HZ BOOKS

Designed for Microsoft® Windows NT® Windows® 98

附 赠 CD-ROM

Network Programming for Microsoft Windows

微 软 公 司
核心技术书库

（美）Anthony Jones Jim Ohlund 著
京京工作室 译

Windows 网络编程技术

内容涉及
Windows 2000
Windows CE

机械工业出版社
China Machine Press

Microsoft Press

微软公司核心技术书库

Windows网络编程技术

(美) Anthony Jones
Jim Ohlund 著

京京工作室 译

机械工业出版社
China Machine Press

本书专门讨论Windows网络编程技术，覆盖Windows 95/98/NT 4/2000/CE平台。内容包括NetBIOS和Windows重定向器方法、Winsock方法、客户端远程访问服务器方法。本书论述深入浅出、用大量实例详解了微软网络API函数的应用。配套光盘包含了所有实例代码，方便读者使用。本书适合中、高级程序设计人员以及网络设计与管理人员参考。

本书版权登记号：图字：01-1999-2863

图书在版编目(CIP)数据

Windows网络编程技术 /（美）琼斯（Jones, A.），（美）奥朗德（Ohlund, J.）著；京京工作室译. -北京：机械工业出版社，2000.3

（微软公司核心技术书库）

书名原文：Network Programming for Microsoft Windows

ISBN 7-111-07809-8

Ⅰ.W... Ⅱ.①琼... ②奥... ③京... Ⅲ.计算机网络-窗口软件，Windows-程序设计 Ⅳ.TP316.7

中国版本图书馆CIP数据核字（1999）第57979号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码100037)

责任编辑：吴 怡

北京市密云县印刷厂印刷 · 新华书店北京发行所发行

2000年3月第1版第1次印刷

787mm × 1092mm 1/16 · 31.25印张

印数：0 001-6 000册

定价：68.00元（附光盘）

译 者 序

随着计算机网络技术的飞速发展，越来越多的人认识到了网络编程的重要性。Internet的应用，如电子商务等，都要求通过网络编程为用户提供具有高度互动性的内容。为适应新时代的要求，各行各业都产生了“上网”的需求，越来越多的程序员需要掌握这门技能。

微软公司独立研制了许多出色的网络技术，但在这之前，市面上尚无一本参考书系统讲解如何利用一系列相关的应用程序编程接口（API），来配合运用这些技术。本书开创历史之先河，首次完整披露了如何在32位平台上使用传统的API，如NetBIOS，以及如何运用一些新型网络API，如Winsock 2和远程访问服务等等。本书详细解释了大量网络专用函数，它们适用于Windows 95、Windows 98、Windows NT 4、Windows 2000以及Windows CE操作系统平台。本书主要面向中、高级程序设计人员，但即便是编程新手，也同样能从本书入门，迅速掌握网络编程技术。

本书的两位作者Anthony Jones和Jim Ohlund都是网络编程领域的专家，并都就职于微软公司。为使我国计算机同行更好地掌握这门方兴未艾的技术，我们翻译了本书，希望她能成为您的好帮手。

由于本书涉及到的网络编程专业要求比较高，书中内容新，加之翻译时间要求紧，疏漏之处在所难免。如果读者发现了不准确或不完善的地方，请务必告诉我们，以便再版时修订。

译 者

1999年11月

前　言

本书深入探讨了大量网络专用函数，适用于Windows 95、Windows 98、Windows NT 4、Windows 2000以及Windows CE。主要面向中、高级程序设计人员，但即使对编程新手，本书也同样是一本有价值的参考书。

怎样使用本书

本书主要内容有：

- 使用NetBIOS和Windows重定向器进行网络编程。
- Winsock。
- 客户端远程访问服务器。

第1章讲述的是NetBIOS。根据我们在“微软开发人员支持小组”工作的经验，目前有许多公司仍在使用这一技术。而且到目前为止，除本书以外已找不到讲解如何在Win32中编写NetBIOS应用的更为详尽的资料了。第1章也提供了一系列有用的方法，指导编写健壮的和跨平台的应用（特别是由于许多开发者仍需通过NetBIOS同老系统通信）。

