

Chapter 16

Network Monitoring

This chapter introduces the user to various techniques for monitoring network performance and operation.

Concepts Learned in this Chapter

- Remote network device monitoring
- Traffic monitoring
- GUI Network Status

Table of Contents

Network Monitoring.....	1
16.1 Simple Network Monitoring Protocol.....	3
16.1.1 Software Installation and Activation.....	3
16.1.2 net-snmp Configuration.....	4
16.1.2.1 Script of snmpd.conf File Creation.....	5
16.1.2.2 snmpd.conf Example.....	9
16.1.2.3 Script of snmp.conf File Creation.....	12
16.1.2.4 snmp.conf output file.....	23
16.1.3 MIB Files.....	25
16.1.4 MIB Assignment	26
16.1.5 SNMP Commands	28
16.1.5.1 snmptranslate.....	29
16.1.5.2 snmpget.....	31
16.1.5.3 snmpgetnext.....	32
16.1.5.4 snmpwalk.....	33
16.1.5.5 snmptable.....	34
16.1.5.6 snmpset.....	36
16.1.5.7 snmptrap.....	37
16.1.6 MIB Files.....	38
16.1.6.1 BEGIN.....	38
16.1.6.2 IMPORTS.....	39
16.1.6.3 MODULE IDENTITY.....	39
16.1.6.4 OBJECT IDENTIFIER.....	40
16.1.6.5 MIB Groups.....	40
16.1.6.6 OBJECT TYPES.....	40
16.1.1.7 Required Vendor MIB Files.....	41
16.1.1.8 MIB Initial Numbering.....	42
16.1.2 Making it work.....	43
16.1.2.1 Linux Host Agent (RH 9).....	43
16.1.2.2 Configuring Cisco Host Agent.....	47
16.1.2.3 snmp Data Collection from Cisco.....	48
16.2 snmp Analysis Programs	49
16.2.1 nino	49
16.2.2 Multi-Router Traffic Grapher.....	49
16.2.3 OpenNMS	50
16.X Commands Used in this Chapter.....	50
16.X Chapter Review Questions.....	50

16.1 Simple Network Monitoring Protocol

Simple Network Monitoring Protocol, or **snmp**, allows for a monitoring station to query or set information on a remote host, which is configured as a snmp server. This data may then be stored or utilized for evaluation network performance. The host in general will typically be a router or server, but other devices may also be monitored.

The items that may be monitored are dependent upon the vendor and the **Management Information Base**, or **MIB**, definition. A specific query may be made against the device, requesting the status of a specific **Object Identifier**, or **OID**. The vendor must support a given option to be monitored, and the mib must be written for it.

There are two parts to the software for operating snmp. In order for a host to be monitored, it must have an **snmp agent**. This software responds to queries and configuration. The second software application that is required is the **snmp monitor**. This software issues a query to a host for information. For our initial discussion, we will set up an snmp monitor for observing various Cisco equipment, which is pre-configured with an agent.

After learning this procedure, the user will learn that setting up and using snmp queries on the fly is not an easy task. It is best that a script file exists which contains all of the queries that are desired, and that they be run on a periodic basis.

16.1.1 Software Installation and Activation

The monitor software that will be used in this discussion is **net-snmp**, which may be downloaded from the site <http://net-snmp.org>. Installation instructions are in Appendix H4. It is assumed at this point that the initial installation has been completed. In order to make the software operational, the following commands need to be issued:

```
# chkconfig --list | grep snmp
snmpd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
snmptrapd 0:off 1:off 2:off 3:off 4:off 5:off 6:off
# chkconfig snmpd on
# chkconfig snmptrapd on
# chkconfig --list | grep snmp
snmpd          0:off 1:off 2:off 3:on 4:on 5:on 6:off
snmptrapd 0:off 1:off 2:off 3:on 4:on 5:on 6:off
# service snmpd restart
# service snmptrapd restart
```

The snmpd daemon is the server software that allows a monitor to query an agent. When configured, an agent is capable of transmitting a packet to notify the agent that something is wrong. In this case, it sends a **trap**. The monitor is then able to receive the trap signal and issue appropriate alerts. In order for the monitor to act on the trap, the **snmptrapd** daemon must be activated.

16.1.2 net-snmp Configuration

In order to have an operational system, net-snmp must be configured. This is a multi-step process requiring the creation of at least two of the three possible configuration files. The two required files are **snmp.conf** and **snmpd.conf**, the third file, **snmptrapd.conf** is required if the agent is to transmit system error or trap messages.

The very first time net-snmp is viewed, the configuration files do not exist. They are normally maintained in the **/etc/snmp** directory. There is a perl program available that assists with the creation of all three configuration files, called **snmpconf**. This program is maintained in the **/usr/bin/** directory. To run the program, we need to select several options:

- i Install the created files into the **/usr/share/snmp** directory.
- p Install the created files into the **~/.snmp** directory.
- r all Read in all of the found .conf files.
- h List all of the startup options.
- g Ask a series of questions from the specified script file.

If we have an existing configuration file in the **/etc/snmp** directory, we typically will want to read it in, we will therefore use the “**-r all**” option. If we know of a specific set of questions that need to be answered, we can specify a question dataset file by using the “**-g**” option. Finally, we need to create a new file, which will eventually be copied to the **/etc/snmp** directory, this requires either the “**-i**” or “**-p**”. The questions for the **snmpd.conf** file are provided by a data file, **/usr/share/snmp/snmpconf/snmpd.conf /basic_setup**. Issue the command:
snmpconf

When the **snmpconf** perl program is run by itself, it provides the options to create one of three files:

The following installed configuration files were found:

1: /usr/local/share/snmp/snmpd.conf

Would you like me to read them in? Their content will be merged with the output files created by this session.

Valid answer examples: "all", "none", "3", "1,2,5"

Read in which (default = all):

***I can create the following types of configuration files for you.
Select the file type you wish to create:
(you can create more than one as you run this program)***

***1: snmpd.conf
2: snmptrapd.conf***

3: *snmp.conf*

In order to configure the server agent, we first need to create the `snmpd.conf` file, so we would select “1”. To configure the monitor station, we need to select “3”. Additionally, if we wish the agent to transmit trap messages, then we would configure the `snmptrapd.conf` option, selecting “2”.

If we desire to observe the setup configuration without creating a new file automatically in the `/usr/share/snmp` directory, leave off the `-i` or `-p`, and you will be prompted as to whether you want to rename the new file, overwrite, append, or skip it. If you wish to view the questions without creating or modifying the file, use only the `-g` option. It is recommended that the created file be renamed as **`snmpd.conf.new`**. The file may then be reviewed prior to overwriting to the **`snmpd.conf`**. The following files show the creation process and an example configuration files for `snmpd.conf(.new)` file and `snmp.conf`.

16.1.2.1 Script of `snmpd.conf` File Creation

The following is the script that is used to create the `snmpd.conf` file. Responses are shown in bold. Default responses are typically taken unless otherwise shown (bold print).

Script started on Fri 02 Apr 2004 07:29:35 PM CST

*** Beginning basic system information setup ***

Do you want to configure the information returned in the system MIB group (contact info, etc)? (default = y):

Configuring: syslocation

Description:

The [typically physical] location of the system.

Note that setting this value here means that when trying to perform an snmp SET operation to the `sysLocation.0` variable will make the agent return the "notWritable" error code. IE, including this token in the `snmpd.conf` file will disable write access to the variable.

arguments: location_string

The location of the system: **ricepad**

Finished Output: syslocation ricepad

Configuring: syscontact

Description:

The contact information for the administrator

Note that setting this value here means that when trying to perform an snmp SET operation to the `sysContact.0` variable will make the agent return the "notWritable" error code. IE, including this token in the `snmpd.conf` file will disable write access to the variable.

arguments: contact_string

The contact information: **dennis**

Finished Output: syscontact dennis

Do you want to properly set the value of the `sysServices.0` OID (if you don't know, just say no)? (default = y):

Configuring: sysservices

Description:

The proper value for the sysServices object.
arguments: syservices_number

does this host offer physical services (eg, like a repeater) [answer 0 or 1]: **0**
 does this host offer datalink/subnetwork services (eg, like a bridge): **0**
 does this host offer internet services (eg, supports IP): **1**
 does this host offer end-to-end services (eg, supports TCP): **1**
 does this host offer application services (eg, supports SMTP): **1**

Finished Output: syservices 76

*** BEGINNING ACCESS CONTROL SETUP ***

Do you want to configure the agent's access control? (default = y):
 Do you want to allow SNMPv3 read-write user based access (default = y):

Configuring: rwuser

Description:

a SNMPv3 read-write user

arguments: user [noauth|auth|priv] [restriction_oid]

The SNMPv3 user that should have read-write access: **dennis**

The minimum security level required for that user [noauth|auth|priv, default = auth]:

