本章的重点仍然是一些网络工具，它们可用于管理系统及其它的网络设备，例如交换器、路由器、集线器和其它支持SNMP的设备等。这部分介绍的网络工具很有用，它们对于你已经掌握的软件来说，是一个很好的补充。这里先引入几个概念：   
•系统节奏（system heartbeat）它是一条SNMP的get-request请求，管理员使用该请求  
来 确定管理代理和系统的一般的可连接性。举例说明明，网络管理员可以查询管理代理的系统时钟MIB变量，可以确定每次连续的查询都比前一次查询晚一些。各次 连续查询应该表明时间在向前移动。可用于这方面的MIB变量是数据对象unitTime ，它是Sun系统管理代理的sunSystem组的一部分。此外，MIB-II的system组中的数据对象sysUpTime也可用于这个方面，所有的 SNMP管理代理应该都可以使用这个数据对象。  
•系统开/关消息 如果一个系统因某种原因被关闭或重启，此时系统应该以trap形式向  
指定的网络管理系统发送一条报文。trap是一种由管理代理主动发送的报文，它表明某些特定条件或事件。UCD管理代理的配置文件和其它工具可用于向一个或多个网络管理系统续传trap报文。  
•协议统计信息 因为许多SNMP管理代理支持MIB-II标准，所以有可能实现协议性  
能监控和系统监控。这包括IP、ICMP、TCP、SNMP协议，网络接口计数器，及一些Sun系统性能数据对象。  
•系统进程活动性 Sun MIB扩展和UCD管理代理的应用，使得实现系统进程的监控  
也成为可能。通过监控系统的关键进程，我们可以使用进程数据对象来确保某个网络或一个独立系统的正常运转。  
•路由监控 路由监控管理代理可用于确定一个系统的路由配置、报告发现的错误。举  
例说明明，假如某个组织确定每台机器都必须有一个缺省的路由，我们可以通过查询这些设备中的SNMP管理代理来验证这个路由。  
•接口性能 MIB-II还提供了这样一类数据对象，其中包含某个SNMP设备中安装的  
各个接口的性能信息。所以，我们可以获得这些性能信息，实现监控各个激活接口的性能。  
•配置控制 许多网络设备（例如路由器和交换器）在它们用于网络之前，必须先被配  
置。另外，在网络要求变化时，网络设备中的配置也必须相应改变。这部分列出的工具支持设备配置的改动。  
13.1  UCD命令  
UCD 软件包不仅提供了一个功能强大的SNMP管理代理，而且还提供了一系列使用方便的工具，它们可用于管理支持SNMP协议的网络设备。这些工具支持 SNMPv1和SNMPv2协议；另外，还支持一部分SNMPv3协议。在使用上述工具时，你可以通过命令行选项来选择你要使用的SNMP的版本号。此 外，这些工具还可以用于编译实现复杂网络管理功能或自定义任务的命令解释程序或其它程序。如果我们完全安装了整个UCD软件包，那么在系统中还同时安装了 一套完整的UNIX帮助信息手册，它包括了本章介绍的所有UCD工具。该手册的缺省位置是 /usr/local/man。表13-1列出了UCD软件包提供的工具及应用程序：  
表13-1  UCD的SNMP工具  
  UCD工具                          说 明  
snmpbulkwalk 通过SNMP的bulk-request访问MIB数据对象  
snmpdelta 监控SNMP变量的变化  
snmpget 访问一个或多个MIB数据对象  
snmpgetnet 连续遍历整棵SNMP MIB结构树，访问所有支持的MIB数据对象  
snmpnetstat 获得管理代理的接口配置信息  
snmpset 设置一个或多个MIB数据对象为指定值  
snmpstatus 获得重要的MIB数据对象信息  
snmptable 访问一整张SNMP表格  
snmptest 与另一个SNMP管理代理实体通信  
snmptranslate 把MIB数据对象信息转换成更有实际意义的信息  
snmptrap 向一个或多个管理员发送SNMP trap报文  
snmptrapd 从网络中接收SNMP trap报文  
snmpwalk 访问一个组的MIB数据对象  
   
UCD工具的基本句法是：  
 snmpcmd protocol\_version [additional\_options]hostname community  
 object [object]  
   
具体说明如下：  
•字符串snmpcmd是一个占位符号，它表示上面表中列出的UCD命令之一。  
• 字符串protocol\_version的值是1、2c或2p 。该字符串标识符指定应该使用的SNMP  
协 议的版本号。这里1代表标准SNMPv1 。2c 代表的SNMPv2版本在支持的SNMP协议数据单元方面与SNMPv1不同，但它使用与SNMPv1相同的基于分区（community）的安全模型。 2p 代表的SNMPv2版本使用基于各组成部分（party）的安全模型。  
•字符串addition\_options的选项内容列在表6.2和表6.3中，用于控制UCD工具的操  
作属性和显示属性。在后面会介绍到最通用的一部分命令行选项。  
•字符串hostname也是一个占位符，它表示网络上的一个主机的名字，这个主机中包  
含了与命令行protocol\_version信息相匹配的SNMP管理代理。在这里也可以用一个有效的标准形式的IP地址来替换主机名。  
•字符串object表示一个应该被访问（使用SNMP GET）或被改动（使用SNMP SET）  
的MIB OID。它可以是使用带点的数字形式的表示式，也可以是带点的名字形式的表示式。在进行SNMP GET操作时，这时要得到的是附加的数据对象信息。需要注意的是，在命令行中一次可以指定多个数据对象。  
13.1.1  通用的命令行选项  
UCD 工具的命令行中可以使用很多个命令行选项参数。记住一些最通用的命令行选项可以使我们更加方便的使用这些工具。所有的工具都支持的命令行选项可以分成两 类：操作选项和显示选项。操作选项控制着各个工具的性能，而显示选项控制着如何显示MIB数据对象和其相应的值、及其它一些信息。  
表13-2列出的是显示选项，一共有8个：  
表13-2  通用的显示选项  
  选 项                    说 明  
   -d 清除SNMP包  
   -D 显示故障排除信息  
   -h 显示帮助信息  
   -f 显示完整的数据对象标识符路径  
   -q 使显示结果更适合程序解析  
   -s 只显示标识符的后缀部分  
   -S 同时显示标识符的后缀部分和MIB名  
   -V 显示版本信息  
   
