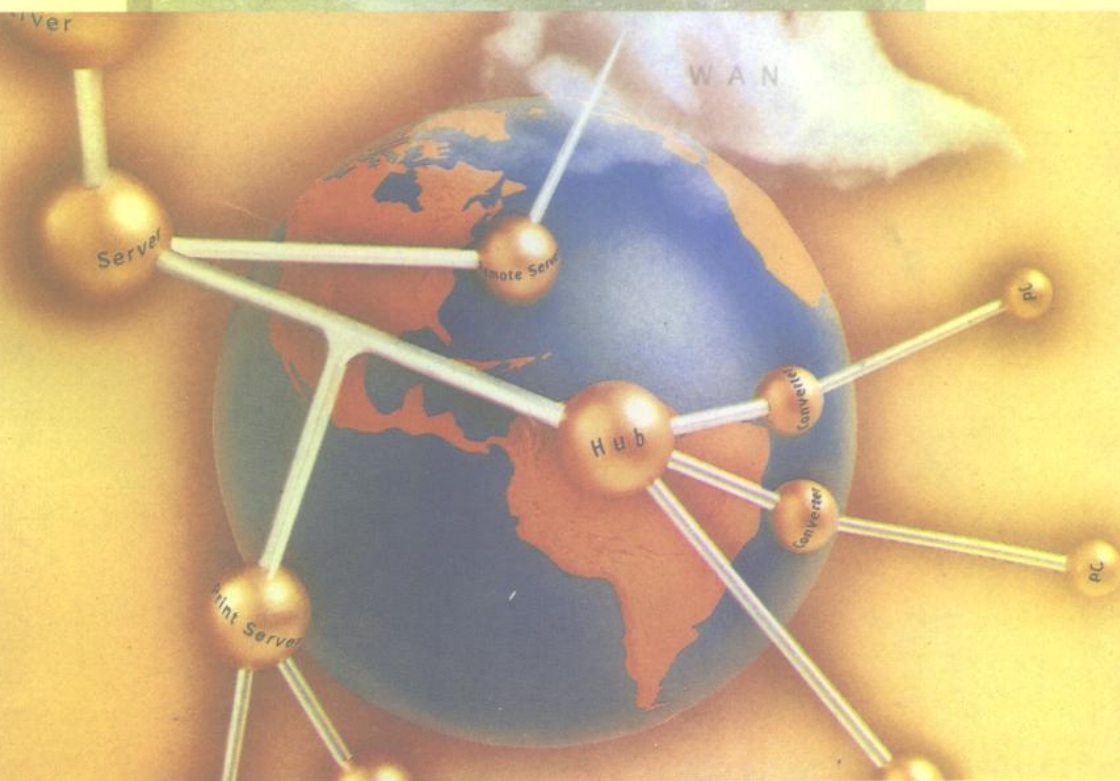


# Windows 界面下的 网络编程

张宝社 张宝峰 王艳辉 编著



中国科学技术大学出版社

TP393.4

Z10

497030

# Windows 界面下的网络编程

张宝社 张宝峰 王艳辉 编著

中国科学技术大学出版社

1997·合肥

## 内 容 简 介

本书以简单明了的语言和形象生动的例子,介绍了 WINSOCK (Windows Socket) 网络编程,共分上、下两篇。上篇主要介绍了网络编程的基础知识,其中包括网络通信模型、TCP/IP 协议、socket 编程界面、WINSOCK 的消息结构等。下篇主要介绍了五个网络协议和相应的网络程序。这些网络协议包括文件传输协议(FTP)、邮件发送接收协议(SMTP 和 POP3)、超文本传输协议(HTTP)、Gopher 协议、CHECKER 协议(作者自定义的)。在附录中,详细介绍了 WINSOCK 的函数集和超文本制作语言(HTML)。

本书论述通俗、技术实用,适合于计算机网络、计算机软件以及计算机应用等专业的大专院校师生和计算机软件开发人员使用与参考。

### 图书在版编目(CIP)数据

Windows 界面下的网络编程/张宝社 张宝峰 王艳辉 编著。

—合肥:中国科学技术大学出版社,1997年4月

ISBN 7-312-00887-9

- I Windows 界面下的网络编程
- II 张宝社 张宝峰 王艳辉
- III ①WINSOCK (Windows Socket) ②网络编程
- IV TP