The OID that this community should be restricted to [if appropriate]:

Finished Output: rwuser dennis

Do another rwuser line? (default = y): **n**

Do you want to allow SNMPv3 read-only user based access (default = y):

Configuring: rouser

Description:

a SNMPv3 read-only user

arguments: user [noauth|auth|priv] [restriction_oid]

Enter the SNMPv3 user that should have read-only access to the system: **drice**

The minimum security level required for that user [noauth|auth|priv, default = auth]:

The OID that this community should be restricted to [if appropriate]:

Finished Output: rouser drice

Do another rouser line? (default = y): **n**

Do you want to allow SNMPv1/v2c read-write community access (default = y):

Configuring: rwcommunity

Description:

a SNMPv1/SNMPv2c read-write access community name

arguments: community [default|hostname|network/bits] [oid]

Enter the community name to add read-write access for: **ricepad**

The hostname or network address to accept this community name from [RETURN for all]:

The OID that this community should be restricted to [RETURN for no-restriction]:

Finished Output: rwcommunity ricepad

Do another rwcommunity line? (default = y): **n**

Do you want to allow SNMPv1/v2c read-only community access (default = y):

Configuring: rocommunity

Description:

a SNMPv1/SNMPv2c read-only access community name

arguments: community [default|hostname|network/bits] [oid]

The community name to add read-only access for: **ricepad**

The hostname or network address to accept this community name from [RETURN for all]:

The OID that this community should be restricted to [RETURN for no-restriction]:

Finished Output: rocommunity ricepad
 Do another rocommunity line? (default = y): n

*** Beginning trap destination setup ***

Do you want to configure where and if the agent will send traps? (default = y):
 Do you want the agent to send snmp traps on snmp authentication failures? (default = y):

Configuring: authtrappable
 Description:

Should we send traps when authentication failures occur
 arguments: 1 | 2 (1 = yes, 2 = no)

Should traps be sent when authentication failures occur? (1=yes, 2=no): 1

Finished Output: authtrappable 1

Configuring: trapcommunity
 Description:

Default trap sink community to use
 arguments: community-string

The default community name to use when sending traps: **ricepad**

Finished Output: trapcommunity **ricepad**
 Do you want the agent to send snmpv2c informs to a trap receiver (default = y):

Configuring: informsink
 Description:

A SNMPv2c inform (acknowledged trap) receiver
 arguments: host [community] [portnum]

A host name that should receive the trap: **fried.ricepad.org**
 The community to be used in the trap sent [optional]: **ricepad**
 The port number the trap should be sent to [optional]:

Finished Output: informsink fried.ricepad.org ricepad
 Do another informsink line? (default = y): n
 Do you want the agent to send snmpv2c traps to a trap receiver (default = y):

Configuring: trap2sink
 Description:

A SNMPv2c trap receiver
 arguments: host [community] [portnum]

A host name that should receive the trap: **fried.ricepad.org**
 The community to be used in the trap sent [optional]:
 The port number the trap should be sent to [optional]:

Finished Output: trap2sink fried.ricepad.org
 Do another trap2sink line? (default = y): n
 Do you want the agent to send snmpv1 traps to a trap receiver (default = y):

Configuring: trapsink
 Description:

A SNMPv1 trap receiver
 arguments: host [community] [portnum]

A host name that should receive the trap: **fried.ricepad.org**
 The community to be used in the trap sent [optional]:
 The port number the trap should be sent to [optional]:

Finished Output: trapsink fried.ricepad.org
 Do another trapsink line? (default = y): n

```
*****
*** Beginning monitoring setup ***
*****
```

Do you want to configure the agent's ability to monitor various aspects of your system? (default = y):
 Do you want to configure the agents ability to monitor processes? (default = y):

Configuring: proc

Description:

Check for processes that should be running.
 proc NAME [MAX=0] [MIN=0]

NAME: the name of the process to check for. It must match exactly (ie, http will not find httpd processes).
 MAX: the maximum number allowed to be running. Defaults to 0.
 MIN: the minimum number to be running. Defaults to 0.

The results are reported in the prTable section of the UCD-SNMP-MIB tree
 Special Case: When the min and max numbers are both 0, it assumes you want a max of infinity and a min of 1.

Name of the process you want to check on: **cpuload**

Maximum number of processes named 'cpuload' that should be running [default = 0]:

Minimum number of processes named 'cpuload' that should be running [default = 0]:

Finished Output: proc cpuload

Do another proc line? (default = y): **n**

Do you want to configure the agents ability to monitor disk space? (default = y):

Configuring: disk

Description:

Check for disk space usage of a partition.
 The agent can check the amount of available disk space, and make sure it is above a set limit.

disk PATH [MIN=100000]

PATH: mount path to the disk in question.
 MIN: Disks with space below this value will have the Mib's errorFlag set.
 Can be a raw byte value or a percentage followed by the % symbol. Default value = 100000.

The results are reported in the diskTable section of the UCD-SNMP-MIB tree

Enter the mount point for the disk partition to be checked on: **/dev/hda3**

Enter the minimum amount of space that should be available on /dev/hda3: **10000000**

Finished Output: disk /dev/hda3 1000000

Do another disk line? (default = y): **n**

Do you want to configure the agents ability to monitor load average? (default = y):

Configuring: load

Description:

Check for unreasonable load average values.
 Watch the load average levels on the machine.

load [1MAX=12.0] [5MAX=12.0] [15MAX=12.0]

1MAX: If the 1 minute load average is above this limit at query time, the errorFlag will be set.
 5MAX: Similar, but for 5 min average.
 15MAX: Similar, but for 15 min average.

The results are reported in the laTable section of the UCD-SNMP-MIB tree

Enter the maximum allowable value for the 1 minute load average: **3**
 Enter the maximum allowable value for the 5 minute load average: **4**
 Enter the maximum allowable value for the 15 minute load average: **5**

Finished Output: load 3 4 5
 Do another load line? (default = y): **n**
 Do you want to configure the agents ability to monitor file sizes? (default = y):

Configuring: file

Description:

Check on the size of a file.

Display a files size statistics.

If it grows to be too large, report an error about it.

file /path/to/file [maxsize_in_bytes]

if maxsize is not specified, assume only size reporting is needed.

The results are reported in the fileTable section of the UCD-SNMP-MIB tree

Enter the path to the file you wish to monitor: **/home/dennis**
 Enter the maximum size (in bytes) allowable for /home/dennis: **5000000**

Finished Output: file /home/dennis 5000000
 Do another file line? (default = y): **n**

Error: An snmpd.conf file already exists in this directory.

'overwrite', 'skip', 'rename' or 'append'? : **r**

Save to what new file name instead (or 'skip')? **snmpd.conf.new**

The following files were created:

snmpd.conf.new installed in /usr/local/share/snmp

Obviously, the values that you enter are dependent upon your requirements.

16.1.2.2 snmpd.conf Example

The following is a copy of the snmpd.conf.new file that was created from the above script. All lines that start with a “#” are comments and are ignored.

```
#####
#
# snmpd.conf
#
# - created by the snmpconf configuration program
#
#####
# SECTION: System Information Setup
#
# This section defines some of the information reported in
# the "system" mib group in the mibII tree.

# syslocation: The [typically physical] location of the system.
# Note that setting this value here means that when trying to
# perform an snmp SET operation to the sysLocation.0 variable will make
# the agent return the "notWritable" error code. IE, including
# this token in the snmpd.conf file will disable write access to
# the variable.
# arguments: location_string

syslocation Unknown (edit /etc/snmp/snmpd.conf)
```

```

syslocation ricepad

# syscontact: The contact information for the administrator
# Note that setting this value here means that when trying to
# perform an snmp SET operation to the sysContact.0 variable will make
# the agent return the "notWritable" error code. IE, including
# this token in the snmpd.conf file will disable write access to
# the variable.
# arguments: contact_string

syscontact Root <root@localhost> (configure /etc/snmp/snmp.local.conf)
syscontact dennis

# syservices: The proper value for the sysServices object.
# arguments: syservices_number

syservices 76

#####
# SECTION: Access Control Setup
#
# This section defines who is allowed to talk to your running
# snmp agent.

# rwuser: a SNMPv3 read-write user
# arguments: user [noauth|auth|priv] [restriction_oid]

rwuser dennis

# rouser: a SNMPv3 read-only user
# arguments: user [noauth|auth|priv] [restriction_oid]

rouser drice

# rocommunity: a SNMPv1/SNMPv2c read-only access community name
# arguments: community [default|hostname|network/bits] [oid]

rocommunity ricepad

# rwcommunity: a SNMPv1/SNMPv2c read-write access community name
# arguments: community [default|hostname|network/bits] [oid]

rwcommunity ricepad

#####
# SECTION: Trap Destinations
#
# Here we define who the agent will send traps to.

# trapsink: A SNMPv1 trap receiver
# arguments: host [community] [portnum]

trapsink fried.ricepad.org

# trap2sink: A SNMPv2c trap receiver
# arguments: host [community] [portnum]

trap2sink fried.ricepad.org

# informsink: A SNMPv2c inform (acknowledged trap) receiver
# arguments: host [community] [portnum]

informsink fried.ricepad.org ricepad