上 面列出的选项中，有三个是用于控制以何种格式显示MIB路径的。这三个选项是-f 、-s和-S 。选项–f 用于显示完整的数据对象标识符路径信息。选项–s 只允许打印路径的后缀部分，即只显示MIB数据对象标识符的最后一个部分符号。最后一个是选项-S ，它要求显示MIB数据对象标识符的后缀部分和MIB名。如果你需要收集一个设备的SNMP信息，并把这些信息作为另一个程序的输入信息，这时使用选项 -q 会很方便。选项-q代表一种快速格式（quick format），使用它可以使输出结果以另一种格式显示。首先，等号被去掉了；这使得显示结果更适合其它程序解析，因为现在数据是以列格式显示。其次，上 面的sysObjectID和sysUpTime的格式也改变了。要显示故障排除信息，需要使用选项-D 。该选项可以显示包信息，包括容量、目的地址，及包的十六进制和ASCII形式的映像。  
表13-3列出了操作选项，与显示选项的个数相同，也是8个。  
表13-3  通用的操作选项  
  选 项                    说 明  
   -c 设置时钟值  
   -m 指定要载入的MIB模块名单  
   -M 指定要在其中搜索MIB文件的目录名单  
   -p 使用指定端口与管理代理通信  
   -r 指定重试次数  
   -R 请求随机访问管理代理的MIB表  
   -t 指定各次重试的时间间隔  
   -v 指定协议版本  
   
缺 省状态下，我们可以在系统中的标准位置找到MIB数据对象。比如说，MIB数据对象system.sysContact.0通常就位于结构树 .iso.org.dod.internet.mgmt.mib-2 中。UCD工具支持随机访问MIB 。使用这种访问方式时，MIB数据对象system.sysContact可以以sysContact形式键入，不用再写上组名system了。要指定只 访问某个MIB数据对象时，因为同样的名字可能会出现多次，所以还需要指定MIB的名字，然后再加上数据对象的名字，就象这样，SNMPv2- MIB:sysContact.0 。要使管理代理允许随机访问，需要使用命令行选项 –R 。当要搜寻的某个数据对象不在标准位置时，或者是管理代理中有多个数据对象使用相同的名字时，这个选项最有用。SNMP管理代理在与任何软件工具通信时， 有时必须把MIB结构树的数字形式的标识符（例如1.3.6.）转化为名字形式的标识符（例 如.iso.org.dod.internet.mgmt.mib-2）。一般来讲，没有MIB文件时，UCD工具显示数字形式的MIB信息。这是因为这 些工具从管理代理处只接收到数字形式的信息；它们不知道如何把这些数字形式的标识符映射成相应的名字字符串。缺省状态下，查询命令只支持一套MIB的核心 定义。当涉及非标准的或厂商定义的MIB时，查询命令可用的只有数字形式的标识符。要解决这个问题，我们可以使用选项 –m 和 –M 。选项 –m 用于指定要载入管理代理的模块的名单，这些模块必须在UCD工具进行SNMP查询管理代理前载入。一次要指定多个模块时，各个模块名之间必须用冒号分隔。 一个MIB模块就是一个包含管理代理的MIB定义的文件。使用选项 –m ，我们可以提供正确的MIB模块，这样上面的输出结果就可以有名字形式的显示了。选项 –M 也很有用，这是因为我们可以指定一个管理代理搜寻MIB文件的目录名单。  
13.1.2  环境变量  
每种UCD工具都要用到几个环境变量， 它们用于为某个操作参数设置全局值，或者是为某个命令行选项设置快捷方式。这些环境变量包括：PREFIX、MIBS、MIBDIR、SUFFIX。变量 PREFIX可用于定义MIB数据对象标识符的前缀部分。这个变量的缺省值是 .iso.org.dod.internet.mgmt.mib-2 。如果定义了该变量，当UCD工具访问某个MIB数据对象时，变量值将加在数据对象名的前面。在使用一个非标准MIB的情况下，定义这个值会很有用。变量 MIBS和MIBDIR用于把附加的MIB模块载入管理代理。变量MIBDIR的功能和命令行选项 –M 的作用是完全相同的。因为它们可用于所有的UCD工具，所以使用起来都很方便。变量SUFFIX用于锁定命令行选项 –s ，该选项用于显示MIB数据对象路径的后缀部分（或最后一个元素）。  
13.1.3  snmpbulkwalk命令  
snmpbulkwalk 用于通过SNMPv2 的SNMP GET BULK请求命令与其它网络实体通信。与snmpwalk类似，snmpbulkwalk也要遍历整棵MIB结构树，直到访问了所有的节点或者有错误发 生。前面已经讨论过，BULK请求提供了一种比SNMPv1的GET请求更有效的传输大量数据的机制。举例说明明，假定我们要访问MIB-II的 interface组。使用SNMPv1的snmpwalk命令我们可以获得157个包的信息；而使用snmpbulkwalk命令时，获得同样的信息只 要4个包。这样就大大节省了网络带宽，将提高管理代理和管理员系统的性能。  
要遍历一个叫cisco-gw1的设备的整棵MIB结构树，需要使用如下命令：  
 snmpbulkwalk –v 2c cisco-gw1 public  
snmpbulkwalk 命令只用于使用SNMPv2协议的管理代理。这是因为在SNMPv2协议问世前，根本就没有GET BULK工具。如果你试图在一个SNMPv1管理代理系统中使用上面这条命令，这会产生一个错误。举例说明，下面这条命令要从设备remote-gw3的 MIB结构树的system组部分开始进行遍历。这个例子中，remote-gw3只支持SNMPv1协议。  
 snmpbulkwalk –v 2c remote-gw3 public system  
因为这个设备不理解SNMP的GET BULK请求，所以最终会超时，在snmpbulkwalk命令后将显示如下信息：  
 Timeout:No Response from remote-gw3  
这 里显示出超时错误，这是因为虽然向设备发送了请求，但设备中的管理代理不懂SNMP请求的格式。因此，设备没有响应这个请求包，所以导致超时。需要注意的 是，snmpbulkwalk命令需要指定两个SNMPv2名字（2p或2c）之一。如果你在使用snmpbulkwalk命令时没有指定使用哪个版本的 SNMPv2协议，就会有如下的错误信息显示：  
 snmpbulkwalk:Cannot send v1 PDU on V2 session  
13.1.4  snmpdelta命令  