第2章到第4章探讨了Windows重定向器、邮槽以及命名管道的知识。你或许已经知道，邮槽及命名管道均是建立在“重定向器”的基础上的。因此，我们决定用单独一章来讲述重定向器的问题，以便铺垫足够多的背景知识，切实掌握这三种技术。“邮槽”是一种不可靠的、单向的、面向消息的API，并不依赖系统中运行的具体协议。“命名管道”则提供了更多的特性，比如可靠性、双向式、面向流与消息的数据传输等等。通过重定向器，在毋需新的网络API的前提下，命名管道也为Windows NT提供了安全保障。

本书第二部分专门讲解Winsock API的问题。第5章是对Winsock的一个概括，介绍了程序员最常用的一些Winsock协议。所有Winsock应用都必须先创建一个套接字，否则无法进行网络通信。这一章向大家介绍了每种协议的功用。而第6章，我们则具体解释了针对每一种类型的协议，如何创建一个套接字，以及如何进行简单的名字解析。

第7章是本书精华的开始。在这一章，我们介绍了基本的客户机／服务器编程模型，并探讨了涉及建立连接和接受、数据传输以及其他方面的大多数Winsock函数。第8章接着上一章的主题，探讨了Winsock提供的各种I/O（输入／输出）方法。由于第7章仅是一个简单的概括，讨论的是最简单的I/O方法，所以到第8章，我们便深入讲解了其他I/O方法的运用。如果你以前从未接触过Winsock，那么从第5章到第7章的知识便为你以后使用API提供了一个坚实基础。

第二部分后几章都从某个特定的方面探讨了Winsock。套接字选项和I/O控制命令是第9章的主题。在这一章，我们探讨了影响套接字或协议本身行为的大多数命令。不论是作为自学内容，还是作为参考资料，该章都是非常有用的。

第10章将重点转向Winsock 2名字注册和解析。这是一种与协议无关的方法，可注册服务名称，并将其解析为基层协议的地址。随着近期Windows 2000及“活动目录”（Active

Directory）的发布，这一章的内容显得尤其重要。

第11章探讨了“多点传送”通信方式，包括IP多播以及ATM多播。在第12章，我们则探讨了服务质量的问题。这是一种激动人心的新技术，可担保为特定的应用分配足够的网络带宽。第13章则将重点放在原始IP套接字上面。在这一章，我们解释了Winsock应用如何通过原始IP套接字来使用Internet控制消息协议（ICMP）以及Internet组成员资格协议（IGMP）；另外还就原始套接字编程的其他问题进行了探讨。

第14章讨论了Winsock服务提供者接口（SPI）。利用该接口，程序员可在Winsock及更低层的服务提供者之间安装一个新层，以便控制套接字及协议行为，或者对名字的注册及解析进行控制。这属于一种高级特性，软件开发人员通过它可极大扩展Winsock的功能。

在第二部分的第15章讨论了微软Visual Basic Winsock控件。由于许多开发人员仍在使用Visual Basic以及这种控件，所以我们决定增加这一章的内容。尽管控件本身能力有限，不能很好地利用Winsock的某些高级特性，但对VB开发人员来说，假如他们需要的只是实现简单的、易于使用的网络通信，那么控件还是非常有用的。

第三部分将重点放在客户端远程访问服务器（RAS）上。由于Internet以及拨号上网方式的普遍运用，专为RAS增加一章是颇有必要的。为网络应用增加了拨号功能后，程序员会发现它将特别受到用户的欢迎，因为这样一来，整个程序使用起来更容易了。换言之，最终用户不必知晓如何建立一个拨号连接，便可直接使用你编制的网络应用。

本书最后部分包括了三个附录。其中，附录A是一个颇有价值的NetBIOS命令索引。对每个命令都列出了要求的输入及输出参数。附录B讲解了新增的IP助手函数，可就当前计算机的网络配置提供有用的信息。附录C是Winsock错误代码索引，详细讲解了各种错误，并概括了其出现的原因。

我们衷心希望本书成为你的得力学习工具及参考手册。我们有理由相信，本书是迄今为止最为全面的Windows网络编程技术指南。

配套光盘的用法

本书每章都展示了一系列示范代码，向大家阐述如何运用我们介绍的网络API函数。这些示例也同时包括在配套光盘中。要想安装它们，请将CD插入你的光盘驱动器，“自动播放”特性能自动启动安装程序。另外，假如没有自动播放，你也可以找到位于根目录的PressCD.exe，自行开始安装。既可将示范代码安装到你的计算机，亦可从CD直接使用（在Examples\Chapters\Chapter XX下）。