```

```

# trapcommunity: Default trap sink community to use
# arguments: community-string

trapcommunity ricepad

# authtrapenable: Should we send traps when authentication failures occur
# arguments: 1 | 2 (1 = yes, 2 = no)

authtrapenable 1

#####
# SECTION: Monitor Various Aspects of the Running Host
#
# The following check up on various aspects of a host.

# proc: Check for processes that should be running.
# proc NAME [MAX=0] [MIN=0]
#
# NAME: the name of the process to check for. It must match
# exactly (ie, http will not find httpd processes).
# MAX: the maximum number allowed to be running. Defaults to 0.
# MIN: the minimum number to be running. Defaults to 0.
#
# The results are reported in the prTable section of the UCD-SNMP-MIB tree
# Special Case: When the min and max numbers are both 0, it assumes
# you want a max of infinity and a min of 1.

proc cpuload

# disk: Check for disk space usage of a partition.
# The agent can check the amount of available disk space, and make
# sure it is above a set limit.
#
# disk PATH [MIN=100000]
#
# PATH: mount path to the disk in question.
# MIN: Disks with space below this value will have the Mib's errorFlag set.
# Can be a raw byte value or a percentage followed by the %
# symbol. Default value = 100000.
#
# The results are reported in the dskTable section of the UCD-SNMP-MIB tree

disk /dev/hda3 10000000

# load: Check for unreasonable load average values.
# Watch the load average levels on the machine.
#
# load [1MAX=12.0] [5MAX=12.0] [15MAX=12.0]
#
# 1MAX: If the 1 minute load average is above this limit at query
# time, the errorFlag will be set.
# 5MAX: Similar, but for 5 min average.
# 15MAX: Similar, but for 15 min average.
#
# The results are reported in the laTable section of the UCD-SNMP-MIB tree

load 3 4 5

# file: Check on the size of a file.
# Display a file's size statistics.
# If it grows to be too large, report an error about it.
#
# file /path/to/file [maxsize_in_bytes]

```

```
#
#   if maxsize is not specified, assume only size reporting is needed.
#
#   The results are reported in the fileTable section of the UCD-SNMP-MIB tree

file /home/dennis 5000000

#####
# SECTION: Extending the Agent
#
#   You can extend the snmp agent to have it return information
#   that you yourself define.

# pass: Run a command that interpretes the request for an entire tree.
#   The pass program defined here will get called for all
#   requests below a certain point in the mib tree. It is then
#   responsible for returning the right data beyond that point.
#
#   arguments: miboid program
#
#   example: pass .1.3.6.1.4.1.2021.255 /path/to/local/passtest
#
#   See the snmpd.conf manual page for further information.
#
#   Consider using "pass_persist" for a performance increase.

pass .1.3.6.1.4.1.4413.4.1 /usr/bin/ucd5820stat

#
# Unknown directives read in from other files by snmpconf
#
com2sec notConfigUser default public
group notConfigGroup v1 notConfigUser
group notConfigGroup v2c notConfigUser
view systemview included .1.3.6.1.2.1.1
view systemview included .1.3.6.1.2.1.25.1.1
access notConfigGroup "" any noauth exact systemview none none
```

When the `snmpd.conf.new` file was created, it was stored in the `/usr/share/snmp` directory. Assuming it is what we wish to use, we need to copy it to the `/etc/snmp` directory. Issue the command:

```
cp /usr/share/snmp/snmpd.conf.new /etc/snmp/snmpd.conf
```

16.1.2.3 Script of `snmp.conf` File Creation

The following is a copy of a sample script used to create the `snmp.conf` file. .

The following installed configuration files were found:

```
1: /usr/local/share/snmp/snmpd.conf
```

Would you like me to read them in? Their content will be merged with the output files created by this session.

Valid answer examples: "all", "none", "3", "1,2,5"

Read in which (default = all):

I can create the following types of configuration files for you.
Select the file type you wish to create:
(you can create more than one as you run this program)

- 1: snmpd.conf
- 2: snmptrapd.conf
- 3: snmp.conf

Other options: quit

Select File: **3**

The configuration information which can be put into snmp.conf is divided into sections. Select a configuration section for snmp.conf that you wish to create:

- 1: Default Authentication Options
- 2: Debugging output options
- 3: Output style options
- 4: Textual mib parsing

Other options: finished

Select section: **1**

Section: Default Authentication Options

Description:

This section defines the default authentication information. Setting these up properly in your ~/.snmp/snmp.conf file will greatly reduce the amount of command line arguments you need to type (especially for snmpv3).

Select from:

- 1: The default port number to use
- 2: The default snmp version number to use.
- 3: The default snmpv1 and snmpv2c community name to use when needed.
- 4: The default snmpv3 security name to use when using snmpv3
- 5: The default snmpv3 context name to use
- 6: The default snmpv3 security level to use
- 7: The default snmpv3 authentication type name to use
- 8: The default snmpv3 authentication pass phrase to use
- 9: The default snmpv3 privacy (encryption) type name to use
- 10: The default snmpv3 privacy pass phrase to use

Other options: finished, list

Select section: **2**

Configuring: defversion

Description:

The default snmp version number to use.
override: with -v on the command line.
arguments: 1|2c|3

Enter the default snmp version number to use (1|2c|3): **2c**

Finished Output: defversion 2c

Section: Default Authentication Options

Description:

This section defines the default authentication information. Setting these up properly in your ~/.snmp/snmp.conf file will greatly reduce the amount of command line arguments you need to type (especially for snmpv3).

Select from:

- 1: The default port number to use
- 2: The default snmp version number to use.
- 3: The default snmpv1 and snmpv2c community name to use when needed.
- 4: The default snmpv3 security name to use when using snmpv3
- 5: The default snmpv3 context name to use
- 6: The default snmpv3 security level to use
- 7: The default snmpv3 authentication type name to use
- 8: The default snmpv3 authentication pass phrase to use
- 9: The default snmpv3 privacy (encryption) type name to use
- 10: The default snmpv3 privacy pass phrase to use

Other options: finished, list

Select section: **3**

Configuring: defcommunity

Description:

The default snmpv1 and snmpv2c community name to use when needed.

If this is specified, you don't need to include the community name as an argument to the snmp applications.

override: with -c on the command line.

arguments: communityname

Enter the default community name to use: **ricepad**

Finished Output: defcommunity ricepad

Section: Default Authentication Options

Description:

This section defines the default authentication information. Setting these up properly in your ~/.snmp/snmp.conf file will greatly reduce the amount of command line arguments you need to type (especially for snmpv3).

Select from:

- 1: The default port number to use
- 2: The default snmp version number to use.
- 3: The default snmpv1 and snmpv2c community name to use when needed.
- 4: The default snmpv3 security name to use when using snmpv3
- 5: The default snmpv3 context name to use
- 6: The default snmpv3 security level to use
- 7: The default snmpv3 authentication type name to use
- 8: The default snmpv3 authentication pass phrase to use
- 9: The default snmpv3 privacy (encryption) type name to use
- 10: The default snmpv3 privacy pass phrase to use

Other options: finished, list

Select section: **list**

Lines defined for section "Default Authentication Options" so far:

defaultport

defversion 2c

defcommunity ricepad

Section: Default Authentication Options

Description:

This section defines the default authentication information. Setting these up properly in your ~/.snmp/snmp.conf file will greatly reduce the amount of command line arguments you need to type (especially for snmpv3).

Select from:

- 1: The default port number to use
- 2: The default snmp version number to use.
- 3: The default snmpv1 and snmpv2c community name to use when needed.
- 4: The default snmpv3 security name to use when using snmpv3
- 5: The default snmpv3 context name to use
- 6: The default snmpv3 security level to use
- 7: The default snmpv3 authentication type name to use
- 8: The default snmpv3 authentication pass phrase to use
- 9: The default snmpv3 privacy (encryption) type name to use
- 10: The default snmpv3 privacy pass phrase to use

Other options: finished, list

Select section: **finished**

The configuration information which can be put into snmp.conf is divided into sections. Select a configuration section for snmp.conf that you wish to create:

- 1: Default Authentication Options
- 2: Debugging output options
- 3: Output style options
- 4: Textual mib parsing

Other options: finished

Select section: **2**

Section: Debugging output options

Description:

This section allows debugging output of various kinds to be turned on or off.