snmpdelta 命令用于收集一个SNMP管理代理实体的MIB中整数值的变化情况。这条命令监控指定的整型（Integer）数据对象，它将显示该数据对象所有时间的变 化情况。在跟踪网络错误时，这条命令很有用。举例说明，你可以借助于MIB数据对象ifInDiscards来确定某个接口丢弃了多少个包。要监控设备 remote-gw的这个数据对象，需要使用如下命令：  
 snmpdelta –R remote-gw public ifInDiscards.1  
这条命令将产生如下的输出结果：  
if InDiscards.1 /1  sec:800  
if InDiscards.1 /1  sec:8353  
if InDiscards.1 /1  sec:449  
if InDiscards.1 /1  sec:3883  
if InDiscards.1 /1  sec:541  
if InDiscards.1 /1  sec:8316  
if InDiscards.1 /1  sec:4576  
if InDiscards.1 /1  sec:8763  
if InDiscards.1 /1  sec:9568  
这条命令支持许多个补充的命令行选项参数，还支持前面介绍过的通用选项。表13-4列出了可用的补充选项。  
表13-4  补充的snmpdelta命令行选项  
  选 项                    说 明  
   -f 读取配置文件  
   -k 在输出结果中显示秒（second）  
   -l 把设置写入文件  
   -m 显示获得过的最大值  
   -p 指定查询时间  
   -P 指定报告所有查询请求结果的时间  
   -s 显示timestamp信息  
   -S 把数据记录到日志文件中  
   -t 确定时间间隔  
   -T 产生表格形式的输出结果  
使用选项 –m 时，snmpdelta将显示从被查询的MIB数据对象处获得的最大值。一般情况下，snmpdelta将显示各次查询的值；当它获得一个新的最大值时，就把它显示在Max栏中。下面这条命令，  
snmpdelta –R remote-gw public –m inOutOctets.1  
将产生如下的输出结果：  
if OutOctets.1 /1  sec:25784  (Max:25784)  
if OutOctets.1 /1  sec:21287  (Max:25784)  
if OutOctets.1 /1  sec:2743  (Max:25784)  
if OutOctets.1 /1  sec:8611  (Max:25784)  
if OutOctets.1 /1  sec:4473  (Max:25784)  
if OutOctets.1 /1  sec:10939  (Max:25784)  
if OutOctets.1 /1  sec:1882  (Max:25784)  
if OutOctets.1 /1  sec:9258  (Max:25784)  
if OutOctets.1 /1  sec:22751  (Max:25784)  
if OutOctets.1 /1  sec:28615  (Max:28615)  
if OutOctets.1 /1  sec:18599  (Max:28615)  
if OutOctets.1 /1  sec:28459  (Max:28615)  
if OutOctets.1 /1  sec:2662  (Max:28615)  
需要注意，当snmpdelta获得一个新的最大值时，它将更新Max栏中相应的值。  
当监控大量的MIB数据对象或数据时，你也许会发现把输出信息存入一个日志文件可能会方便些。使用选项 –l 可以把数据以如下格式存入一个文件：  
 {device}-{MIB object}  
因此，命令  
 snmpdelta –R monet public –I ifOutOctets.1  
将把数据对象ifOutOctecs的数值变化记录到文件monet-ifOutOctets.1中。检查这个文件的内容，我们可以看到和正常显示时相同格式的数据。  
如 果你希望以一种更具结构性的格式显示snmpdelta的输出结果，可以使用选项 –T 。举例说明明，如果snmpdelta需要同时监控几个MIB数据对象，并把数据加入到一个电子表格或其它程序中，它可以显示如下表格格式的输出结果。于 是，相应的命令和输出结果如下：  
 snmpdelta –R remot – gw public – T ifInDiscards.1 ifInDiscards.2  
   if InDiscards.1 ifInDiscards.2  
   1592.00 9950.00  
   9136.00 2506.00  
3338.00 5.00  
6338.00 2624.00  
8665.00 9971.00  
5609.00 569.00.  
9282.00 7086.00  
9153.00 8374.00  
8653.00 8195.00  
9877.00 3827.00  
这里显示的是两个接口丢包的数目，它们以列状格式排列。  
13.1.5   snmpget命令  
snmpget 命令用于从一个管理代理实体获得信息。这条命令使用SNMP的GET请求，它要求一个或多个有效的数据对象名作参数，将返回指定数据对象相应的值。如果有 错误产生，将显示一条说明信息，它可以帮助管理人员查明故障原因。如果有一个或多个MIB数据对象的值无法获得，系统将只显示这些可以获取值的数据对象。 MIB数据对象的名字格式已经在variables中指定。这条命令的句法如下：  
snmpget [common arguments]MIB-object [MIB-object]  
要从某个Cisco路由器中获得MIB数据对象sysDescr和sysContact的值，需要使用如下命令：  
snmpget remote-gw public system.sysdescr.0 system.sysUptime.0  
该命令使用的路由器模型和配置决定了有如下形式的输出结果：  
system.sysDescr.0 = Cisco Internetwork Operating System Software   
IOS (tm) 3000 Software (IGS-INR-L), Version 11.0(17), RELEASE   
SOFTWARE (fcl) Copyright (c) 1986-1997 by Cisco Systems, Inc.    