注意，本光盘要求运行32位微软Windows操作系统的机器才能运行。

除示范代码外，光盘内还提供了最新版本的微软Platform SDK。之所以要提供这个SDK，是因为许多例子需要依赖最新的头文件和函数库，这些文件和函数自Windows 2000 Beta 3版之后才有。

本书技术支持

我们已竭尽全力保证本书及配套光盘内容的准确无误。如果你发现了其中有错，请访问http://mspress.microsoft.com/support，向微软出版社报告你的发现。

本书陈列的许多函数定义及表格都获得了微软Platform SDK文档编制组的鼎力相助和许

可。有些材料建立在早期版本的文档上，所以可能会发生变化。要想了解最新的Platform SDK信息，或索取最近的更新和错误修正文件，请访问http://msdn.microsoft.com/developer/sdk/platform.asp。

如果对本书有任何建议、意见或者问题，请用普通邮政或电子邮件的方式，同微软出版社和出版中译版的机械工业出版社华章公司取得联系：

Microsoft Press
Attn:Network Programming for Microsoft Windows editor
One Microsoft Way
Redmond, WA 98052-6399
mspinput@microsoft.com

本书英文版书号为：ISBN 0-7356-0560-2

机械工业出版社华章公司
北京市西城区百万庄南街1号，邮编：100037
E-mail:huazhang @ public 3.bta.net.cn

目　录

第一部分 传统网络API

本书第一部分讲述的是传统的网络接口NetBIOS、重定向器以及通过重定向器进行的各类网络通信。尽管本书大部分内容均围绕Winsock编程这一主题展开，但是，API比起Winsock来，仍然具有某些独到之处。其中，第1章探讨的是NetBIOS接口，它和Winsock类似，也是一种与协议无关的网络API。NetBIOS提供了异步调用，同时兼容于较老的操作系统，如OS/2和DOS等等。第2章讨论了重定向器的问题，它是接下去的两个新主题——邮槽（第3章）和命名管道（第4章）的基础。重定向器提供了与传输无关的文件输入／输出方式。邮槽是一种简单的接口，可在Windows机器之间实现广播和单向数据通信。最后，命名管道可建立一种双向信道，这种信道提供了对Windows安全通信的支持。

第1章 NetBIOS

“网络基本输入／输出系统”（Network Basic Input/Output System, NetBIOS）是一种标准的应用程序编程接口（API），1983年由Sytek公司专为IBM开发成功。NetBIOS为网络通信定义了一种编程接口，但却没有详细定义物理性的“帧”如何在网上传输。1985年，IBM创制了NetBIOS扩展用户接口（NetBIOS Extended User Interface, NetBEUI），它同NetBIOS接口集成在一起，终于构成了一套完整的协议。由于NetBIOS接口变得愈来愈流行，所以各大厂商也开始在其他如TCP/IP和IPX/SPX的协议上实施NetBIOS编程接口。到目前为止，全球已有许多平台和应用程序需要依赖于NetBIOS，其中包括Windows NT、Windows 2000、Windows 95和Windows 98的许多组件。

注意 Windows CE并不支持NetBIOS API，只是用TCP/IP作为其传送协议，并同时支持NetBIOS的名字与名字解析。

Win32 NetBIOS接口向后兼容于早期的应用程序。本章要讨论的是NetBIOS编程基础。首先向大家介绍的是NetBIOS的一些基本知识，从NetBIOS的名字及LANA编号开始，接着，我们围绕NetBIOS提供的基本服务展开讨论，比如面向会话和“无连接”通信等等。在每一节，都展示了一个简单的客户机和服务器示例。在本章最后，我们陈列了程序员需留意的一系列陷阱以及易犯的错误。在本书的附录A中，大家可找到一份命令索引，其中对每个NetBIOS命令都进行了总结，包括必要的参数，以及对其行为的简单说明。

OSI 网络模型

“开放系统互连”（OSI）模型从一个很高的层次对网络系统进行了描述。OSI模型总共包含了七层。从最顶部的“应用层”开始，一直到最底部的“物理层”，这七个层完整阐述了最基本的网络概念。图1-1展示的正是OSI模型的样子。