Select from:

- 1: Turns debugging output on or off (0|1)
- 2: Debugging tokens specify which lines of debugging
- 3: Print packets as they are received or sent
- 4: Silence warnings about unknown tokens in configuration files

Other options: finished, list

Select section: **1**

Configuring: dodebugging

Description:

Turns debugging output on or off (0|1)
arguments: (0|1)

Turn debugging on (0|1): **0**

Finished Output: dodebugging 0

Section: Debugging output options

Description:

This section allows debugging output of various kinds to be turned on or off.

Select from:

- 1: Turns debugging output on or off (0|1)
- 2: Debugging tokens specify which lines of debugging
- 3: Print packets as they are received or sent

4: Silence warnings about unknown tokens in configuration files

Other options: finished, list

Select section: **3**

Configuring: dumppacket

Description:

Print packets as they are received or sent

arguments: (1|yes|true|0|no|false)

command line equivalent: -d

Print packets as they are received or sent: 1

Finished Output: dumppacket 1

Section: Debugging output options

Description:

This section allows debugging output of various kinds to be turned on or off.

Select from:

- 1: Turns debugging output on or off (0|1)
- 2: Debugging tokens specify which lines of debugging
- 3: Print packets as they are received or sent
- 4: Silence warnings about unknown tokens in configuration files

Other options: finished, list

Select section: **list**

Lines defined for section "Debugging output options" so far:

dodebugging 0

dumppacket 1

Section: Debugging output options

Description:

This section allows debugging output of various kinds to be turned on or off.

Select from:

- 1: Turns debugging output on or off (0|1)
- 2: Debugging tokens specify which lines of debugging
- 3: Print packets as they are received or sent
- 4: Silence warnings about unknown tokens in configuration files

Other options: finished, list

Select section: **finished**

The configuration information which can be put into snmp.conf is divided into sections. Select a configuration section for snmp.conf that you wish to create:

- 1: Default Authentication Options
- 2: Debugging output options
- 3: Output style options
- 4: Textual mib parsing

Other options: finished

Select section: **3**

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numericly or textually
- 3: Print OIDs numericly or textually
- 4: When OIDs contain a index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **1**

Configuring: logtimestamp

Description:

Should timestamps be shown on the output
arguments: (1|yes|true|0|no|false)

Should timestamps be shown on the output: **1**

Finished Output: logtimestamp 1

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numericly or textually
- 3: Print OIDs numericly or textually
- 4: When OIDs contain a index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **2**

Configuring: prinnumericenums

Description:

Print enums numericly or textually
command line equivalent: -Oe
arguments: (1|yes|true|0|no|false)

Print enums numericly: **0**

Finished Output: prinnumericenums 0

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numericly or textually
- 3: Print OIDs numericly or textually
- 4: When OIDs contain a index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **3**

Configuring: printnumericoids

Description:

Print OIDs numericly or textually
 command line equivalent: -On
 arguments: (1|yes|true|0|no|false)

Print enums numericly: **0**

Finished Output: printnumericoids 0

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numericly or textually
- 3: Print OIDs numericly or textually
- 4: When OIDs contain a index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **4**

Configuring: dontbreakdownoids

Description:

When OIDs contain a index to a table, they are broken into the displayable pieces and shown to you. For example the oid vacmSecurityModel.0.3.119.101.115 is nicely broken down by default and the string hidden in the oid is shown to you as vacmSecurityModel.0."wes". This token and the -Ob option disables this feature and displays it as vacmSecurityModel.0.3.119.101.115 again.
 command line equivalent: -Ob
 arguments: (1|yes|true|0|no|false)

Disable the breaking-down of OIDs?: **1**

Finished Output: dontbreakdownoids 1

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numerically or textually
- 3: Print OIDs numerically or textually
- 4: When OIDs contain an index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **6**

Configuring: quickprinting

Description:

Make the output simple for quick parsing

This option removes the equal sign and value identifying leaving just the oid and the value on the output for easier parsing in scripts
command line equivalent: -Oq
arguments: (1|yes|true|0|no|false)

Make the output simple for quick parsing: **0**

Finished Output: quickprinting 0

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numerically or textually
- 3: Print OIDs numerically or textually
- 4: When OIDs contain an index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **7**

Configuring: numeric timeticks

Description:

Print timeticks as a number and not a time-string
command line equivalent:
arguments: (1|yes|true|0|no|false)

Print timeticks as a number and not a time-string: **1**

Finished Output: numeric timeticks 1

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numerically or textually
- 3: Print OIDs numerically or textually
- 4: When OIDs contain a index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **list**

Lines defined for section "Output style options" so far:

```
logtimestamp 1
printnumericenums 0
printnumericoids 0
dontbreakdownoids 1
quickprinting 0
numeric timeticks 1
```

Section: Output style options

Description:

This section allows you to control how the output of the various commands will be formatted

Select from:

- 1: Should timestamps be shown on the output
- 2: Print enums numerically or textually
- 3: Print OIDs numerically or textually
- 4: When OIDs contain a index to a table, they are broken
- 5: Should the quotation marks in broken down oids be escaped
- 6: Make the output simple for quick parsing
- 7: Print timeticks as a number and not a time-string
- 8: Shorten OIDs printed to the screen

Other options: finished, list

Select section: **finished**

The configuration information which can be put into snmp.conf is divided into sections. Select a configuration section for snmp.conf that you wish to create:

- 1: Default Authentication Options
- 2: Debugging output options
- 3: Output style options
- 4: Textual mib parsing

Other options: finished

Select section: **4**

Section: Textual mib parsing

Description:

This section controls the textual mib parser. Textual mibs are parsed in order to convert OIDs, enumerated lists, and ... to and from textual representations and numerical representations.

Select from:

- 1: Specifies directories to be searched for mibs.
- 2: Specifies a list of mibs to be searched for and loaded.
- 3: Loads a particular mib file from a particular path
- 4: Should errors in mibs be displayed when the mibs are loaded
- 5: Should warnings about mibs be displayed when the mibs are loaded
- 6: Be strict about about mib comment termination.
- 7: Should underlines be allowed in mib symbols (illegal)
- 8: Force replacement of older mibs with known updated ones

Other options: finished, list

Select section: **4**

Configuring: showmiberrors

Description:

Should errors in mibs be displayed when the mibs are loaded
arguments: (1|yes|true|0|no|false)

Should errors in mibs be displayed when the mibs are loaded: **1**

Finished Output: showmiberrors **1**

Section: Textual mib parsing

Description:

This section controls the textual mib parser. Textual mibs are parsed in order to convert OIDs, enumerated lists, and ... to and from textual representations and numerical representations.

Select from:

- 1: Specifies directories to be searched for mibs.
- 2: Specifies a list of mibs to be searched for and loaded.
- 3: Loads a particular mib file from a particular path
- 4: Should errors in mibs be displayed when the mibs are loaded
- 5: Should warnings about mibs be displayed when the mibs are loaded
- 6: Be strict about about mib comment termination.
- 7: Should underlines be allowed in mib symbols (illegal)
- 8: Force replacement of older mibs with known updated ones

Other options: finished, list

Select section: **6**

Configuring: strictcommentterm

Description:

Be strict about about mib comment termination.
Strictly follow comment rules about parsing mibs.
arguments: (1|yes|true|0|no|false)

Be strict about about mib comment termination: **0**

Finished Output: strictcommentterm **0**

Section: Textual mib parsing

Description:

This section controls the textual mib parser. Textual mibs are parsed in order to convert OIDs, enumerated lists, and ... to and from textual representations and numerical representations.

Select from:

- 1: Specifies directories to be searched for mibs.
- 2: Specifies a list of mibs to be searched for and loaded.
- 3: Loads a particular mib file from a particular path
- 4: Should errors in mibs be displayed when the mibs are loaded
- 5: Should warnings about mibs be displayed when the mibs are loaded
- 6: Be strict about about mib comment termination.
- 7: Should underlines be allowed in mib symbols (illegal)
- 8: Force replacement of older mibs with known updated ones

Other options: finished, list

Select section: **list**

Lines defined for section "Textual mib parsing" so far:

```
showmiberrors 1
strictcommentterm 0
```

Section: Textual mib parsing

Description:

This section controls the textual mib parser. Textual mibs are parsed in order to convert OIDs, enumerated lists, and ... to and from textual representations and numerical representations.

Select from:

- 1: Specifies directories to be searched for mibs.
- 2: Specifies a list of mibs to be searched for and loaded.
- 3: Loads a particular mib file from a particular path
- 4: Should errors in mibs be displayed when the mibs are loaded
- 5: Should warnings about mibs be displayed when the mibs are loaded
- 6: Be strict about about mib comment termination.
- 7: Should underlines be allowed in mib symbols (illegal)
- 8: Force replacement of older mibs with known updated ones

Other options: finished, list

Select section: **finished**

The configuration information which can be put into snmp.conf is divided into sections. Select a configuration section for snmp.conf that you wish to create:

- 1: Default Authentication Options
- 2: Debugging output options
- 3: Output style options
- 4: Textual mib parsing

Other options: finished

Select section: finished

I can create the following types of configuration files for you.