Compiled Thu 04-Sep-97 14:17 by richv   
system.sysUpTime.0 = Timeticks: (134381144) 15 days, 13:16:51.44  
需 要注意的是，路由器包含一个较长的sysDescr字符串。当你要识别在系统中运行的模型及当前网络操作系统（IOS——internetwork operating system）的版本时，这一点会很有用。IOS是Cisco的系统软件，它为许多Cisco硬件设备提供了路由和处理的功能。在这个例子中，系统模型是 运行着版本11.0的3000 ，这个系统已经运转了15天。  
13.1.6  snmpgetnext命令  
snmpgetnext命令用 于通过SNMP的GET NEXT请求获得一个或多个MIB数据对象的信息。对于命令行中每一个指定的数据对象，snmpgetnext命令都要找到MIB结构树中的下一个数据对 象。当你知道要访问的MIB数据对象的确切结构时，使用这个工具可以很方便的获得一系列相关数据对象的信息。为了进一步说明这个问题，举一个例子，后面的 这条命令将获得数据对象sysContact.0后的下一个MIB数据对象的有关信息。如下的命令：  
 snmpgetnext 10.0.2.220 public system.sysContact.0  
将显示：  
system.sysName.0=remote-gw  
可以看到，数据对象sysName.0正好处于数据对象sysContact.0的下一个位置。  
13.1.7   snmpnetstat命令  
snmpnetstat 命令与UNIX的netstat命令相似，它们都提供一些与设备接口和路由表有关的基本信息。这个工具的优点就是它使得获得一个SNMP设备的接口信息更 加容易。比如说，其中包括的设备有路由器、交换器、网络监控探测器，及其它的一些支持标准SNMP MIB的设备。这条命令的功能非常强大；确定SNMP设备的接口信息需要查询许多个独立的MIB数据对象，或者是直接访问某个设备并运行复杂的系统命令。 与netstat命令相似，snmpnetstat支持许多控制显示的命令行选项，表13-5列出了可用的命令行选项。  
表13-5  snmpnetstat的命令行选项  
  选 项                 说 明  
   -a 显示所有的套接口连接的状态  
   -I 显示系统定义的所有接口的状态  
   -I 显示指定网络接口的信息  
   -o 以简短形式显示接口的状态  
   -n 显示数字形式的网络地址  
   -p 显示网络协议的统计信息  
   -r 显示路由信息  
   -s 显示总协议的网络统计信息  
要显示所有网络接口的配置信息，需要使用选项 –I 。下面的这个例子中，snmpnetstat命令用于查询一个叫cisco-gw3的本地路由器。  
snmpnetstat -i cisco-gw3 public  
Name   Mtu  Network   Address      lpkts  lerrs   Opkts  Oerrs  Oueue  
Ethern 1500  10.0.2     10.0 .2.1     13377  315 132503 501 115  
Serial 1500  135.111.81 135.111.81.2 431375 127 462082 816 50  
Serial\* 1500  none     none         1934 154465 57336 998 150  
   
我 们可以看到，输出结果应该不陌生，与UNIX的netstat命令的结果相似。它们的主要区别只是接口的名字不同，以及这里去掉了表示接口冲突的一栏。在 接口名方面，Cisco软件使用接口类型和索引号作为全接口名（例如Serial）。不过，在这个例子中，接口名都被截短了，只包含接口类型信息，它们并 不是实际的Cisco接口名。所以，使用snmpnetstat命令时，输出的并不是全接口名，其中的索引号部分没有显示出来。  
要以一种简短的显示格式列出所有可用的接口，需要使用选项 –o 。从下面的输出结果也可以看出，这里只列出了输入字节和输出字节两列。可以把这个显示结果与使用选项 –i 的输出结果做一个比较。  
snmpnetstat -o cisco-gw4 public  
 Name Network Address           loctets       0octets  
 Ethern 10.0.2 10.0.2.1 487708 12778317  
 Serial     135.111.81 135.111.81.2 4331197 559999  
设 备cisco-gw4只包含了两个接口：一个是Ethernet，另一个是Serial 。要单独显示一个接口，可以使用选项 –I ，在后面再加一个接口名。这个选项可与interval一起使用。当使用interval参数启动snmpnetstat命令时，它将显示指定接口的持续 变化的网络统计数据。显示结果中有一列是指定接口的信息，还有一列是其它所有接口的汇总信息。输出结果的第一行表示的是设备重启以来的汇总信息。其余各行 中的数值随着指定的间隔时间不断变化。下面这条命令  
snmpnetstat -I Ethernet cisco-gw4 public 10  
input   (Ether      output             input (Total)        output  
 packets errs packets errs colls packets errs packets errs colls  
 68355 39800 131733 198 0 499131 3924  648945  730 0  
 178 62 93 68 0 376 255 236 22 0  
 46 58 142 84 0 172 167 268 42 0  
 93 63 67 60 0 210 13-4 359 14 0  
 119 49 169 85 0 326 187 385 18 0  
显 示了cisco-gw3中的Ethernet接口的有关包活动性信息，这些数据随时间而变化。该命令将持续显示输出结果，直到键入 control-c （^C）才可以停下。前两列代表Ethernet接口的输入包和输入错误；接下来的两列代表输出包和输出错误。第五列表示的是冲突的数目。其余的四列是设 备中定义的所有接口的汇总信息。  
如果要显示这个设备的路由信息，需要使用选项 –r ，就象下面这个例子：  
 snmpnetstat -r cisco-gw3 public  
 Routing tables  
 Destination Gateway Flags Interface  
 default 161.135.59.1 UG if0  
 155.161.75/25 161.135.59.9 U Serial0  
 155.161.114.128/26 rembrandt U Ethernet0  
 161.135                    161.135.59.1 UG if0  
 161.135.59/26 161.135.59.9 U Serial0  
 161.135.59.64/26 161.135.59.8 UG if0  
 161.135.59.128/26 rembrandt U Ethernet0  