中国科学技术大学出版社出版发行  
(安徽省合肥市金寨路 96 号,邮编:230026)  
中国科学技术大学印刷厂印刷  
全国新华书店经销

开本:787×1092/16 印张:13 字数:302 千

1997 年 4 月第 1 版 1997 年 4 月第 1 次印刷

印数:1—5000 册

ISBN 7-312-00887-9/TP·174 定价:16.00 元

# 前 言

Internet, 国际互联网络, 几年前对中国人还是一个陌生的名词, 但现在越来越多的中国人成为它的用户, 特别是在中国教育科研网(CERNET)开通之后。Internet 是目前国际上规模最大的计算机网间网, 它的普及和对世界的冲击, 是它的设计者最初所料所未及的。现代社会是一个高速发展的信息时代, Internet 的出现是我们这个信息时代的一个里程碑。甚至有人认为它就是人们所设想的未来信息高速公路的雏型。

Internet 是在 70 年代中期美国国防部的 ARPANET 广域网络的基础上建立和发展起来的。最初是为美国的大学和科研机构的学术讨论和交流提供服务的, 它继承了 ARPANET 网的网络体系结构和协议标准(TCP/IP 协议)。

现在, Internet 已波及各个行业, 它的触觉已深向世界各个地区。今天, 您只要有一台个人电脑, 并和 Internet 联网, 就可以给远在万里之外的朋友或生意上的合伙人或学术上的合作者发送电子邮件, 浏览公用的网络上的电子出版物, 并从世界其它的地方获取电子资源。本书的主要参考资料就是作者从 Internet 网上获取得到的。您还可以和朋友不用面对面地聊天, 在网上发布广告、购物, 甚至在家里办公。

最初的网络软件是以 UNIX 操作系统为软件开发环境的。UNIX+TCP/IP 在过去十几年里一直是计算机网络软件的“既成事实”的开发系统平台。直到 1992 年初这种情况才发生了变化, 因为出现了 WINSOCK(WINDOWS+TCP/IP)。WINSOCK 的出现标志着 Internet 和 Microsoft Windows 的联姻, 一个计算机软件和硬件的新时代开始了。这也是个人计算机蓬勃发展的必然结果。

在个人计算机上, Microsoft Windows 有三个特点非常适合于作为 Internet 的软件开发平台。

首先, 自 Microsoft Windows 1985 年秋问世以来的十年中, 它已成为使用 Intel 微处理器的个人计算机的工业标准图形环境。Internet 上的资源非常丰富, 资源的表现形式也多种多样, 不仅有纯文本, 还有图形、图像。现在, 越来越多的资源是文本和图像及语音共存的, 用户浏览这些资源就需要一个图形环境。而 Microsoft Windows 提供的与设备无关的图形界面, 不仅满足了用户, 而且对软件开发也提供了完备的图形环境。

其次, 就是 Microsoft Windows 软件的消息结构和多任务环境非常适合于 Internet 的网络信息传输。在 UNIX 界面下, 网络软件的效率和网络的传输速度联系密切, 网络的拥塞情况是决定网络软件效率的最关键因素。在 Windows 界面下, 任何网络的信息传送只作为其多任务中的一个, 将不影响其它任务的操作和完成。也就是说, 当由于网络拥塞, 信息传输速度变慢, 由此所导致的空闲时间 Windows 将分配给其它任务。这样, 在个人计算机的 Windows 界面下, 可同时运行多个网络软件和非网络软件。这一切的实现, 主要是靠 Windows 的消息发布机制来完成的。

最后是 Microsoft Windows 的动态链接。不同的网络软件共享同一个 WINSOCK 动态链接库(WINSOCK.DLL), 这合理地解决了个人计算机内存资源有限的问题。

WINSOCK(WINDOWS+TCP/IP)是1992年初,由Microsoft公司主持、美国主要的软件公司Sun Microsystems、FTP、Microdync、Berkeley BSD等参加,一起讨论、研究和组织开发的。它是以TCP/IP为其协议标准。它不仅继承了UNIX+TCP/IP开发环境中的所有socket函数,并增加了适用于Windows界面的socket函数。也就是说,WINSOCK是UNIX SOCKET的超集。它继续沿用UNIX SOCKET的数据类型。

随着Internet在中国的发展,我国将需要大量Internet网络软件开发和资源设计、维护的专业人才。有人称Internet是当今的淘金路。Bill Gates曾说微软公司的一个决策失误就是没有及时踏上Internet这条淘金之路。捷足先登的美国24岁的软件神童Andersen和他的网景公司(Netscape)正在Internet上大显身手。Internet这条淘金之路已成为人们的视线焦点。朋友,您不可能对这新的淘金路视而不见?假如您愿意踏上这条路,那这本书将是您的上路指南。

本书分上下两篇。上篇为基础篇,包括第一章到第七章,主要介绍Internet的TCP/IP协议、socket网络编程界面、WINSOCK编程界面的基础知识。下篇为应用篇,包括第八章到第十二章,分别介绍了Internet上的FTP协议、SMTP协议、POP3协议、HTTP协议、GOPHER协议以及相应的应用程序。其中第十二章,作者自定义了CHECKER协议。附录分为A、B两部分。附录A介绍了WINSOCK的函数库,附录B介绍了超文本制作语言(HTML)。