Select the file type you wish to create:

(you can create more than one as you run this program)

- 1: snmpd.conf
- 2: snmptrapd.conf
- 3: snmp.conf

Other options: quit

Select File: **quit**

The following files were created:

snmp.conf

These files should be moved to /usr/local/share/snmp/ if you want them used by everyone on the system. In the future, if you add the -i option to the command line I'll copy them there automatically for you.

Or, if you want them for your personal use only, copy them to /root/.snmp . In the future, if you add the -p option to the command line I'll copy them there automatically for you.

(END)

16.1.2.4 snmp.conf output file

The following listing is the output of the above configuration.

```
#####
#
# snmp.conf
#
# - created by the snmpconf configuration program
#
#####
# SECTION: Default Authentication Options
#
# This section defines the default authentication
# information. Setting these up properly in your
# ~/.snmp/snmp.conf file will greatly reduce the amount of
# command line arguments you need to type (especially for snmpv3).

# defaultport: The default port number to use
# This token specifies the default port number you want packets to
# be sent to and received from.
# override: with -p on the command line.
# arguments: portnum

# defaultport

# defversion: The default snmp version number to use.
# override: with -v on the command line.
# arguments: 1|2c|3

defversion 2c

# defcommunity: The default snmpv1 and snmpv2c community name to use when needed.
# If this is specified, you don't need to include the community
# name as an argument to the snmp applications.
# override: with -c on the command line.
# arguments: communityname

defcommunity ricepad

#####
# SECTION: Debugging output options
#
# This section allows debugging output of various kinds to
# be turned on or off.

# dodebugging: Turns debugging output on or off (0|1)
# arguments: (0|1)

dodebugging 0
```

```

# dumppacket: Print packets as they are received or sent
# arguments: (1|yes|true|0|no|false)
# command line equivalent: -d

dumppacket 1

#####
# SECTION: Output style options
#
# This section allows you to control how the output of the
# various commands will be formatted

# logtimestamp: Should timestamps be shown on the output
# arguments: (1|yes|true|0|no|false)

logtimestamp 1

# printnumericenums: Print enums numericly or textually
# command line equivalent: -Oe
# arguments: (1|yes|true|0|no|false)

printnumericenums 0

# printnumericoids: Print OIDs numericly or textually
# command line equivalent: -On
# arguments: (1|yes|true|0|no|false)

printnumericoids 0

# dontbreakdownoids: When OIDs contain a index to a table, they are broken
# into the displayable pieces and shown to you.
# For example the oid vacmSecurityModel.0.3.119.101.115
# is nicely broken down by
# default and the string hidden in the oid is shown
# to you as vacmSecurityModel.0."wes". This token and the -Ob
# option disables this feature and displays it as
# vacmSecurityModel.0.3.119.101.115 again.
# command line equivalent: -Ob
# arguments: (1|yes|true|0|no|false)

dontbreakdownoids 1

# quickprinting: Make the output simple for quick parsing
# This option removes the equal sign and value identifies leaving
# just the oid and the value on the output for easier parsing in scripts
# command line equivalent: -Oq
# arguments: (1|yes|true|0|no|false)

quickprinting 0

# numerictimeticks: Print timeticks as a number and not a time-string
# command line equivalent:
# arguments: (1|yes|true|0|no|false)

numerictimeticks 1

#####
# SECTION: Textual mib parsing
#
# This section controls the textual mib parser. Textual
# mibs are parsed in order to convert OIDs, enumerated
# lists, and ... to and from textual representations
# and numerical representations.

```



```
# showmiberrors: Should errors in mibs be displayed when the mibs are loaded
# arguments: (1|yes|true|0|no|false)

showmiberrors 1

# strictcommentterm: Be strict about about mib comment termination.
# Strictly follow comment rules about parsing mibs.
# arguments: (1|yes|true|0|no|false)

strictcommentterm 0

(END)
```

16.1.3 MIB Files

Of course, the MIBs must be defined in order to work. The generic MIBs are located in the **/usr/local/share/snmp/mibs** directory, but another copy is also maintained in the **/usr/share/snmp/mibs** directory. Vendor specific mibs are generally obtained from the equipment vendor, and always end with the suffix of “.my”. In our case, we need to download several files from the Cisco web site (<http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml>), which is a general selection opening page. In our case, we will download mib files for the following Cisco equipment:

1720 Router	3640 Router
2950 LAN Switch	8510 Catalyst Switch

The files that we will download include:

```
CISCO-C2900-MIB.my
CISCO-ENTITY-VENDORTYPE-OID-MIB.my
CISCO-FLASH-MIB.my
CISCO-IP-STAT-MIB.my
CISCO-RHINO-MIB.my *
CISCO-SMI.my
ENTITY-MIB.my
IF-MIB.my
RMON-MIB.my
SNMPv2-MIB.my
SNMPv2-CONF.my
SNMPv2-SMI.my
```

To install the files, we need to copy the above files to the **/usr/local/share/snmp/mibs** directory.

How we determine which files are downloaded will be reviewed later.

Now we are able to issue queries to the Cisco equipment. This is probably the most difficult part, as it requires the user to accept on blind faith that the statements exist, which will be demonstrated later. First, we need to discuss the MIB layout.

16.1.4 MIB Assignment ¹

A MIB is a tree structure of assignments. The tree may be viewed in two different formats, one with numbers and a second as “run-on” words that have some form of meaning (?). Trying to decipher either is difficult because the one with numbers has no logical meaning and the version with words isn’t obvious.

So let’s look at a few examples – there is no better way of diving into the problem.

To start, the top of the MIB structure that is used for snmp is an ISO standard (there are others). Next, with respect to our requirements, we are going to monitor an organizational network. Third, the standard was developed for the Department of Defense (DOD). Finally, for the fourth requirement, all of the monitoring will be performed over the Internet. So we can initially structure the tree diagram as:

	<u>Mib Word</u>	<u>Mib Value</u>
ISO	iso	1
□ Organization	org	3
□ Department of Defense	dod	6
□ Internet	internet	1

Below this level, we have more options. To start, under Internet we have (very highly truncated!):

	<u>MIB Word</u>	<u>MIB Value</u>
Internet		1-3-6-1
Directory	directory	1
Management	mgmt	2
MIB-2	mib-2	1
System	system	1
System Description	sysDescr	1
Object ID	sysObjectID	2
Time Ticks	sysUpTime	3
System Contact	sysContact	4
System Name	sysName	5
System Location	sysLocation	6
System Services	sysServices	7
System Last Change	sysORLastChange	8
System OR Table	sysORTable	9
System Index	sysORIndex	1
System OR ID	sysORID	2
System OR Description	sysORDescr	3
System OR Up Time	sysORUpTime	4
System Interfaces	interfaces	2
Interface Number	ifNumber	1
Interface Table	ifTable	2
Interface Entry	ifEntry	1
Interface Description	ifDescr	2

¹ <http://www.net-snmp.org/tutorial-5>

Interface Type	ifType	3
Interface Other	other	1
Interface Regular (1822)	regular (1822)	2
Interface HDH (1822)	hdh1822	3
Interface DDNX25	ddnX25	4
...		
Interface GFP	gfp	221
Interface MTU	ifMtu	4
Interface Speed	ifSpeed	5
Interface Address	ifAddress	6
And on and on ...		

Another more traditional way to look at this tree is:

```

.iso (.1)
|----.org (.3)
|----.dod (.6)
|----.internet (.1)
|----.directory (.1)
|----.mgmt (.2)
|----.mib2 (.1)
|----.system (.1)
|----.sysDescr (.1)
|----.sysObjectID (.2)
|----.sysUpTime (.3)
|----.sysContact (.4)
|----.sysName (.5)
|----.sysLocation (.6)
|----.sysServices (.7)
|----.sysORLastChange (.8)
|----.sysORTable (.9)
|----.sysORIndex (.1)
|----.sysORID (.2)
|----.sysORDescr (.3)
|----.sysORUpTime (.4)
|----.interfaces (.2)
|----.ifNumber (.1)
|----.ifTable (.2)
|----.ifEntry (.1)
|----.ifIndex (.1)
|----.ifDescr (.2)
|----.ifType (.3)
|----.other (.1)
|----.regular1822 (.2)
|----.hdh1822 (.3)
|----.ddnX25 (.4)

```

```

| | | | | | | |---- ...
| | | | | | | |----.gfp (.221)
| | | | | | | |----.ifMtu (.4)
| | | | | | | |----.ifSpeed (.5)
| | | | | | | |----.ifPhysAddress (.6)
.....

```

This goes on and on – and if you do a printout of the default list, it is 30 pages long! And that is just the basics, not including that which is required for a specific vendor.