 170.5 161.135.59.1 UG if0  
   
使用这条命令也可以获得每个网络协议的统计信息，这时需要使用选项–s 和选项–p 。下面这个例子中只显示了有关IP协议的数据：  
snmpnetstat -s -P ip monet public   
ip:  
39787 total datagrams received  
188 datagrams with header errors  
148 datagrams with an invalid destination address  
105 datagrams forwarded  
248 datagrams with unknown protocol  
180 datagrams discarded  
39942 datagrams delivered  
39626 output datagram requests  
208 output datagrams discarded  
256 datagrams with no route  
236 fragments received  
277 datagrams reassembled  
305 reassembly failures  
330 datagrams fragmented  
239 fragmentation failures  
225 fragments created  
13.1.8  snmpset命令  
snmpset 命令是UCD软件包中最有用也是功能最强的一条命令。本章介绍的多数工具只涉及到从一个SNMP管理代理出获得数据对象的值，而这个工具可用于修改那些可 改动的MIB管理代理数据对象的值。改动一个MIB数据对象是很有意义的，因为这意味着改动一个管理代理的配置或操作状态。举例说明，假定某个有几个接口 的路由器用作重要的远距离办公网之间的远程连接点。对该路由器中的一个或多个接口执行一条snmpset命令时，它可以影响到本地网络和远程的办公室之间 的连接性。因为snmpset命令的功能非常强大，所以在一个活动的网络中使用该命令时要特别小心。snmpset命令的基本句法如下：  
snmpset [common arguments] MIB-objectID type value [MIB –objectID type value]  
MIB-objectID是一个要指定新值的MIB数据对象。参数type表示要修改的数据对象的类型；value表示该数据对象应该设置的新值。type是一个字符，它表示数据对象的类型，这些类型是ASN.1定义的。表13-6列出了snmpset支持的类型。  
表13-6  snmpset的数据对象类型  
  字 符            数据对象类型  
i INTEGER  
s STRING  
x HEXADECIMAL STRING  
d DECIMAL STRING  
n NULL OBJECT  
o OBJECTID  
t TIMESTICKS  
a IPADDRESS  
平时我们不能随便使用snmpset命令，只在某些情况下才可以使用它。下面列出这些情况：禁止或激活某个网络接口、使用新的管理信息（比如说sysContact）更新设备、重新设置某些计数器、重启某个设备或管理代理、修改某些配置参数。  
13.1.9  snmpstatus命令  
snmpstatus命令可以从使用snmpget命令的SNMP网络实体处获得重要信息。该命令的句法如下：  
snmpstatus host community  
当 远程的管理代理使用一般的分区名public时，这条SNMP命令可能不会接收它，因为public用于缺省状态。对某个设备使用snmpstatus命 令时，将显示如下信息：设备的IP地址、MIB数据对象sysDescr、MIB数据对象sysUpTime、所有设备接口接收和发送的包的总数（即 ifInUcastPkts\* 与ifInNUCastPkts\*之和）、接收和发送的IP包的总数（即ipInReceives.0 与ipOutRequests.0之和）、活动的接口数、禁止的接口数。因此对一个叫switch-2200的设备运行这条命令：  
snmpstatus switch-2200 private  
将会得到输出结果：  
[10.0.2.2401->[Model: LinkSwitch 2000, h/w rev: 02-03, s/w rev: 07-   
01-00-00] Up: 90:23:02  
Interfaces: 17, Recv/Trans packets: 56034/59048 | IP: 39025/47234   
8 interfaces are down!  
使 用这个工具很方便，在不了解某个设备或指定的要查询的MIB数据对象时，我们使用这个工具可以很快获得一个网络设备当前的状态。可以看到，这里的设备是运 行着7.0版本的交换软件的LinkSwitch 2000（3Com公司制造）。这个设备已经运行了90小时23分钟。注意，这里snmpstatus已经探测到该设备有8个接口关闭了。不过，这种情况 下这并不算什么问题，因为这个设备是一个Ethernet交换器，它的接口连接在个人计算机或工作站上。因此，有一些用户经常重启它们的计算机，当它们每 晚离开办公室时，又关掉系统的电源。这就使得某些交换器端口看起来关闭了，这也是为什么snmpstatus的输出结果如此显示的原因。  
13.1.10  snmptable命令  
snmptable命令提供了通过SNMP的GETNEXT请求获得完整MIB表格的信息的功能。这条命令的句法如下：  
snmptable [common options][additional options] host community tableID  
该 命令必要的参数包括host、community、tableID 。tableID必须是一张真实的SNMP表格，例如interfaces.ifTable 。ifTable表格中包含一系列MIB数据对象，它们表示某个设备的接口的性能和特性信息。MIB-II中定义了一些表格，可以使用这条命令来显示它 们。这些表格是：  
The Interface Table (interfaces.ifTable)  
The IP Address Table (ip.ipAddrTable)  
The IP Routing Table (ip.ipRouteTable)  
The IP Net Media Table (ip.ipNetToMediaTable)  
The TCP Connection Table (tcp.tcpConnTable)  
The UDP Listener Table (udp.udpTable)  
snmptable命令的主要功能是用户可以使用该命令来显示SNMP表格，并把某些重要的表格数据输入到其它程序中用于报告和控制。举例说明，要显示一张有关TCP连接的表格tcpConnTable ，需要使用如下命令：  
monet# snmptable 10.0.2.240 public tcp.tcpConnTable  
SNMP table: tcp.tcpConnTable  
tcpConnState tcpConnLocalAddress tcpConnLocalPort tcpConnRemAddress tcpConnRemPort  