本书的所有程序都经过严格调试,可在下面三种环境中编译:

1. 7.0版本Microsoft C/C++ Professional Development System for Windows
2. Borland C/C++ 3.1(或更高版本) for Windows
3. Visual C++(但必须重新改写相应的制作文件,即\*.MAK文件)

在本书完成之际,作者对中国科技大学出版社黄德老师的大力支持表示感谢。同时,对李洪梅小姐在本书的录排工作中表现的耐心和付出的劳动表示感谢。没有他们的支持和工作,这本书不可能这么早与读者见面。

尽管作者殚精竭虑,但错误和不足在所难免,请惠于批评指正。

作者

1996年11月

# 目 录

|         |   |
|---------|---|
| 前言..... | 1 |
|---------|---|

## 上篇 基础篇

|                               |    |
|-------------------------------|----|
| 第一章 网间网通信要解决的问题.....          | 3  |
| 第二章 Internet 的网间网通信模型 .....   | 4  |
| 第三章 TCP/IP 网间网体系结构与协议层次 ..... | 6  |
| 3.1 TCP/IP 网间网体系结构与协议层次 ..... | 6  |
| 3.2 IP 地址 .....               | 7  |
| 3.3 域名系统 .....                | 8  |
| 第四章 socket 编程界面 .....         | 10 |
| 4.1 socket 的基本概念 .....        | 10 |
| 4.2 socket 的实现 .....          | 11 |
| 4.3 socket 的类型 .....          | 13 |
| 第五章 WINSOCK 的消息结构 .....       | 15 |
| 第六章 迟滞函数 .....                | 19 |
| 第七章 WINSOCK 的加载、注销和错误码 .....  | 21 |
| 7.1 WINSOCK 的加载与注销 .....      | 21 |
| 7.2 WINSOCK 的错误码 .....        | 21 |

## 下篇 应用篇

|                              |     |
|------------------------------|-----|
| 第八章 FTP 协议和 FTP 客户程序 .....   | 25  |
| 8.1 FTP 协议 .....             | 25  |
| 8.2 FTP 客户程序 .....           | 33  |
| 第九章 电子邮件协议和客户程序 .....        | 59  |
| 9.1 简单邮件传输协议 SMTP .....      | 59  |
| 9.2 POP3 协议 .....            | 64  |
| 9.3 邮件客户程序 .....             | 70  |
| 第十章 HTTP 协议和 WWW 服务器程序 ..... | 91  |
| 10.1 记号规则和一般语法 .....         | 91  |
| 10.2 HTTP 协议 .....           | 94  |
| 10.3 WWW 服务器程序 .....         | 102 |

|             |                           |     |
|-------------|---------------------------|-----|
| <b>第十一章</b> | <b>GOPHER 协议和客户程序</b>     | 117 |
| 11.1        | Internet Gopher 协议        | 117 |
| 11.2        | Gopher 客户程序               | 119 |
| <b>第十二章</b> | <b>CHECKER 协议和客户服务器程序</b> | 139 |
| 12.1        | CHECKER 协议                | 139 |
| 12.2        | CHECKER 客户服务器程序           | 139 |

## 附 录

|             |                        |     |
|-------------|------------------------|-----|
| <b>附录 A</b> | <b>WINSOCK 的函数集</b>    | 156 |
| A.1         | WINSOCK 的基本函数          | 156 |
| A.2         | WINSOCK 的数据库函数         | 172 |
| A.3         | WINSOCK 的 Windows 扩展函数 | 176 |
| <b>附录 B</b> | <b>超文本制作语言(HTML)</b>   | 191 |
| <b>参考文献</b> |                        | 201 |

# 第一章 网间网通信要解决的问题

Internet 是广域网，在它上的通信是网间网的通信，也是不同主机的进程间的相互通信。

网间网进程通信的首要问题是进程标识。在同一主机，不同进程可以用进程号唯一标识。但在网间网环境中，各主机独立分配的进程号是不能作为进程标识符的。在 Internet 上，通信是在“A 主机上的 5 号进程与 B 主机上的 6 号进程”之间进行的。那么对进程的标识首先要有本地主机的网络地址。在 Internet 上，对进程的进一步标识的概念是端口号。端口号是 TCP 和 UDP 与应用程序打交道的访问点。

网间网进程通信需要解决的第二个问题是多重协议的标识。以 UNIX 操作系统为例，其编程界面支持的协议包括 TCP/IP 的 TCP 和 UDP，XNS(Xerox Network System)的 SPP (Sequential Packet Protocol)和 IDP (Internetwork Datagram Protocol)，ARPANET 的 IMPLINK 以及 UNIX 内部进程的通信协议等等。网间网在进程通信时，必须从如此众多的通信协议中作出选择。

