、令牌环网

网络层：IP、ICMP

传输层：TCP、UDP

应用层：HTTP、FTP

2 标识一次连接的基本要素有哪些

IP地址、端口号

3 TCP socket创建的代码片段（C）

SOCEKT sockfd=socket(AF\_INET,SOCK\_STREAM,0);

if(sockfd==INVALID\_SOCKET){

printf(“socket error: %d\n”,WSAGetLastError());

}

4 UDP socket创建的代码片段（C）

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

if( udps == INVALID\_SOCKET ){

//出错处理！

}

5 TCP的通讯模型

①单线程

②多线程

6 UDP的通讯模型

7 服务器两种bind含义及区别，bind的代码片段

①代码片段：

struct sockaddr\_in myaddr;

memset(&myaddr,0,sizeof(myaddr));

myaddr.sin\_family=AF\_INET;

myaddr.sin\_port=htons(PORT);

myaddr.sin\_addr.s\_addr=htonl(INADDR\_ANY);

if(bind(sockfd,(struct sockaddr \*)&myaddr,sizeof(myaddr))==

SOCKET\_ERROR){

printf("Bind to port %d error\n"，WSAGetLastError())；

closesocket(sockfd);

………

}

②两种bind含义及区别

(1)服务器指定套接字的公认端口号，不指定IP地址。

服务器调用函数bind时，如果设置套接字的IP地址为特殊的INADDR\_ANY，表示它愿意接收来自任何网络设备接口的客户机连接。这是服务器最经常使用的绑定方式。发送时，源IP是默认输出端口网卡的IP。这是多网卡的服务器上常用的方式。

(2)服务器指定套接字的公认端口号和IP地址。

服务器调用函数bind时，如果设置套接字的IP地址为某个本地IP地址，这表示服务器只接收来自对应于这个IP地址的特定网络设备接口的客户机连接。如果这台机器只有一个网络设备接口，这和第一种情况是没有区别的，但当这台机器有多个网络设备接口时，我们可以用这种方式来限制服务器的接收范围。

8 listen的功能及第二个参数的含义

功能：将一个已绑定的套接字转换为倾听套接字。

第二个参数的函数：TCP协议将到达的连接请求排队，函数listen的第二个参数指定这个队列的最大长度。 要创建一个倾听套接字，必须首先调用函数socket创建一个主动套接字，然后调用函数bind将它与服务器套接宇地址绑定在一起，最后调用函数listen进行转换。这3步操作是所有TCP服务器所必须的操作。若该请求队列已满，但又有新的请求到达时，则新的请求将被拒绝。

9 accept的功能，accept返回的socket与参数中的socket有何区别

功能：从倾听套接字的已完成连接队列中接收—个连接。

区别：accept函数的参数中的socket是等待连接的倾听套接字描述符，而函数返回值为一个新的套接字描述符，标识这个接收的连接。这两个套接字的作用是完全不同的：一个服务器进程通常只需创建一个倾听套接字，在服务器进程的整个活动期间，用它来接收所有客户机的连接请求，在服务器进程终止前关闭这个倾听套接字；而对于每个接收的连接，TCP协议都创建一个新的连接套接字，来标识这个连接，服务器使用这个连接套接字与客户机进行通信操作，当服务器处理完这个客户机请求时，关闭这个连接套接字。

10 连接型的UDP解决什么问题，用于什么场合，代码片段

代码片段：

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

sockaddr\_in sip;

memset(&sip,0,sizeof(sip));

sip.sin\_family=AF\_INET;

sip.sin\_port=htons(9999);

sip.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

connect(s,(sockaddr\*)&sip,sizeof(sip));

用途：

客户端要求内核进行UDP包的过滤

11 结构化数据的传输方案（有汉字的字符串、有彩色照片、二叉树）

有汉字的字符串：

①统一按UTF-8进行编码（字节数组）

②初始4字节用于表示字节数据的长度，之后为字节数据本身

有彩色的照片的身份证照片：分包？

二叉树：JSON

12 Win32多线程代码

函数申明：

unsigned int CALLBACK MyThread(void \* param)