 listen 0.0.0.0 23 0.0.0.0 0  
 listen 0.0.0.0 111 0.0.0.0 0  
 listen 0.0.0.0 513 0.0.0.0 0  
 established 10.0.2.240 23 10.0.2.75 33441  
必须注意的是，这条命令要求一张有效的SNMP表格。因此，如果在命令行中提供的是一个非表格的数据对象，将有错误产生。下面就是这样一个例子：  
monet # snmptable 10.0.2.240 public system.sysContact.0  
was that a table? System.sysContact.0.1.1  
这 里我们在snmptable的命令行中提供了一个MIB数据对象system.sysContact.0（它不是一张MIB表格）。可以看到，这里显示了 一条错误信息，其中包括该表格的第一个元素（即 .1.1）。这条命令还支持其它一些可控制输出格式的命令行选项。表13-7列出了这些选项。  
表13-7  补充的snmptable命令行选项  
   选 项                    说 明  
    -b 显示一个简洁的报头  
    -f 指定字符分隔符  
    -h 不显示报头信息  
    -H 只显示表格的报头信息  
    -w 指定将要显示的表格的宽度  
    -x 为列出的每个条目分配MIB索引  
   
最有用的选项之一是选项 –f ，可用于指定列分隔字符。所以，在显示上面那个TCP表格时，如果要使用冒号作为域分隔符，需要使用如下命令：  
monet# snmptable 10.0.2.240 public -f : tcp.tcpConnTable  
SNMP table: tcp.tcpConnTable  
tcpConnState:tcpConnLocalAddress:tcpConnLocalPort:tcpConnRemAddress:t  
cpConnRemPort  
listen:0.0.0.0:23:0.0.0.0:0  
listen:0.0.0.0:111:0.0.0.0:0  
listen:0.0.0.0:513:0.0.0.0:0  
established:10.0.2.240:23:10.0.2.75:33441  
   
现 在，显示结果中的每一列都被字符 ：分隔开了，于是很容易把这个信息作为其它程序的输入。选项 –b 可用于提供更加简洁的报头信息，就象下面列出的输出信息那样。这里的表格at.atTable中包含IP地址和物理地址的地址转换映射信息。这个例子中列 出了三列，包括IfIndex、PhysAddress和NetAddress。其中IfIndex显示的是接口表格中的索引，它说明这里的映射对应表格 中的哪些接口。PhysAddress是数据链路地址（这个例子中是Ethernet），NetAddress是与网络中各个设备有关的IP地址。  
snmptable cisco-gw5 public -b at.atTable  
SNMP table: at.atTable  
 lflndex    PhysAddress NetAddress  
 2 "08 00 20  8D  81  82" 10.0.1.100  
 2 "00 60 47  1F  76  8A" 10.0.1.120  
 14 "00 60 08  91  4A  64" 10.0.2.110  
 14 "00 60 95  AD  F2  1F" 10.0.2.111  
 14 "00 80 5F  E2  8B  30" 10.0.2,117  
 14 "00 60 08  94  37  FC" 10.0.2.200  
 14 "08 00 20  86  2F  C2" 10-0.2.221  
 14 "00 60 08  62  C7  3E" 10.0.11.226  
 14 "08 00 20  7A  CA  49" 10.0.2.234  
 14 "08 00 20  25  70  E7" 10.0.2.250  
 14 "00 20 AF  CA  E6  99" 10.0.2.251  
 14 "00 60 08  1D  FD  40" 10.0.2.252  
 14 "00 80 A3  03  4E  9B" 10.0.2.257  
13.1.11  snmptest命令  
snmptest 命令提供了一种简单的类似shell的功能，它使得管理代理与另一个使用SNMP协议的网络实体间的通信更加容易。不象其它大多数的UCD工 具，snmptest命令是交互式的，用提示输入信息的方式来完成任务。该命令支持三中操作模式：get ，getnext和set 。使用 $N 命令就把snmptest命令置成GETNEXT模式；而使用 $S 命令就置成SET模式。如果要返回GET模式，需要使用 $G 命令。缺省状态下，snmptest处于GET模式，此时它提示用户指定一个MIB数据对象，于是该命令将从某个SNMP设备中获得这个数据对象的信息。  
13.1.12  snmptranslate命令  
snmptranslate 命令用于把SNMP的MIB数据对象信息转换成可读的文本形式。当对一个MIB数据对象运行这条命令时，它把数据对象信息或者转换成SMI值，或者转换成 符号形式。如果没有特别指定选项时，该命令就显示SMI值，这是缺省形式。这个工具支持几个命令行选项，已在表13-8中列出。该命令的主要作用就是在不 读取MIB定义文件的情况下显示MIB数据对象的所有特征。  
表13-8  snmptranslate的命令行选项  
  选 项                  说 明  
    -d 显示数据对象的说明信息  
    -p 显示载入的MIB文件中的符号表格  
    -n 以符号格式显示数据对象  
    -s 显示OID的最后一部分符号  
    -R 在访问数据对象时，使用随机访问模式  
    -w 当符号冲突时，显示警告信息  
    -W 显示比使用选项-w时更详细的警告信息  
   
在使用不带任何命令参数的snmptranslate命令时，显示结果是带点的数字形式的数据对象标识符。这种情况下，使用下面这条命令转换MIB数据对象system.sysDescr：  
 snmptranslate system.sysDescr  
结果是：  
 .1.3.6.1.2.1.1.1  
要显示一个MIB数据对象稍完整一点的说明信息，需要使用选项 –d 。下面举例说明明，请看这条命令：  
snmptranslate –d system.sysDescr  
在执行命令后，将有如下的输出结果：  
.1.3.6.1.2.1.1.1  
SYNTAX OCTET STRING  
DISPLAY-HINT ''255a''  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION ''A textual description of the entity. This value  
should include the full name and version identification of the   
system's hardware type, software operating-system、and networking   
software..''  