The above is difficult to really understand, and is costly in terms of bandwidth when transmitting information across the network. So a numeric system was devised (the numbers on the far right) to equate to a given statement.

With respect to the general operation of the SNMP structure, the first four values of the MIB are set. These are:

<u>Position</u>	<u>Value</u>	<u>Meaning</u>
1	.1	iso
2	.3	org (organization)
3	.6	dod (Department of Defense)
4	.1	Internet

So if the user desires to learn the System Description of a specific router, then either of the following descriptions may be used:

iso.org.dod.internet.mgmt.mib-2.system.sysDescription
.1.3.6.1.2.1.1

Of course the first format is a little (?) more obvious, but the second is much more efficient to transmit.

This is the most difficult part, because when we read the MIB files, we observe a large set of terms that relate to one of the dotted digits.

16.1.5 SNMP Commands ²

Now that we have loaded the monitoring software and imported the vendor (Cisco) mibs, we need to learn how to utilize it to learn what is going on a given system. The following information is derived from the online manual, **man snmpcmd**. This manual page describes the general command format, but it is not a true command.

The general format of the commands is:

snmpcmd [Options] Agent [Parameter]

Options are formed as a major-minor letter set. The major option is preceded with a

“ — ”, and the minor is a sub option, if such is required.

Common options include:

² Linux man snmpcmd

- v Specifies which version of the snmp definition to use, of which there are three. Version 1 is considered depreciated but is the default version, present use is version 2c or 3, where the default may be set in the **/etc/snmp/snmpd.conf** file.
- c Community designator. Specifies the community string for snmp version 1 or 2c. This overrides the snmp.conf file “defcommunity” token. The Community designator is kind of like a password, in that it must be included in the command.
- O Specifies output options:
The default output appears as:

```
snmpget -c public -v 1 localhost system.sysUpTime.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (14096763) 1 day,
15:09:27.63
```

 - q Removes the equal sign.
 - Q Removes the type information
 - f Gives the complete OID.
 - n Prints the OID in numeric format.
- I Specifies input options:
 - b Given only a piece of a mib, will display all mibs that meet the query designation
 - R Use a random search lookup for a partially specified mib.
- T Format output in a table format:
 - B Find all nodes that match a specified pattern.
 - D Print extended information (description) about a mib node.

Additional common options and sub-options exist.

The agent is the hostname or IP address of the remote host. If the hostname is given, it must be noted as either a Fully Qualified Host Name so that it may be looked up via a DNS or the name must be specified in the system’s **/etc/hosts** file.

The parameter is used to specify which mib is to be queried or set. This is commonly known as the **Object Identifier (OID)**.

The most common commands that will be generally used include:

snmptranslate	This command translates between a mibs numeric format and English format.
snmpget	Used to query the status of a specific mib.
snmpgetnext	Obtains the next mib level of the tree.
snmpwalk	Queries a set or table of data from the host agent.
snmptable	Displays an snmp table in a column format.
snmpset	Configures a specific mib on the host agent.
snmptrap	Sets the agent to listen for specific signals from a host agent when there is something wrong at the host.

16.1.5.1 snmptranslate

The **snmptranslate** command is used to translate a query command into the full length command or into the numeric format. This command does not issue a

query to an agent, rather it provides a translation between the word and numeric formats. The full command description is in **man snmptranslate** page.

The format of the command is:

snmptranslate [options] OID

Common options include:

- h Display a brief usage message and then exits.
- m miblist Respond to a list of multiple, colon separated, list of mibs.
- Td Print full details of the specified OID.
- Tp Print a graphical tree, rooted at the specified OID.
- w n Specify the width of the output, as some responses are quite large.

In the simplest format, the command translates between a numeric value and textual format. Examples include:

```
$ snmptranslate .1.3.6.1.2.1.1.3.0
SNMPv2-MIB::sysUpTime.0
```

To translate between the textual format to numeric.

```
$ snmptranslate -On SNMPv2-MIB::system.sysUpTime.0
.1.3.6.1.2.1.1.3.0
```

Mibs are long and difficult to remember. The **-IR** provides a random-access lookup function.

```
$ snmptranslate sysUpTime.0
Invalid object identifier: sysUpTime.0
$ snmptranslate -IR sysUpTime.0
SNMPv2-MIB::sysUpTime.0
```

A wildcard may be specified within a mib if the correct listing is unknown.

```
$ snmptranslate -lb 'sys.*ime'
system.sysUpTime
```

To obtain a list of all nodes that match a given pattern.

```
$ snmptranslate -TB 'vacm.*table'
SNMP-VIEW-BASED-ACM-MIB::vacmViewTreeFamilyTable
SNMP-VIEW-BASED-ACM-MIB::vacmAccessTable
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityToGroupTable
SNMP-VIEW-BASED-ACM-MIB::vacmContextTable
```

Multiple options may be specified.

```
$ snmptranslate -On -Td -lb 'sys.*ime'
.1.3.6.1.2.1.1.3
sysUpTime OBJECT-TYPE
-- FROM SNMPv2-MIB, RFC1213-MIB
SYNTAX TimeTicks
MAX-ACCESS read-only
```

STATUS*current*
DESCRIPTION *"The time (in hundredths of a second) since the network management portion of the system was last re-initialized."*
::= { iso(1) org(3) dod(6) internet(1) mgmt(2) mib-2(1) system(1) 3 }

And finally, to display a tree of the diagram, from a specifies level in the tree.

```
$ snmptranslate -Tp -IR system
+--system(1)
|
+-- -R-- String      sysDescr(1)
|      Textual Convention: DisplayString
+-- -R-- ObjID       sysObjectID(2)
+-- -R-- TimeTicks   sysUpTime(3)
+-- -RW- String      sysContact(4)
|      Textual Convention: DisplayString
+-- -RW- String      sysName(5)
|      Textual Convention: DisplayString
+-- -RW- String      sysLocation(6)
|      Textual Convention: DisplayString
+-- -R-- Integer     sysServices(7)
+-- -R-- TimeTicks   sysORLastChange(8)
|      Textual Convention: TimeStamp
|
+--sysORTable(9)
|
+--sysOREntry(1)
|
+-- ---- Integer     sysORIndex(1)
+-- -R-- ObjID       sysORID(2)
+-- -R-- String      sysORDescr(3)
|      Textual Convention: DisplayString
+-- -R-- TimeTicks   sysORUpTime(4)
|      Textual Convention: TimeStamp
```

The command **snmptranslate -Tp** will provide the full tree – very long!

16.1.5.2 snmpget

The **snmpget** command can be used to retrieve data from a remote host given its host name, authentication information and an OID. The full command description is in the **man snmpget** page.

The format of the command is:

```
snmpget [-Cf] [common arguments] objectID [objectID]
```

The use of the command is best illustrated through various examples.
 Simple query of information.

```
$ snmpget -c demopublic -v 2c 1720c.ourlab.com
system.sysUpTime.0
system.sysUpTime.0 = Timeticks: (586731977) 67 days, 21:48:39.77
```

In this example, the host name is specified by **1720c.ourlab.com**. The OID is system.sysUpTime.0 . There are two points that need to be highlighted in our command.

1. We did not specify the full query name. The initial part – iso.org.dod.internet.mgmt is understood, and need not be specified.
2. The query of **system.sysUpTime** is terminated in **Zero**. This is a simple way of instructing the agent that it is the end of the query (no real query elements end in a zero).

The **-c demopublic** is the community string for the request, and the query was made for snmp version 2c (**-v 2c**).

Reversal of **-c** and **-v**.

```
$ snmpget -v 2c -c demopublic 1720c.ourlab.com
system.sysUpTime.0
system.sysUpTime.0 = Timeticks: (586752671) 67 days, 21:52:06.71
```

The position for the version and the community string may be reversed.

Abbreviation of the OID.

So far we have specified the OID as “system.sys...”, but that may be abbreviated.

```
$ snmpget -v 2c -c demopublic 1720c.ourlab.com sysUpTime.0
system.sysUpTime.0 = Timeticks: (586752671) 67 days, 21:52:06.71
```

A common problem with the issuance of the command is to forget the terminating zero. If this happens, the agent will report an error.

16.1.5.3 snmpgetnext

The **snmpgetnext** command, which is similar in usage to the **snmpget** command, is used to retrieve the **next** oid in the mib tree of data. Instead of returning the data you requested, it returns the **next** OID in the tree and its value. The full command description is in the **man snmpgetnext** page.

The format of the command is:

```
snmpgetnext [ common arguments ] objectID
```

This is illustrated in examples.