综合以上两点，网间网中全局唯一地标识一个通信进程需要用一个三元组：(协议，本地主机的网络地址，本地主机的进程端口号)。而一个完整的网间网进程通信实例由两个通信进程组成，因此一个完整的网间网进程通信需要一个五元组来标识：(协议，本地主机的网络地址，本地主机的进程端口号，远程主机的网络地址，远程主机的进程端口号)。

网间网的进程通信需要解决的第三个问题是进程间的相互作用模式，即应用程序间的相互作用模式。任何进程通信的最终目的无外乎是相互影响，使对方发生某种变化。相互影响，即相互作用，因此进程通信实质上是进程间的相互作用。进程之间如何发生相互作用呢？最简单的想法是根据一个相关建立全局唯一的进程通信，相互交换信息后，双方方便相互影响，从而达到相互作用的目的。

这三个问题将在以后的章节详细讨论。

## 第二章 Internet 的网络通信模型

在使用 Internet 时,我们经常要和各种服务器(Server)打交道,如域名服务器、时间服务器、WWW 服务器、FTP 服务器、MAIL 服务器等等。域名服务器为用户解析域名,把域名转换为 IP 地址;时间服务器为用户提供计时服务;WWW、FTP 服务器为用户提供其相应的网上资源;MAIL 服务器为用户转发和接受电子邮件。当我们有意或无意中使用这些服务器时,我们就成为其服务对象,就是客户。更准确地说,服务器是提供相应服务的软件或进程,客户是接受服务的软件或进程。这就是 Internet,即使用 TCP/IP 协议标准的网间网,最主要的通信进程间的相互作用模型——客户/服务器模型(Client/Server Model)。

这里要注意,客户和服务是对网络的通信进程而言的,不是针对网络用户或个人计算机。用户可以把个人计算机变为一个服务器,成为网上的服务器。谁为服务器,谁为客户,主要是取决于通信进程中双方的地位,是对具体的进程而言的。在一台个人计算机上,用户可设置不同的通信进程,那么用户可能既是某个进程的客户,又是另一个进程的服务器。

对客户/服务器模型最形象的比喻,就是日本动画片《一休》中经常出现的画面——“提问—回答”。先由一方主动向另一方提出问题,另一方的反应就是回答。

客户就是提问方,服务器就是回答方。在网络上,客户和服务器分别是两个应用程序(进程)。客户向服务器发出服务请求,服务器作出响应,这就是客户/服务器模型中进程相互作用的简单过程,如图 2.1 所示。

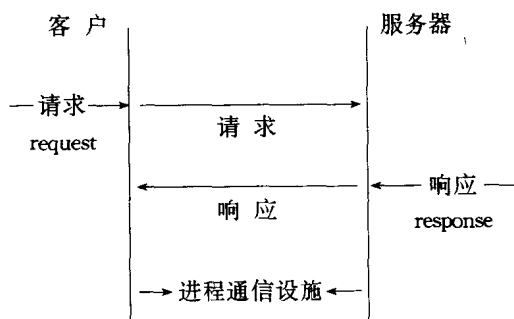


图 2.1 客户/服务器模型中进程相互作用的简单过程

图中“进程通信设施”既可以是单机进程通信设施,也可以是基于 TCP/IP 的网间网进程通信设施。

Internet 中的时间服务、文件服务、地址解析等等应用程序都采用了客户/服务器模型。本书中的五个程序也是建立在此模型之上的。读者通过这五个程序会对客户/服务器模型有深刻的理解。

为什么选择客户/服务器模型?

对这个问题的回答要从客户/服务器模型的特点谈起。客户/服务器模型最重要的特点是非对等相互作用，即客户和服务端处于不平等的地位，服务器拥有客户所不具备的硬件软件资源和运算能力，服务器提供服务，客户请求服务。在网间网中大量客观地存在着资源分布和运算能力不均等的现象。小到一个物理网络，往往是某些主机拥有大量外存，某些主机只有很少外存或没有外存，有些主机拥有打印机，有些没有；大到整个网间网，少数网点拥有超级速算能力，大量的网点由 PC 机构成等等。

同时，网间网还大量存在人为的不均等现象。人为的不均等主要是指信息而言，比如名字信息、地址信息。这些信息往往以数据库形式存在于少数特权主机中，供局部或全局访问。造成人为信息分布不均等的原因在于这些信息一般具有公共性，为节约整个网络的资源计，没有必要在每台机上都维持一个拷贝；对于一些动态变化的信息，假如在每台主机上都维持一份相同的拷贝，那么为保证所有拷贝的一致性，要进行的动态修改和刷新工作，其开销将是非常巨大的。