层	描　述
应用层	为用户提供相应的界面，以便使用提供的连网功能
表示层	完成数据的格式化
会话层	控制两个主机间的通信链路（开放、操作和关闭）
传输层	提供数据传输服务（可靠或不可靠）
网络层	在两个主机之间提供一套定址/寻址机制，同时负责数据包的路由选择
数据链路层	控制两个主机间的物理通信链路；同时还要负责对数据进行整形，以便在物理媒体上传输
物理层	物理媒体负责以一系列电子信号的形式，传出数据

图1-1　OSI网络模型

对应OSI模型，NetBIOS主要在会话和传输层发挥作用。

1.1　Microsoft NetBIOS

如前所述，NetBIOS API实施方案适用于为数众多的网络协议，使得编程接口“与协议无关”。换言之，假如根据NetBIOS规范设计了一个应用程序，它就能在TCP/IP、NetBIOS甚至IPX/SPX上运行。这是一项非常有用的特性，因为对一个设计得当的NetBIOS应用程序来说，它几乎能在任何机器上运行，无论机器连接的物理网络是什么。然而，我们也必须留意几个方面的问题。要想使两个NetBIOS应用（程序）通过网络进行正常通信，那么对它们各自运行的机器来说，至少必须安装一种两者通用的协议。举个例子来说，假定小张的机器只安装了TCP/IP，而小马的机器只安装了NetBEUI，那么对小张机器上的NetBIOS应用来说，便无法同小马机器上的应用进行通信。

除此以外，只有部分协议实施了NetBIOS接口。Microsoft TCP/IP和NetBEUI在默认情况下已提供了一个NetBIOS接口；然而，IPX/SPX却并非如此。为此，微软专门提供了一个IPX/SPX版本，在其中实现了该接口。在设计网络时，这个问题必须注意。安装协议时，具有NetBIOS能力的IPX/SPX协议通常会自动提醒你注意这方面的问题。例如，Windows 2000提供的协议本身就叫作“NWLink IPX/SPX/NetBIOS兼容传送协议”。而在Windows 95和Windows 98中，请留意IPX/SPX协议属性对话框，其中有一个特殊的复选框，名为“希望在IPX/SPX上启用NetBIOS”。

另外要注意的一个重要问题是NetBEUI并非是一种“可路由”协议。假定在客户机和服务器之间存在一个路由器，那么这种协议在两部机器上的应用便无法沟通。收到数据包后，路由器便会将其“无情地”地抛弃。TCP/IP和IPX/SPX则不同，它们均属“可路由”协议，不会出现这方面的问题。要注意的是，假如你需要在很大程度上依靠NetBIOS，那么在配置网络时，至少应安装一种可路由的传送协议。要想深入了解各种协议的特征以及相应的注意事项，请参阅第6章。

1.1.1　LANA编号

从编程角度思考，大家或许会觉得奇怪，传送协议与NetBIOS如何对应起来呢？答案便在于LAN适配器（LAN adapter, LANA）编号，它是我们理解NetBIOS的关键。在最初的NetBIOS实施方案中，每张物理网卡都会分配到一个独一无二的值：即LANA编号。但到Win32下，这种做法便显得有些问题。因为对一个工作站来说，它完全可能同时安装了多种网络协议，也可能安装了多张网卡。

每个LANA编号对应于网卡及传输协议的唯一组合。例如，假定某工作站安装了两张网卡，以及两种具有NetBIOS能力的传输协议（如TCP/IP和NetBEUI），那么总共就有四个LANA编号。下面是一种对应关系的例子：

0. TCP/IP——网卡1
1. NetBEUI——网卡1
2. TCP/IP——网卡2
3. NetBEUI——网卡2

通常，LANA编号的范围在0到9之间，除LANA 0之外，操作系统并不按某种固定的顺序来分配这些编号。那么，LANA 0有什么特殊含义呢？LANA 0代表的是“默认”LANA！NetBIOS问世早期，许多应用都采用硬编码的形式，只依赖LANA 0进行工作。在那时，大多数操作系统也只支持一个LANA编号。考虑到向后兼容的目的，我们可将LANA 0人工分配给一种特定的协议。