Get next value in the index.

```
$ snmpgetnext -v 2c -c demopublic 1720c.ourlab.com
system.sysUpTime.0
system.sysContact.0 = Wes Hardaker wjhardaker@ucdavis.edu
```


Recall from earlier discussion that part of the initial mib tree appeared as:

System	system	1
System Description	sysDescr	1
Object ID	sysObjectID	2
Time Ticks	sysUpTime	3
System Contact	sysContact	4

So we can see that the next entry after sysUpTime is sysContact.

Continued next queries.

Once the first query element is known, we are able to learn what the next one is, so we can continue the queries.

```
$ snmpgetnext -v 2c -c demopublic test.net-snmp.org
system.sysUpTime.0
system.sysContact.0 = Wes Hardaker wjhardaker@ucdavis.edu
$ snmpgetnext -v 2c -c demopublic test.net-snmp.org
system.sysContact.0
system.sysName.0 = net-snmp
$ snmpgetnext -v 2c -c demopublic test.net-snmp.org
system.sysName.0
system.sysLocation.0 = UCDavis
```

16.1.5.4 snmpwalk

The **snmpwalk** command essentially performs a whole series of getnexts automatically for you, and stops when it returns results that are no longer inside the range of the OID that you originally specified. If you wanted to get **all of the information** stored on a machine in the system MIB group, for instance, you could use this command. The full command description is in the **man snmpwalk** page.

The format of the command is:

```
snmpwalk [ common arguments ] objectID
```

```
$ snmpwalk -v 2c -c demopublic 1720c.ourlab.com system
system.sysDescr.0 = 1720c net-snmp B.10.20 A 9000/715
system.sysObjectID.0 = OID:
enterprises.244lab.dal.ourlab.com.SnmpAgent.hpux10
system.sysUpTime.0 = Timeticks: (586998396) 67 days, 22:33:03.96
system.sysContact.0 = Wes Hardaker wjhardaker@ucdavis.edu
system.sysName.0 = net-snmp
system.sysLocation.0 = UCDavis
system.sysORLastChange.0 = Timeticks: (0) 0:00:00.00
system.sysORTable.sysOREntry.sysORIndex.1 = 1
system.sysORTable.sysOREntry.sysORIndex.2 = 2
system.sysORTable.sysOREntry.sysORIndex.3 = 3
system.sysORTable.sysOREntry.sysORIndex.4 = 4
system.sysORTable.sysOREntry.sysORIndex.5 = 5
```

```

system.sysORTable.sysOREntry.sysORID.1 = OID:
.iso.org.dod.internet.snmpV2.snmpModules.snmpMIB
system.sysORTable.sysOREntry.sysORID.2 = OID:
.iso.org.dod.internet.snmpV2.snmpModules.snmpVacmMIB.vacm
MIBConformance.vacmMIBGroups.vacmBasicGroup
system.sysORTable.sysOREntry.sysORID.3 = OID:
.iso.org.dod.internet.snmpV2.snmpModules.snmpFrameworkMIB.
snmpFrameworkMIBConformance.snmpFrameworkMIBComplian
ces.snmpFrameworkMIBCompliance
system.sysORTable.sysOREntry.sysORID.4 = OID:
.iso.org.dod.internet.snmpV2.snmpModules.snmpMPDMIB.snmp
MPDMIBConformance.snmpMPDMIBCompliances.snmpMPDCom
pliance
system.sysORTable.sysOREntry.sysORID.5 = OID:
.iso.org.dod.internet.snmpV2.snmpModules.snmpUsmMIB.usmMI
BConformance.usmMIBCompliances.usmMIBCompliance
system.sysORTable.sysOREntry.sysORDescr.1 = The Mib module
for SNMPv2 entities.
system.sysORTable.sysOREntry.sysORDescr.2 = View-based
Access Control Model for SNMP.
system.sysORTable.sysOREntry.sysORDescr.3 = The SNMP
Management Architecture MIB.
system.sysORTable.sysOREntry.sysORDescr.4 = The MIB for
Message Processing and Dispatching.
system.sysORTable.sysOREntry.sysORDescr.5 = The management
information definitions for the SNMP User-based Security Model.
system.sysORTable.sysOREntry.sysORUpTime.1 = Timeticks: (0)
0:00:00.00
system.sysORTable.sysOREntry.sysORUpTime.2 = Timeticks: (0)
0:00:00.00
system.sysORTable.sysOREntry.sysORUpTime.3 = Timeticks: (0)
0:00:00.00
system.sysORTable.sysOREntry.sysORUpTime.4 = Timeticks: (0)
0:00:00.00
system.sysORTable.sysOREntry.sysORUpTime.5 = Timeticks: (0)
0:00:00.00

```

16.1.5.5 snmptable

The **snmptable** command displays a SNMP table for you in an easy to read column based fashion. Consider the sysORTable, which you saw all the data from in the last section. It's hard to get a good feel about the correlation of data when its displayed in a long list, like the output of snmpwalk gives you. Instead, snmptable nicely formats it for you. The full command description is in the **man snmptable** page.

The format of the command is:

```
snmptable [ common arguments ] objectID
```

```
$ snmptable -v 2c -c demopublic test.net-snmp.org sysORTable
SNMP table: system.sysORTable
```

```
sysORIndex
  sysORID                                sysORDescr
  sysORUpTime
  1
  .iso.org.dod.internet.snmpV2.snmpModules.snmpMIB
  The Mib module for SNMPv2 entities. 0:0:00:00.00
  2
  .iso.org.dod.internet.snmpV2.snmpModules.snmpVacmMIB.vacm
  MIBConformance.vacmMIBGroups.vacmBasicGroup
  View-based Access Control Model for SNMP. 0:0:00:00.00
  3
  .iso.org.dod.internet.snmpV2.snmpModules.snmpFrameworkMIB.
  snmpFrameworkMIBConformance.snmpFrameworkMIBCompliance
  ces.snmpFrameworkMIBCompliance The
  SNMP Management Architecture MIB. 0:0:00:00.00
  4
  .iso.org.dod.internet.snmpV2.snmpModules.snmpMPDMIB.snmp
  MPDMIBConformance.snmpMPDMIBCompliances.snmpMPDCom
  pliance
  The MIB for Message Processing and Dispatching. 0:0:00:00.00
  5
  .iso.org.dod.internet.snmpV2.snmpModules.snmpUsmMIB.usmMI
  BConformance.usmMIBCompliances.usmMIBCompliance The
  management information definitions for the SNMP User-based
  Security Model. 0:0:00:00.00
```

This may or may not be easier to read because some of the responses can be quite long. The answer to this is to format the width of the output.

```
$ snmptable -v 2c -c demopublic -Cw 80 1720c.ourlab.com
sysORTable
SNMP table: system.sysORTable
```

```
sysORIndex
  sysORID
  1
  .iso.org.dod.internet.snmpV2.snmpModules.snmpMIB
  2
  .iso.org.dod.internet.snmpV2.snmpModules.snmpVacm
  MIB.vacmMIBConformance.vacmMIBGroups.vacmBasicGro
  up
  3
  .iso.org.dod.internet.snmpV2.snmpModules.snmpFram
```

```

eworkMIB.snmpFrameworkMIBConformance.snmpFramework
MIBCompliances.snmpFrameworkMIBCompliance
    4
.iso.org.dod.internet.snmpV2.snmpModules.snmpMPDM
IB.snmpMPDMIBConformance.snmpMPDMIBCompliances.sn
mpMPDCompliance
    5
.iso.org.dod.internet.snmpV2.snmpModules.snmpUsmM
IB.usmMIBConformance.usmMIBCompliances.usmMIBComp
liance

```

SNMP table: system.sysORTable, part 2

```

sysORDescr sysORUpTime
                                The Mib
module for SNMPv2 entities. 0:0:00:00.00
                                View-based
Access Control Model for SNMP. 0:0:00:00.00
                                The SNMP
Management Architecture MIB. 0:0:00:00.00
                                The MIB for Message
Processing and Dispatching. 0:0:00:00.00
The management information definitions for the SNMP
User-based Security Model. 0:0:00:00.00

```

In this document, it is hard to appreciate the formatting of the output. But if you had a long sheet of paper (legal landscape format), one would observe that the formatting of the table is right justified, and this does lead to improved reading. For an improved formatting, see the web page at <http://www.net-snmp.org/tutorial-5/commands/snmp.html>.

16.1.5.6 snmpset

The **snmpset** command is used to actually modify information on the remote host. For each variable you want to set, you need to specify the OID to update, the data type and the value you want to set it to. The full command description is in the **man snmpset** page.

The format of the command is:

snmpset [common arguments] objectid type value

A type and a value to set must accompany each object identifier. Each variable name is given in the format specified in variables.