可见，不均等现象是不可避免的，而客户/服务器模型体现了这种现象并很好地适应了这种现象。

总之，客户/服务器模型起源于网间网中资源、运算能力和信息不均等的现实。这是它成为网间网应用程序相互作用主要模型的第一个原因。

第二个原因是技术性的。网间网进程通信与单个计算机的进程之间的通信不同的一个特点是，网间网的通信完全是异步的 (asynchronous)，谁也不知道谁会在什么时候发起一次进程通信，相互通信的进程之间既不存在父子关系，也不共享内存缓冲区。因此需要一种机制，为准备通信的进程之间建立联系，为二者的数据交换提供同步。客户/服务器模型完美地解决了上述问题。按照该模型，每次通信均由随机启动的客户进程发起，服务器进程从开机就处于等待状态。这样可以保证服务器随时对客户请求作出响应。另外，客户与服务器的请求/应答模式为相互通信的进程之间的数据交换传输的同步提供了有力的支持。

综上所述，客户/服务器模型并不是凭空想象的，它是客观现实与技术实现相结合的产物。

我们知道，通信和资源共享是计算机网络的两大功能，通信是手段，资源共享是真正目的。一旦引入客户/服务器模型，我们才真正感觉到通信手段的意义，以及资源共享目的的存在。从某种意义上，我们可以说，客户/服务器模型是对计算机网络两大功能及其关系的深刻体现和完美表达。

# 第三章 TCP/IP 网间网体系结构与协议层次

在这一章,简单介绍 TCP/IP 的网络分层、IP 地址和域名系统。

## 3.1 TCP/IP 网间网体系结构与协议层次

TCP/IP 协议分为四个层次:应用层,传输层(TCP),网间网层(IP),网络接口层。

应用层是向用户提供一组常用的应用程序,如文件传输访问、电子邮件等。严格说起来,TCP/IP 网间网协议只包含下二层(不含硬件),应用程序不能算 TCP/IP 的一部分。就上面提到的常用应用程序,TCP/IP 制定了相应的协议标准,所以也把它们作为 TCP/IP 的内容。事实上,用户完全可以在网间网上(即传输层上)建立自己的专用应用程序,这些专用应用程序要用到 TCP/IP,但不属于 TCP/IP。

传输层(TCP)提供应用程序间(即端到端)的通信,其功能包括:格式化信息流和提供可靠的传输。为实现后者,传输层协议规定接收端必须发回确认,并且假如分组报文丢失,必须重新发送。传输层(TCP)还要解决不同应用程序的识别问题,因为在一般的通用计算机中,常常是多个应用程序同时访问网间网。为区别各应用程序,传输层在每一分组中增加了识别信源和信宿应用程序的信息。

网间网层(IP)负责相邻计算机之间的通信,其功能有三个方面:

(1)处理来自传输层的分组发送请求,收到请求后,将分组装入 IP 数据报,填充报头,选择去往信宿机的路径,然后将数据报发往适当的网络接口。

(2)处理输入数据报:首先检查其合法性,然后进行寻径。假如该数据报已到达信宿机,则去掉报头,将剩余部分(传输层分组)交给适当的传输协议;假如该数据报尚未到达信宿机,则转发该数据报。

(3)处理 ICMP 报头,处理路径,流程,拥塞等问题。

网络接口层是 TCP/IP 软件的最低层,负责接收 IP 数据报并通过网络发送之,或者从网络上接收物理帧,抽出 IP 数据报,交给 IP 层。

TCP/IP 协议分层模型如图 3.1 所示。

协议软件分层的好处在于简化软件设计。首先在分层模型中,只需要与硬件相接的协议层了解网络结构和硬件特性,这样就对其它各层屏蔽了低层硬件细节。其次,在网络上传输数据是个复杂的过程,把这个复杂过程分成一系列线性关系的相对简单的处理过程,有利于网络软件实现的结构化,这对软件纠错和硬件故障检查非常有帮助。另外,协议分层的原则是:信宿机第  $n$  层收到的对象应当与信源机第  $n$  层发出的对象完全一致,这也使网络协议