13.1.13  snmptrap命令  
snmptrap 命令将向某个指定的SNMP管理员发送一条SNMP trap报文。把这条命令插入一个shell程序或其它程序中，可用于向任何数量的SNMP网络管理员发送trap 。在该命令的命令行中可以指定一个或多个数据对象的标识符，同时必须有每个数据对象的类型及相应的值。snmptrap命令支持SNMPv1和 SNMPv2两种格式。该命令的基本句法如下：  
snmptrap –v 1 [command arguments]enterprise-oid agent generic-trap  
specific-trap uptime[object ID type value]  
其 中enterprise-oid域标识了产生trap的网络管理子系统。agent是发送trap的主机。generic-trap相应于预先定义的一个 SNMP trap 。specific-trap值标明了trap更具体的一些属性。uptime域表示设备上次初始化到发送trap之间的timestamp 。另外，object ID、tupe、value等域作为相关trap的附加信息。这些附加域和变量绑定在一起，其中可以包含与trap相关的任何类型的信息。 enterprise-oid、agent、uptime域不需要在命令行中指定。可以使用几个连续的空字符“ ”来指定这些域为缺省值。其中enterprise-oid域的缺省值是1.3.6.1.4.1.3.1.1（或者是 enterprises.cmu.1.1）。缺省的agent值是运行snmptrap命令的机器的主机名。uptime域的值从本地系统的MIB数据对 象system.sysUpTime.0中获得。  
举一个例子，假定我们要向一个叫rembrandt的网络管理系统发送一条link-down trap报文。进一步考虑，我们要通知它某个端口已经关闭，可以在该trap的绑定变量中指定端口信息。将用到下面这条命令：  
 snmptrap –v l monet public '' monet 20 '' interfaces.iftable.ifentry./  
 ifindex.l i l  
在 这个例子中，对于enterprise-oid和uptime域，我们使用的是缺省值。同时，我们指定了具体的接口（ifindex.1），并把它的值设 置为1（这说明它是该设备的第二个接口）。2 表示这里是一个link-down trap ，而 0 说明为specific-trap域提供的值是0 。如果我们再看一下rembrandt接收到的trap ，可以获得以下的输出结果：  
1998 –11-27 17:48:45 monet [10.0.3.126] enterprises.3.1.1:  
  Link Down Trap(0) Uptime: 1:29:06  
  Interfaces.ifTable.ifEntry.ifInde.1=1  
实 际上，大多数的link-down报文都没有实际意义。然而，当它们从一个关键设备（例如核心的路由器和交换器）处传来时，某个关闭的接口很可能会导致网 络故障。当你在配置网络设备发送的trap时，一定要非常注意，因为激活trap可以引起“trap洪灾”（trap flood）。trap洪灾是指网络中有大量的trap被发送，这是由于某个端口以每秒数次的频率在 开/关 状态下振荡而引起的。多数情况下，这表示发送trap的设备发生了故障。不过，只要稍微注意，就能避免这种问题。  
13.1.14  snmptrapd命令  
snmptrapd 命令用于接受和记录SNMP trap 。从端口162处发送的trap或者被UNIX的syslog工具记录下来，或者显示在终端上。这些报文在发送时使用LOG-WARING记录级；如果可 以接收到的话，它们将被发送给LOG\_LOCAL记录级。只有超级用户才可以运行这条命令，因为该命令要从一个保留的端口处接收信息。在执行这条命令时， 如果没有使用任何选项，这将使该命令在后台运行，将与调用它的shell分离。snmptrap命令支持几个命令行选项参数，包括 -P 、-D 、-d 、-q 。选项 –P 将通知snmptrapd在标准输出上显示接收到的任何trap ；选项 –d 将显示trap包的映像。因此，要显示接收到的trap和trap包，需要使用如下的命令：  
snmptrapd –p -d  
如果还是使用前面那个例子中的主机rembrandt ，运行这条命令的输出结果如下：  
 1998-11-27 22:56:47 UCD-snmp version 3.5  
 received 69 bytes from 10.0.3.126:-32566:  
 0000: 30 82  00 41 02 01 00 04  06 70 75 62  6C 69 63  A4 O..A public.  
 0016: 82 00 32 06 08 2B 06 01  04 01 03 01  01 40 04  OA.2 +    @..  
 0032: 00 03  7E 02 01 02 02 01  00 43 03 24  5C 96 30 8    2 ~C,$%.O.  
 0048: 00 13 30 82  00 0F 06 0A  2B 06 01 02 01 02 02 01       .0 +  
 0064: 01 01  02 01  01  
 1998-11-27 22:56:51 monet [10.0.3.126] enterprises.3.1.1:  
   Link Down Trap (0) Uptime: 6:37:09   
  interfaces.ifTable.ifEntry.iflndex.1 =1  
第 一行显示了版本号及snmptrapd进程是何时开始运行的。接下来输出结果显示了trap包的字节数、发送trap的主机名（monet）及UDP端口 （这个例子中是32566）。trap的内容有十六进制形式和ASCII形式两种。最后显示的trap信息，其中还包括绑定变量的信息。选项 –D 将显示附加的故障排除信息，其中包括MIB文件的解析信息。用户手册中介绍选项使用选项 –q 可以显示更详细的输出结果，不过，好象这个版本的命令中没有实现该功能。  
13.1.15  snmpwalk命令  
snmpwalk命令通过SNMP的GETNEXT请求，可以遍历管理代理MIB结构树。在命令行中可以使用一个数据对象参数来指定从MIB的哪一部分开始遍历。如果没有指定数据对象参数，snmpwalk将自动搜索，从MIB-II的数据对象开始遍历。所以，这条命令  
snmpwalk monet public  
将 遍历设备monet的整个系统管理代理的MIB 。既然MIB-II包含了大量的数据对象，随意这条命令将产生大量的输出结果。我们可以限制该命令的显示，不再显示管理代理支持的所有的数据对象，而只显 示其中的一个组的管理代理。这样，我们可以使用下面这条命令显示MIB-II的system组的数据对象：  
snmpwalk cisco-gw5 public system  
system.sysDescr.0 - Cisco Internetwork Operating System Software   
IOS (tm) GS Software (RSP-JV-M), Version 11.1(13a)CAl. EARLY   
DEPLOYMENT RELEASE SOFTWARE (fcl)  
Synced to mainline version: 11.1(13a)   
Copyright (c) 1986-1997 by cisco Systems, Inc.    