在Windows 95和Windows 98中，通过选择控制面板中的“网络”图标，可访问一种网络协议的“属性”对话框。在“网络”对话框中选择“配置”选项卡，再从网络组件列表中选择一种网络协议，按下“属性”按钮即可。对具有NetBIOS能力的每一种协议来说，其属性对话框的“高级”选项卡都有一个“设成默认的通信协议”复选框。若选中这个复选框，会重新安排协议的绑定，使默认协议能够分配到LANA 0。注意在任何时候，只能有一种协议才能选中这个复选框。由于Windows 95和Windows 98具有所谓的“即插即用”功能，所以我们没有其他办法可对协议的编号顺序进行更改。

Windows NT 4则允许用户在设置NetBIOS时拥有更大的灵活性。在“网络”对话框的“服务”选项卡中，可从“网络服务”列表框内选择NetBIOS接口，然后点按“属性”按钮。随后便会出现“NetBIOS配置”对话框，在这里可针对每一对网卡／传输协议的组合，分配各自的LANA编号。在这个对话框中，每张网卡都以其驱动程序的名字加以标识；但协议名称却显得有些暧昧。在图1-2中，我们展示了NetBIOS配置对话框的样子。单击其中的“Edit”（编辑）按钮，便可为每种协议单独分配LANA编号。Windows 2000也允许我们单独分配LANA编号。在控制面板中，双击“网络和拨号连接”图标。随后，从“高级”菜单中选择“高级设置”，然后在高级设置对话框中选择“LANA编号”选项卡。

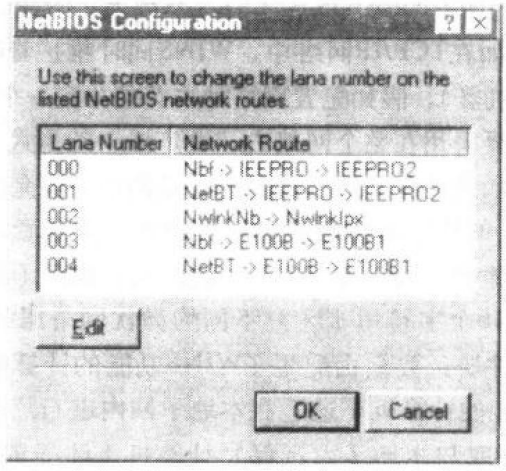

图1-2 NetBIOS配置对话框。这是一部多宿主机器，安装了两张网卡和三种传输协议：TCP/IP（NetBT）、NetBEUI（Nbf）以及IPX/SPX（NwlnkNb）

要想设计出一个“健壮”的NetBIOS应用，必然需要让自己的代码能对任意LANA编号上的连接进行控制。例如，假定小马编写了一个NetBIOS服务器应用，对LANA 2上的客户机进行监听。在小马的机器（即服务器）上，LANA 2正好对应于TCP/IP。后来，小张需要编写一个客户端应用，同小马的服务器通信，所以他决定让自己的程序通过工作站的LANA 2建立连接。然而，小张工作站上的LANA 2对应于NetBEUI。这样一来，两个应用相互间均无法通信——尽管两者都安装了TCP/IP和NetBEUI。为纠正协议的这种差异，小马的服务器应用程序必须对小马工作站上每个可能的LANA编号上的客户机连接进行“监听”。类似地，小张的客户机应用程序需要针对本机每个可能的LANA编号，尝试在其上面的连接。只有这样，小马和小张才能保证自己的应用尽最大可能成功通信。当然，尽管我们需要在代码中对任何LANA编号上的连接进行控制，但并不表示能够百分之百地成功。假如两台机器根本就没有安装一种共通的协议，那么无论如何都是不能成功的!

1.1.2 NetBIOS名字

现在，我们知道了LANA编号是什么，接着再来讨论NetBIOS名字（名称）的问题。对一个进程（或“应用”、“应用程序”）来说，它会注册自己希望与其通信的每个LANA编号。一个NetBIOS名字长度为16个字符，其中第16个字符是为特殊用途保留的。在名字表内添加一个名字时，应将名字缓冲区初始化成空白。在Win32环境中，针对每个可用的LANA编号，每个进程都会为其维持一张NetBIOS名字表。若为LANA 0增添一个名字，意味着你的应用程序只能在LANA 0上同客户机建立连接。对每个LANA来说，能够添加的名字的最大数量是254，编号从1到254（0和255由系统保留）。然而，每种操作系统都设置了一个低于254的最大默认值。重设每个LANA编号时，我们可对此默认值进行修改。