软件有了更大的通用性,即在一个主机上设计的软件可应用于使用相同协议的另外的网络和主机。

本书介绍的软件就是在应用层的网络软件,了解协议分层有助于对应用层软件在概念上有深刻的理解。

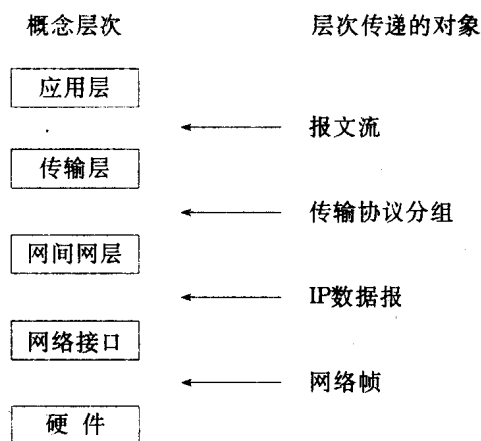


图 3.1 TCP/IP 协议

## 3.2 IP 地址

IP 地址也就是网间网地址。

在一个庞大的广域网环境中,将存在着各种各样的子网,每种子网采用不同的网络技术,即它们有不同的编址方式。假如从物理地址这个层次上看,网络地址是杂乱无章的。物理地址给网络上每台主机提供了一个唯一的标识符,但并没有给网络寻径带来任何好处。网络是用于两台主机之间的通信,从主机 A 到主机 B 必须在网络的物理空间中有一条虚电路,即通信链路。要在主机 A 到主机 B 之间建立一条虚电路,必须经过许多网络节点(node),特别是在象 Internet 这样的广域网上。虚电路的建立,首先需要网络寻径。要在杂乱的网路地址上建立和维护一条通信链路几乎是不可能的。

因此,需要一种不仅唯一标识网络上的主机而且能提供其在网络上的位置的统一格式的地址。网间网地址,即 IP 地址,就应运而生了。

网间网是通过网关将物理网络互联在一起的虚拟网,在任何一个物理网络上,各主机都有一个机器可识别的地址,就是物理地址。物理地址的长度、格式等是物理网络的一部分,物理网络技术不同,物理地址也不同。在一个广域网中可能存在不同类型的子网,不同的子网上的主机可能拥有不同的物理地址。

网间网为统一不同的物理地址,提供了 IP 协议。IP 协议提供了一种全网间网通用的地址格式,并在统一管理下进行地址分配,保证一个 IP 地址对应一台主机。

IP 地址是一个层次型地址,它带有其所标识主机的位置信息。网间网在概念上分为三

个层次，如下图：

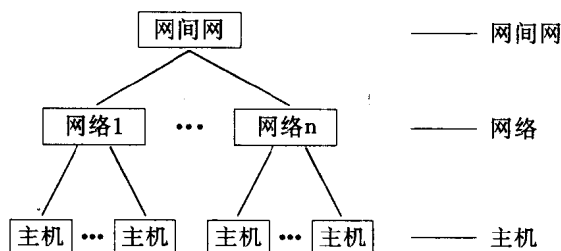


图 3.2 网间网层次结构

网间网地址的格式为：网络号·主机号。

IP 地址带有位置信息，给出一台主机的 IP 地址，马上就知道其位于那个网络上。

TCP/IP 协议规定了长度为 32 比特的 IP 地址。由网络号(netid)和主机号(hostid)各占多少位，IP 地址又分为 5 类(A,B,C,D,E)。不同类的 IP 地址有不同的子网数目，不同子网有不同的主机数目。

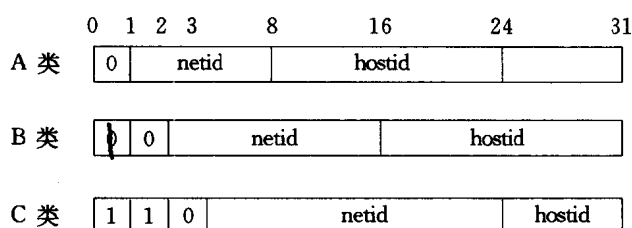


图3.3 网间网IP地址

不同类型 IP 地址的  
网络数和主机数

| 地址类型 | 网络数      | 主机数      |
|------|----------|----------|
| A    | $2^7$    | $2^{24}$ |
| B    | $2^{14}$ | $2^{16}$ |
| C    | $2^{21}$ | $2^8$    |

D 类是多目地址，E 类是留待后用的，不作介绍。

在协议软件中，网间网地址是以二进制形式存在的，这种形式是软件所偏爱的，却令一般的人感到头疼。于是，在面向人的文档中，网间网地址被直观地表示为四个以小数点隔开的十进制整数，其中每个整数对应一个字节。这种表示方法叫做“点分十进制表示”，如：202.38.68.144，其二进制表示为：11001010 00100110 01000100 10010000。

### 3.3 域名系统

在网间网中有两种地址：IP 地址和物理地址。物理地址是物理网络内部使用的地址，不同的物理网络，地址模式各不相同。IP 地址用于 IP 层以上的高层协议中，其目的在于屏蔽物理地址细节。由于采用了统一的 IP 地址，在网间网内部提供了一个全局性的通用地址，网间网上任意一对主机的上层软件(包括应用程序)才能相互通信，IP 地址为上层软件设计实现提供了极大的方便。对一般用户，IP 地址太抽象，为了向一般用户提供一种直观明了的主机标识符，TCP/IP 专门设计一种字符型的主机名字机制，即域名系统。