Compiled Wed 13-Aug-97 04:12 by richardd   
Systep.sysObjectID.0 = 0ID: enterprises.9.1.46  
system.sysUpTime.0 =Timeticks: (236153209) 27 days. 7:58:52.09  
systein.sysContact.0 = Steve Maxwell  
system.sysName.0 = cisco-gw4  
system.sysLocation.0 = Testing Lab  
system.sysServices.0 = 78  
如果你需要遍历指定的管理代理的整个MIB，并把输出结果存到一个文件中，可以使用如下命令：  
snmpwalk cisco-gw5 public.1>walk.out  
上 面这条命令使用 .1 作为snmpwalk的起始位置。这样做将保证每一个数据对象都能显示出来，因为 .1 是整棵MIB结构树的根，从这个起始点开始可以访问所有的数据处理对象。遍历整棵MIB结构树可以帮助我们找到某个指定的MIB数据对象，或者是知道指定 的管理代理到底支持多少个数据对象。要想知道cisco-gw5大约支持多少个数据对象，只需数一下文件中有多少行就可以了。因为snmpwalk显示每 个MIB数据对象需要一行（除非显示内容超过80个字符），所以我们可以使用UNIX的wc命令来统计文件walk.out一共有多少行。  
这条命令  
 wc-1 walk.out  
的输出结果是：  
2242 walk.out  
我 们可以看到，这个命令的输出结果说明设备cisco-gw5包含的管理代理支持2242个MIB数据对象。如果在snmpwalk的命令行中没有特别指定 任何数据对象，该命令将搜索MIB-II数据对象，这是缺省状态。当snmpwalk搜索超出管理代理的MIB终点时，将显示消息“End of MIB” 。举例说明，当管理代理并不支持命令行中指定的MIB数据对象组时，这种现象就可能发生。  
13.2  snmpconf命令  
snmpconf 命令提供了配置设备的功能，它使用snmpset命令对配置文件中定义的MIB数据对象进行操作。配置文件中可能包含一个MIB数据对象的清单和应该为 SNMP设备设置的数据对象值。这个工具提供了一种自动机制，可用于实现大量设备的标准配置，或者是配置一个必须进行多次相同配置的设备。使用该方法，节 省了安装一个设备所需要的时间，并且出现配置错误的可能性也降低了。配置文件可以包含MIB数据对象条目、注释，及空行等。加注释时，需要在每一行开头写 一个 # 号。一个标准的数据对象条目由类型、OID、数据对象值、注释字符串构成。下面就是一个配置文件实例：  
#  
# Setup standard system group information  
#  
S.1.3.6.1.2.1.1.4.0 Steve-Maxwel l   system contact  
S.1.3.6.1.2.1.1.5.0 host.dt.com      system name  
S.1.3.6.1.2.1.1.6.0 Graphics\_Lab    system location  
   
这 个例子有三个配置条目，每一个在第一个域中都包含字符 S ，它标明了第二个域中的MIB数据对象的类型。这里 S 意味着数据对象的类型是STRING 。这个域被认为是一个字符串，因此，其中一定不能包含任何额外的空格。所以，Virginaia-Maxwell是有效的，而Virginia Maxwell就不能放在这个域中。而且，这里也不允许使用双引号。如果使用了“Virginia Maxwell” ，这将会导致一个配置错误。snmpconf工具定义了两个补充的类型：I代表INTEGER ，A代表IPADDRESS 。第二个域是MIB数据对象，这些数据对象的值将被修改，指定为第三个域中的值。需要注意的是，snmpconf只支持完整的数字形式的数据对象标识符。 而另一种格式，即名字形式的数据对象标识符（例如iso.org.dod.internet.mgmt.mib- 2.system.sysLoation.0）或缩写的MIB字符串都会导致配置错误。已经提到了，第三个域是MIB数据对象将要被设置的值。因此，它必 须与第一个域中指定的类型相匹配。所以在这个例子中，每一个配置行要求这个域是一个字符串。第四个也是最后一个域是为注释准备的。它提供的注释内容应该能 用于标识MIB数据对象及其值。这个例子中，MIB数据对象是MIB-II中system组的contact（sysContact）、 name（sysName）、location（sysLocation）。  
只要配置文件包含了 需要/要求 的MIB数据对象及其值，我们就可以把它用于配置SNMP设备。snmpconf命令的基本句法包括：  
snmpconf hostname community configuration\_file  
现在假定上面列出的配置存储在一个叫config1的文件中，我们希望要用这个文件来配置设备nicodemus ，它也有一个分区名private 。所以，需要使用如下命令：  
snmpconf nicodemus private configl  
在执行这条命令之后，将得到如下的输出结果：  
#  
# Setup standard system group information  
#  
Set display string′system contact′to Steve-Maxwell  
Set display string′system name′to host.dt.com  
Set display string′system location′to Graphics\_Lab  
Done.  
我们可以看到，定义的注释内容与set命令成功执行的确认信息同时显示出来了。如果你想检查执行的结果，可以使用snmpwalk命令查询MIB-II的system组。于是，这条命令  
 snmpwalk nicodemus public system  
将产生以下的输出结果：  
system.sysDescr.0 = unknown  
system.sysObjectID.0 = 0ID: enterprises.ucdavis.ucdSnmpAgent.solaris system.sysUpTime.0 = Timeticks: (260363) 0:43:23.63   
system.sysContact.0 = Steve-Maxwell   
system.sysName.0 = host.dt.com   
system.sysLocation.0 = Graphics-Lab   
system.sysServices.0 = 72  
主机nicodemus正在运行UCD的SNMP管理代理，上面只列出了system组的一部分。这个例子中，我们可以确定数据对象sysContact、sysName、sysLocation已经按照snmpconf的配置文件更新过了。  
snmpconf 软件提供了几种方式来指定需要的配置文件。首先，象前面的例子那样在命令行中指定文件名，snmpconf将在当前目录中寻找这个文件。其次，在命令行中 也可以给出文件的全路径名。最后，如果我们设置了环境变量SNMPCONF\_LIB，snmpconf将按照这个变量指定的路径寻找配置文件。  
snmpconf还提供了一些基本的错误检查功能，在有问题发生时，它将显示一些有用的信息。不过，我们很难确定特定类型的错误发生的原因。举例说明，使用了错误的分区名，或指定的设备现在不在网上，这两种情况下显示的错误信息完全相同。所以，这两条命令  
 snmpconf didymus private configl  
 snmpconf monet public confiql  
虽然各自的原因不同，产生同样的错误信息，如下所示：  
#  
# Setup standard system group information  
#  
UNABLE to set display string 'system contact' to Steve-Maxwell  
UNABLE to set display string 'system name' to host.dt.m  
UNABLE to set display string 'system location' to Graphics\_Lab  
Done.  
3 errors encountered.  
   
第 一个命令的目的是设置MIB数据对象didymus，现在从网上无法访问该数据对象。这是说，在执行这条命令时，设备的电源关掉了。第二条命令使用了一个 错误的分区名。这个例子中，正确的分区名应该是private ，不是public 。我们可以看到，对于关闭的系统和错误的分区名，snmpconf的响应是相同的，这就使得你确定错误的原因非常困难。不过，只要你小心一点，完全可以绕 过这个问题，只要在执行snmpconf之前运行ping命令就可以了。所以，你可以避免这种多义性，可以分别解决各个问题。  
snmpconf软件提供了两个命令行选项。第一个是 –version ，它用于显示snmpconf工具及CMU支持库的版本信息。第二个是 –quiet ，它通知snmpconf在执行过程中将不显示任何信息。