另外，NetBIOS名字共有两种类型：唯一名字和组名。“唯一名字”意味着它是独一无二的：网络上不能再有其他任何进程来注册这个名字。如果一台机器已注册了某名字，那么在你注册该名字时，便会收到一条“重复名字”出错提示。大家或许已经知道，微软网络中的机器名采用的便是NetBIOS名字。机器启动时，会将自己的名字注册到本地的“Windows互联网命名服务器”（WINS）。如果事前已有另一台机器注册了同样的名字，WINS服务器便会报错。WINS服务器维护着已注册的所有NetBIOS名字的一个列表。除此以外，随名字一道，还可保存协议特有的一些信息。比如在TCP/IP网络中，WINS同时维护着NetBIOS名字以及注册那个名字的IP地址（亦即相应的机器）。假如配置网络时未为其分配一个WINS服务器，那么如何检查名字是否重复呢？这时便要采用在整个网络内“发广播”的形式。当一名发送者向全网络发出一条特殊的广播消息时，如果没有其他机器回应这条消息，便允许发送者使用该名字。

而在另一方面，“组名”的作用是将数据同时发给多个接收者；或者相反，接收发给多个接收者的数据。组名并非一定要“独一无二”，它主要用于多播（多点发送）数据通信。

在NetBIOS名字中，第16个字符用于区分不同的微软网络服务。各种网络服务和组名需要用一个WINS服务器完成注册。要么由配置了WINS功能的计算机进行名字的直接注册，要么由那些尚未配置WINS功能的计算机，通过在本地子网内进行广播注册。Nbtstat命令是一个非常有用的工具，可用它获取与本地（或远程）计算机上注册的NetBIOS名字有关的信息。在表1-1展示的例子中，Nbtstat-n命令可针对用户“Davemac”，生成这个已注册的NetBIOS名字的列表。Davemac登录进入的那部计算机已被配置成一个主域控制器，而且运行的是

Windows NT Server操作系统，且已安装了Internet信息服务器（IIS）。

表1-1 NetBIOS名字表

名 字	第16个字节	名字类型	服 务
DAVEMAC1	<00>	唯一	工作站服务名
DAVEMAC1	<20>	唯一	服务器服务名
DAVEMACD	<00>	成组	域名
DAVEMACD	<1C>	成组	域控制器名
DAVEMACD	<1B>	唯一	主控浏览器名
DAVEMAC1	<03>	唯一	发信者名
Inet~Services	<1C>	成组	Internet信息服务器组名
IS~DAVEMAC1	<00>	唯一	Internet信息服务器唯一名
DAVEMAC1+++++++	<BF>	唯一	网络监视器名字

只有在安装了TCP/IP协议的前提下，才会安装Nbtstat命令。该工具亦可用来查询远程机器的名字表，方法是在远程机器的名字后面，接上一个-a参数；或在远程机器的IP地址后，接上一个-A参数。

在表1-2中，我们总结了各种不同的Microsoft网络服务为唯一NetBIOS计算机名追加的默认第16个字节值。

表1-2 唯一名字标识符

第16个字节	含 义
<00>	工作站服务名。通常，它对应于NetBIOS计算机名
<03>	收发消息时采用的信使服务名。WINS服务器会将这个名字注册成WINS客户机上的信使服务，并通常追加到计算机名后面，以及当前登录到计算机的用户名的后面
<1B>	域主控浏览器名。这个名字用于标识主域控制器，并指出用什么客户机和其他浏览器同域主控浏览器取得联系
<06>	远程访问服务（RAS）服务器服务
<1F>	网络动态数据交换（NetDDE）服务
<20>	用于为文件共享提供“共享点”的服务器服务名
<21>	RAS客户机
<BE>	网络监视器代理
<BF>	网络监视器工具

表1-3则列出了在常用的一系列NetBIOS组名后，追加的默认第16个字节字符。

如此多的标识符很易使人产生混淆，很难真正记住。所以，请考虑把它作为一个“速查表”或“索引”使用。大家或许不应在自己的NetBIOS名字中使用它们。为防止偶然同你的NetBIOS名字发生冲突，最好避免使用唯一名字标识符。对于组名，恐怕更要引起高度注意——假如你的名字同一个已有的组名相同，那么不会产生任何错误提示。若发生这种情况，结果就是会收到原本发给其他人的数据。