TCP/IP 的域名系统采用了树结构的命名机制,即一个域名的各个部分有树结构层次关系。这种层次关系不仅有利于从域名到 IP 地址的解析,使数据库管理变得简单,也使域名管理实现了由集中管理转向分级管理。即首先由中央管理机构(如 Internet 的 NIC)将最高一级名字空间划分为若干部分,并将各部分的管理权授予相应机构,各管理机构可以将管辖内的名字空间进一步划分成若干个子部分,并将各子部分的管理特权再授予若干子机构。

一个典型的 TCP/IP 层次型主机名语法为:

local • group • site

其中“local”表示本地名,“group”表示管理组名,“site”表示网点名。

举一个例子:

laser • phys • ustc • edu • cn

cn 表示中国地区的 Internet 网络, edu 表示中国教育科研网,ustc 表示中国科技大学, phys 表示物理系, laser 表示一台主机,那么此域名表示的是中国教育科研网中的中国科技大学校园网上的物理系的一台名为“laser”的主机。此域名的管理层次为: Internet 的 NIC 分配域名“cn”和“edu”给中国教育科研网管理机构;中国教育科研网管理机构又将域名“ustc”分配给中国科学技术大学网络信息中心;中国科学技术大学网络信息中心(USTC-NIC)分配给物理系域名“phys”,物理系给其一台主机命名为“laser”。域名“phys”和“laser”由 USTCNIC 管理。

Internet 的第一级域的域名有如下几个:

| 域 名          | 域        |
|--------------|----------|
| COM          | 商业组织     |
| EDU          | 教育机构     |
| GOV          | 政府部门     |
| MIL          | 军事部门     |
| NET          | 主要网络支持中心 |
| ORG          | 上述以外的组织  |
| INT          | 国际组织     |
| COUNTRY CODE | 国家(地理模式) |

图 3.4 Internet 的第一级域名

注意, internet 第一、二级域同属 Internet 的 NIC 集中管理。

## 第四章 socket 编程界面

本章将简单介绍 socket 概念和其在网络编程中的实现。

### 4.1 socket 的基本概念

Windows+TCP/IP 的网络编程继承了 UNIX+TCP/IP 网络编程的 socket 概念。socket 编程界面是由 Berkeley 大学在 UNIX 中首先提出的,目的是解决网间网进程之间的通信问题。

socket 的英文原义是“插座”。这个概念应用在网络编程上很形象生动。日常所见的插座,有的是信号插座,有的是电源插座,有的可以接收信号(或能量),有的可以发送信号(或能量)。假如在电话线与电话机之间安放一个插座,则 socket 非常类似于电话机插座。

将电话系统与面向连接的 socket 机制相比,有惊人相似的地方。以一个国家级电话网为例。电话的通话双方相当于相互通信的两个进程;通话双方所在地区(享有一个全局唯一的区号)相当于一个网络,区号是它的网络地址;区内一个单位的交换机相当于一台主机,主机分配给每个用户的局内号码相当于一个 socket 号。

任何用户在通话之前,首先要占用一部电话机,相当于申请一个 socket 号;同时要知道对方的电话号码,相当于对方有一个固定的 socket 号。然后向对方拨号呼叫,相当于发出连接请求(假如对方不在同一区内,还要拨对方区号,相当于给出网络地址)。对方假如在场并空闲(相当于通信的另一主机开机且可以接受连接请求),拿起电话话筒,双方就可以正式通话,相当于连接成功。双方通话的过程,是向电话机发出信号和从电话机接收信号的过程,相当于向 socket 发送数据和从 socket 接收数据。通话结束后,一方挂掉电话机,相当于关闭 socket,撤消连接。

在电话系统中,一般用户只能感受到本地电话机和对方电话号码的存在,建立通话的过程、语音传输的过程及整个电话系统的技术细节对他都是透明的,这与 socket 机制相似。socket 利用网间网通讯设施实现进程通讯,但它对通讯设施的细节毫不关心,通信设施能提供足够的通信能力,它就满足了。

至此,对 socket 进行了直观的描述。抽象出来,socket 实质上提供了进程通信的端点。进程通信之前,双方首先必须各自创建一个端点,否则是没有办法建立联系并相互通信的。

在网间网内部,每一个 socket 用一个半相关描述:

{ 协议,本地地址,本地端口 }

一个完整的 socket 接口用一个相关描述:

{ 协议,本地地址,本地端口,远地地址,远地端口 }

每一个 socket 有一个本地唯一的 socket 号,由操作系统分配。

最重要的是,socket 是面向客户-服务器模型而设计的,针对客户和服务程序提供不

同的 socket 系统调用。客户随机申请一个 socket(相当于一个想打电话的人可以在任何一台入网电话上拨号呼叫),系统为之分配一个 socket 号;服务器拥有全局公认的 socket,任何客户都可以向它发出连接请求和信息请求(相当于一个被呼叫的电话拥有一个呼叫方知道的电话号码)。