{

//DWORD n=(DWORD)param;

for (int i=1;i<11;i++)

{

printf("线程:%d 打印：%d \n",GetCurrentThreadId(),i);

Sleep(2000);

}

return 1;

}

Main函数中调用：

u\_int ID=0;

HANDLE hThread = (HANDLE) \_beginthreadex(

NULL,0,MyThread,NULL,0,&ID);

if (hThread1!=0 )

{

printf("线程ID:%d handle:%d启动。\n",ID,hThread);

}

WaitForSingleObject(hThread,INFINITE);

CloseHandle(hThread1);

13 UDP的数据包的最大长度

UDP数据包的最大长度=MTU-20（IP数据包头部的最小长度）-8（UDP数据包头长）

14 IP最多可以记录几个路由

IP数据包最大头长60个字节，头部固定长度为20个字节，选项码、选项长度和指针各占一个字节，这样只剩下37（60-20-3）个字节来存放路由器IP地址，每个IP地址需要4个字节，所以最多可以记录9个路由地址。

15 域名解析

BufferedReader bReader = new BufferedReader(new InputStreamReader

(System.in));

String name=bReader.readLine();

InetAddress address=InetAddress.getByName(name);

System.out.println(address.getHostAddress());

16 基于TCP的网络通讯程序

服务器：

ServerSocket ss=new ServerSocket(9999);

Socket cs=ss.accept();

BufferedReader bReader = new BufferedReader(new InputStreamReader(

cs.getInputStream()));

PrintWriter pWriter=new PrintWriter(cs.getOutputStream());

String line=null;

while((line=bReader.readLine())!=null){

pWriter.println("串长："+line.length());

pWriter.flush();

if(line.equalsIgnoreCase("quit")){

break;

}

}

cs.close();

ss.close();

pWriter.close();

bReader.close();

客户端：

Socket cs=new Socket(InetAddress.getByName("127.0.0.1"), 9999);

BufferedReader bReader=new BufferedReader(new InputStreamReader(

cs.getInputStream()));

PrintWriter pWriter=new PrintWriter(cs.getOutputStream());

String line=null;

BufferedReader kbrReader=new BufferedReader(new InputStreamReader(

System.in));

do {

line=kbrReader.readLine();

pWriter.println(line);

pWriter.flush();

String res=bReader.readLine();

System.out.println(res);

} while (!"quit".equalsIgnoreCase(line));

pWriter.close();

bReader.close();

cs.close();

17 简单远程遥控程序

18 基于UDP的网络通讯程序

服务器：

DatagramSocket ds = new DatagramSocket(9999);

byte[] rBuf = new byte[64 \* 1024];

DatagramPacket rdp = new DatagramPacket(rBuf, 64\*1024);

boolean again = true;

while (again) {

ds.receive(rdp);

String s = new String(rdp.getData(), rdp.getOffset(), rdp.getLength());

System.out.println("客户机IP:" + rdp.getAddress() + " 端口号:" + rdp.getPort() + " 发送内容:" + s);

String res = "串长" + s.length();

DatagramPacket sdp = new DatagramPacket(res.getBytes(), res.getBytes().length, rdp.getAddress(),rdp.getPort());

ds.send(sdp);

}

ds.close();

客户端：

DatagramSocket ds = new DatagramSocket();

Scanner sc = new Scanner(System.in);

byte[] rBuf = new byte[64 \* 1024];

DatagramPacket rdp = new DatagramPacket(rBuf, 64 \* 1024);

String line = null;

do {

line = sc.nextLine();

DatagramPacket sdp = new DatagramPacket(line.getBytes(), 0, line.getBytes().length,InetAddress.getByName("127.0.0.1"), 9999);

ds.send(sdp);

ds.receive(rdp);

String res = new String(rdp.getData(), rdp.getOffset(), rdp.getLength());

System.out.println("客户端响应:" + res);