表1-3 组名标识符

第16个字节	含 义
<1C>	一个域组名，在这个组内包含了已注册域名的一系列计算机的特定地址。由域控制器来注册这个名字。WINS将它当作一个域组看待：组内每个成员必须单独更新自己的名字。域组最多只能包容25个名字。若复制的一个静态1C名字同另一个WINS服务器上的某个动态1C名字发生冲突，便会增加成员的一个“联合”，同时将记录标定为“静态”。假如记录是静态的，组内成员便不必定时刷新自己的IP地址

（续）

第16个字节	含　义
<1D>	指定一个主控浏览器的名字，客户机通过它访问主控浏览器。在一个子网上，只能有一个主控浏览器。WINS服务器会对域名注册作出“正”（肯定）响应，但却不会将域名保存在自己的数据库中。假如一台计算机向WINS服务器送出一个域名查询，则WINS服务器会返回一个“负”（否定）响应。若送出域名查询的那台计算机已被配置成h节点或m节点，便会随之广播那个查询，以解析出正确的名字。客户机解析名字的方法是由节点的类型决定的。如客户机配置成b节点解析，便会送出广播包，以便广告并解析出NetBIOS名字。p节点解析采用与WINS服务器的点到点通信方式。而m节点属于b及p节点的一种混合形式：首先使用的是b节点；如有必要，再接着使用p节点。最后一种解析方式是h节点，亦称“混合模式”。它无论如何都会先尝试使用p节点注册和解析，然后只有在解析失败的前提下，才会换用b节点。Windows操作系统默认为h节点
<1E>	一个普通组名。浏览器可向这个名字发送广播数据，并通过对它的监听来挑选一个主控浏览器。这些广播面向的是本地子网，绝对不应通过路由器传输
<20>	一个Internet组名。这种类型的名字由WINS服务器进行注册，以便为了管理方面的目的来标定特定的计算机组。例如，“printersg”可以是一个注册的组名，用于标定由打印服务器构成的一个管理性组
MSBROWSE	不再是单独一个追加的第16位字符，“_MSBROWSE_”需要追加到一个域名后面，并在本地子网上进行广播，向其他主控浏览器通告这个新增的域

1.1.3 NetBIOS特性

NetBIOS同时提供了“面向连接”服务以及“无连接”服务。面向连接的服务，是指它允许两个客户机相互间建立一个会话，或者说建立一个“虚拟回路”。这种“会话”实际是一种双向的通信数据流，通信的每一方都可向另一方发送消息。面向连接的服务可担保在两个端点之间，任何数据都能准确无误地传送。在这种服务中，服务器通常将自己注册到一个已知的名字下。客户机会搜寻这个名字，以便建立与服务器的通信。就拿NetBIOS的情况来说，服务器进程会针对想通过它建立通信的每一个LANA编号，将自己的名字加入与其对应的名字表。而对位于其他机器上的客户来说，就可将一个服务名解析成机器名，然后要求同服务器进程建立连接。大家可以看到，为建立这种虚拟回路，必须采取一些适当的步骤。而且在初次建立连接的时候，还会牵涉到一些额外的开销。“面向连接”或“面向会话”的通信可保证通信具有极高的可靠性，而且数据包的收发顺序亦能确保正确无误。然而，它仍然是一种“以消息为基础”的服务。也就是说，假如已连接好的某个客户机执行一个“读”命令，那么服务器在流中仍然只会返回一个数据包——尽管客户机此时提供了一个足够大的缓冲区，可同时容下几个包！

“无连接”或数据报服务中，服务器并不将自己注册到一个特定的名下，而只是由客户机收集数据，然后将其送入网络，事前不必先建好任何连接（即无连接）。对于数据的目的地址，客户机会将其定义成服务器相应进程对应的NetBIOS名字。这种类型的服务不提供任何保障，但同面向连接的服务相比，却可有更好的性能，如在使用数据报服务（无连接服务）时，省下了建立连接所需的开销。例如，客户机可能向服务器兴冲冲地一下子发出数千字节的数据，但那台服务器早在一两天前便已当机了。除非依赖自服务器传来的响应，否则客户机永远都收不到任何错误提示（在这种情况下，假如在一个特定的时间段内，没有收到任何响应，便