## 4.2 socket 的实现

### 4.2.1 创建 socket

应用程序在使用 socket 之前,首先必须拥有一个 socket。系统调用 socket( ),向应用程序提供创建 socket 的手段,socket( )调用格式如下:

sockid = socket( af, type, protocol )

返回值 sockid 是一个整数,即 socket 号,在本地是唯一的。创建一个 socket 实际上是向系统申请一个属于自己的 socket 号。

其中三个参数:

(1) af:地址族,指出本 socket 所用的地址类型。在当前,系统支持的地址族只有 AF\_INET。地址族与协议族是一一对应的,AF\_INET 地址族对应于 TCP/IP 协议族。

(2) type:类型,指创建的应用程序所希望的通讯服务类型。同一个协议族能提供多种服务类型,比如 TCP/IP 协议族提供的虚电路与数据报就是两种不同的通信服务类型。socket 支持下列几种通信服务类型(对应用程序来说即 socket 类型):

|                |             |
|----------------|-------------|
| sock_STREAM    | 流 socket    |
| sock_DGRAM     | 数据板 socket  |
| sock_RAW       | 原始 socket   |
| sock_SEQPACKET | 定序分组 socket |
| sock_RDM       | 可靠发送的消息     |

其中在 AF\_INET 中,类型 sock\_STREAM 对应于 TCP 协议;类型 sock\_DGRAM 对应于 UDP 协议。

(3) protocol:协议,指出本 socket 申请所希望的协议。指明该参数的原因是前面的两个参数对于描述协议类型不够充分,socket 特别提供这个域,让程序员直接了当地指出具体协议。有些协议族中不止一种协议支持同一类型服务,比如可能某协议有两种无连接数据报协议,尤其是许多协议中往往有多个下层协议支持原始 socket。

协议族,socket 类型和协议的可能组合如下:

| 协议族     | socket 类型   | 协议          | 实际协议 |
|---------|-------------|-------------|------|
| AF_INET | sock_DGRAM  | IPPROTO_UDP | UDP  |
| AF_INET | sock_STREAM | IPPROTO_TCP | TCP  |

图 4.1 socket 类型和协议的组合

综上所述,socket( )系统调用实际上指定了相关五元组中“协议”这一元。

#### 4.2.2 指定本地地址

调用 socket( )是 socket( )通信的第一步,目的在于创建一个希望的 socket,并获取其 socket 号。socket( )系统调用创建 socket 时,只指定了相关五元组的协议元,没有指定其它四元(本地地址、本地端口、远地地址、远地端口),因此需要别的系统调用加以补充。

bind( )将本地 socket 地址(包括本地主机地址和本地端口)与所创建的 socket 号联系起来,即将本地 socket 地址赋予 socket,以指定本地半相关。bind( )的作用相当于给 socket 命名,调用格式为:

```
bind(sockid, localaddr, addrlen)
```

其中:

(1) sockid, 是一个未命名 socket 的 socket 号

(2) localaddr, 本地 socket 地址, 指定本地半相关的其它两元: 本机主机地址和本地端口号。

localaddr 参数是一个指向 socket 地址结构的指针。

(3) addrlen, 地址长度, 指出以字节为单位的地址结构的长度。

各种 socket 地址数据结构包括两大部分: 地址类型和协议地址。网络协议地址又包括主机地址和端口号。

#### 4.2.3 建立 socket 连接

有两个系统调用用于整个相关的建立。其中 connect( )用于建立连接。这里的连接有两层含义: 第一层是指两个 socket 之间的沟通; 第二层指传输层连接(比如 TCP 连接)。

无连接 socket 的进程也可以调用 connect( ), 但这时在本地系统与远地系统之间不进行实际的报文交换, 调用将从本地操作系统直接返回, 也就是说, 并没有真正地建立 socket 连接或传输连接。无连接 socket 的这种调用方式的意义在于通知操作系统, 将来自指定地址的 socket 的数据送往本 socket。

connect( )的调用格式如下:

```
connect(sockid, destaddr, addrlen)
```

其中:

(1) sockid, 是欲建立连接的本地 socket 号。

(2) deataddr, 是一个指向对方 socket 地址(信宿地址)结构的指针。

(3) addlen, 指出对方 socket 地址长度。

accept( )用于面向连接的服务器, 其调用格式为:

```
newsock = accept( sockid, clientaddr, paddrlen)
```

其中:

(1) sockid, 本地 socket 号。

(2) clientaddr, 指向客户地址结构的指针。