} while (!"quit".equalsIgnoreCase(line));

ds.close();

sc.close();

19 广播程序

同UDP网络通讯程序，其中客户端地址为255.255.255.255

20 组播程序vvvvvv

服务器：

MulticastSocket ms = new MulticastSocket(9999);

ms.joinGroup(InetAddress.getByName("226.6.6.6"));

byte[] rBuf = new byte[64 \* 1024];

DatagramPacket rdp = new DatagramPacket(rBuf, rBuf.length);

boolean again = true;

while (again) {

ms.receive(rdp);

String s = new String(rdp.getData(), rdp.getOffset(), rdp.getLength());

System.out.println("从组中接收消息:" + s);

}

ms.leaveGroup(InetAddress.getByName("226.6.6.6"));

ms.close();

客户端：

Scanner sc = new Scanner(System.in);

String line = null;

MulticastSocket ms = new MulticastSocket();

do {

line = sc.nextLine();

DatagramPacket sdp = new DatagramPacket(line.getBytes(), 0, line.getBytes().length,InetAddress.getByName("226.6.6.6"), 9999);

ms.send(sdp);

} while (!"quit".equals(line));

ms.close();

sc.close();

21 画图①IP分片填图、②记录路由、③严格源路由

22 ICMP中的路径MTU发现算法

①构造一个TCP报文，填一个不常用的端口号

②构造一个IP报文，长度设为65535且不可分片，返回一个34 ICMP报文

③减小IP报文的长度重复上述操作，直到获得一个33 报文

23 MyPing程序

①构造一个ICMP的原始套接字

②构造一个ICMP 80报文，数据填入当前时间

③发送并接收报文，若为00，则取出数据部分的时间

④现在时间-00报文中的时间=ping传输的时间

24 arp的请求过程（请求报文、应答报文格式见书）

①广播方式发送请求报文

②单播方式发送应答报文

25 TCP协议中MSS、窗口大小及TCP报文大小三者之间的关系

MSS是TCP数据包每次能够传输的最大数据分段大小，不包括TCP首部长度。而TCP报文大小指TCP首部大小加上数据段大小。窗口大小是接收端的流量控制措施，用来告诉另一端它的数据接收能力。

26 TCP协议四个定时器及其作用

重传定时器：重传定时器是用来计算TCP报文段的超时重传时间的。每发送一个报文段就会启动重传定时器，假设在定时器时间到后还没收到对该报文段的确认，就重传该报文段，并将重传定时器复位，又一次计算；假设在规定时间内收到了对该报文段的确认，则撤销该报文段的重传定时器。

坚持定时器：主要解决零窗口大小通知可能导致的死锁问题。

保活计时器：主要是为了防止两个TCP连接出现长时间的空闲。当客户端与服务器端建立TCP连接后，很长时间内客户端都没有向服务器端发送数据，此时很有可能是客户端出现故障，而服务器端会一直处于等待状态。保活计时器就是解决这种问题而生的。

2MSL计时器：主要是为了确保发送的最后一个ACK报文段能够到达对方，并防止之前与本连接有关的因为延迟等原因而导致已失效的报文被误判为有效。

27 HTTP有状态实现四种方式

url重写、隐藏域、cookie、session

28 设置HTTP头部，浏览器自动打开word、excel

打开word：设置头部content-type:application/msword

打开excel：设置头部content-type:application/x-msexcel

29 基于HTTP的多线程下载文件步骤

①创建线程，用Header请求头部，获取总长

②每个线程下载部分k=总长/n

③启用n个线程，用Range：bytes=k\*(i-1)-k\*i设置请求头

30 FTP断点续存原理

使用REST <offset>命令由特定偏移量重启文件传输，再使用RETR <filename>从服务器上复制文件。

31 FTP主动方式、被动方式及使用场合

PASV命令服务器被动方式打开数据连接，用于客户端存在防火墙之后。

PORT <address>客户端将它的IP和port传送给服务器，服务器主动和客户端发起TCP连接，用于客户端无防火墙。