The type is a single character, one of:

```

i    INTEGER
u    UNSIGNED
s    STRING
x    HEX STRING

```

```

d    DECIMAL STRING
n    NULLOBJ
o    OBJID
t    TIMETICKS
a    IPADDRESS
b    BITS

```

The types are able to be displayed by utilizing the help format of the command.

```

$ snmpset -h |& tail -4
type - one of i, u, t, a, o, s, x, d, n
i: INTEGER, u: unsigned INTEGER, t: TIMETICKS, a: IPADDRESS
o: OBJID, s: STRING, x: HEX STRING, d: DECIMAL STRING
U: unsigned int64, l: signed int64, F: float, D: double

```

This is best illustrated by setting a value within the MIBs.

```

$ snmpget -v 2c -c demopublic test.net-snmp.org
ucdDemoPublicString.0
enterprises.ucdavis.ucdDemoMIB.ucdDemoMIBObjects.ucdDemoPu
blic.ucdDemoPublicString.0 = "hi there"
$ snmpset -v 2c -c demopublic test.net-snmp.org
ucdDemoPublicString.0 s "hello world"
enterprises.ucdavis.ucdDemoMIB.ucdDemoMIBObjects.ucdDemoPu
blic.ucdDemoPublicString.0 = "hello world"
$ snmpget -v 2c -c demopublic test.net-snmp.org
ucdDemoPublicString.0
enterprises.ucdavis.ucdDemoMIB.ucdDemoMIBObjects.ucdDemoPu
blic.ucdDemoPublicString.0= "hello world"

```

The above example was extracted from the net-snmp tutorials, and thus the mib is specific to the example site.

Naturally, the Object ID must exist and the user must have write permissions in order to make modifications.

16.1.5.7 snmptrap

Traps may be used by network devices to signal abnormal conditions to a management station. An agent must be configured to send a notification using an application like **snmptrap** (vendors may have their own version), and the snmp monitor must be set up to receive the notification using the **snmptrapd** application. The full command description is in the **man snmptrap** page.

Just to make things difficult, a trap notification may be in one of two different formats. The older version that supports snmp version 1 and the newer snmp version 2 format. For this discussion, we will assume that the equipment will issue snmp version 2 notifications.

To configure the agent, the following actions must be taken.

1. Within the agent, a MIB file must be created that looks something like the following for a test example.

```

NOTIFICATION-TEST-MIB DEFINITIONS ::= BEGIN
    IMPORTS ucdavis FROM UCD-SNMP-MIB;

    demonotifs OBJECT IDENTIFIER ::= { ucdavis 991 }

    demo-notif NOTIFICATION-TYPE
        STATUS current
        OBJECTS { sysLocation }
        DESCRIPTION "Just a test notification"
        ::= { demonotifs 17 }

END

```

This sets up the definition of the trap that will be transmitted.

At the agent, issuing the following command:

```
$ snmptrap -v 2c -c public localhost '' NOTIFICATION-TEST-
MIB::demo-notif \ SNMPv2-MIB::sysLocation.0 s "just here"
```

At the monitor, the following will be displayed:

```

1999-11-13 08:31:33 localhost [127.0.0.1]:
    SNMPv2-MIB::sysUpTime.0 = Timeticks: (13917129) 1 day,
14:39:31.29
    SNMPv2-MIB::snmpTrapOID.0 = OID: NOTIFICATION-TEST-
MIB::demo-notif
    SNMPv2-MIB::sysLocation.0 = "just here"

```

The setting up of a trap can be quite involved. The simple explanation here should not be taken lightly.

16.1.6 MIB Files

So what does a MIB file look like? If one looks at a MIB file it will conform to a defined structure. In this discussion, we will look at a MIB file defined by Cisco for the 2900 series of Catalyst LAN switches. Recall, that the MIB files are stored in the **/usr/local/share/snmp/mibs** directory.

For a starter, any text following a “– –” is considered a comment. Typically the double dash will be in the first two columns of the line, but may also occur at the end when the author wishes to add additional information that is not relevant to the format.

16.1.6.1 BEGIN

The first line of a MIB file (not including comments) should appear something like:

```
CISCO-C2900-MIB  DEFINITIONS  ::=  BEGIN
```

This specifies the start of the file.

16.1.6.2 IMPORTS

The Imports section specifies other MIB definitions that are required for the general operation. These definitions must be imported from other MIB files. For example:

```
IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE
    Counter32, Gauge32, Integer32
    FROM SNMPv2-SMI
...

```

This specifies that the **snmp** values of Counter32, Gauge32, and Integer32 are obtained from the standard MIB file of SNMPv2-SMI. Additional snmp values are also defined.

16.1.6.3 MODULE IDENTITY

The Module Identity specifies the name of the module. This is the first major part of the (vendor) specific MIB definition. An example of a Module Identity could be:

```
CiscoC2900MIB MODULE-IDENTITY
    LAST-UPDATED "200205300000Z"
    ORGANIZATION "Cisco Systems, Inc."
    CONTACT-INFO
        "Postal: Cisco Systems, Inc."
    ...
DESCRIPTION
    "The MIB module for Catalyst 2900 enterprise specific information"

REVISION "200205300000Z"
DESCRIPTION
    'Comments regarding the revision'

REVISION "200107251345Z"
DESCRIPTION
    'Comments regarding the revision'

...
:: = { ciscoMgmt 87 }
```

We start out our definition with the identifier:

```
.ciscoMgmt
.87
```

or

The length of this section is dependent upon how many device MIBs have been implemented for the specified device. In this case, the numeric value of this module (CiscoC2900MIB) is 1, as in this case it is the only one specified because it is not referenced to a higher level, that is, it is the first identifier level for all following definitions.

It starts the definition line, so it is written as:

```
.ciscoMgmt.ciscoC2900MIB.
.87.1
```

or

16.1.6.4 OBJECT IDENTIFIER

An Object Identifier follows the Module Identity in the snmp definition. In order to know which Module Identity precedes it, the Object Identifier must specify it in the same line.

```
C2900MIBObjects OBJECT IDENTIFIER ::= { cisco2900MIB 1 }
```

Here, the C2900MIBObject is a subset of the cisco2900MIB module identity. It would be written:

```
.ciscoMgmt.cisco2900MIB.c2900MIBObjects
.87.1.
```

or

This section may be further refined in definition with MIB Groups.

16.1.6.5 MIB Groups

The MIB Group provides further refinement to the definition of a specific MIB. In our continuing example we have:

```
c2900SysInfo      OBJECT IDENTIFIER ::= { c2900MIBObjects 1 }
c2900SysConfig    OBJECT IDENTIFIER ::= { c2900MIBObjects 2 }
c2900????3       OBJECT IDENTIFIER ::= { c2900MIBObjects 3 }
c2900Port         OBJECT IDENTIFIER ::= { c2900MIBObjects 4 }
c2900BandwidthUsage OBJECT IDENTIFIER ::= { c2900MIBObjects 5 }
```

Now, the MIB will be written:

```
.ciscoMgmt.cisco2900MIB.c2900MIBObjects.c2900SysInfo
or .87.1.1.1
.ciscoMgmt.cisco2900MIB.c2900MIBObjects.c2900SysConfig
or .87.1.1.2
.ciscoMgmt.cisco2900MIB.c2900MIBObjects.????
or .87.1.1.3
.ciscoMgmt.cisco2900MIB.c2900MIBObjects.c2900Port
or .87.1.1.4
.ciscoMgmt.cisco2900MIB.c2900MIBObjects.c2900BandwidthUsage
or .87.1.1.5
```

16.1.6.6 OBJECT TYPES

The Object Type add even more refinement to the definition of a MIB. Our example continues:

```
c2900InfoBoardRevision OBJECT-TYPE
    SYNTAX Gauge32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Returns the revision number of the main board"
```

³ Additional searching of the Cisco MIB file is required.

on which the FastSwitch firmware resides.”
 :: = { c2900SysInfo 1 }

The MIB is written:

.ciscoMgmt.cisco2900MIB.c2900MIBObjects.c2900SysInfo.c2900InfoBoa
 rdRevision
 or .87.1.1.1.1

c2900InfoPeakBuffersUsed OBJECT-TYPE
 SYNTAX Gauge32
 UNITS “buffers”
 MAX-ACCESS read-only
 STATUS current
 DESCRIPTION
 “The maximum number of 64-byte buffers used in the
 main switch buffer pool.”
 :: = { c2900SysInfo 2 }

The MIB is written:

.cisco2900MIB.c2900MIBObjects.c2900SysInfo.c2900InfoPeakBuffersUsed
 or .87.1.1.1.2

c2900ModuleTable OBJECT-TYPE
 SYNTAX SEQUENCE OF C2900ModuleEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION
 “A list of module entries”
 :: = { c2900MIBObjects 3 }

This MIB is written:

.cisco2900MIB.c2900MIBObjects.c2900SysInfo.c2900ModuleTable
 or .87.1.1.3.3

These examples show only a very few of the Cisco 2900 MIB file – which is 32 pages long.

16.1.1.7 Required Vendor MIB Files

OK, what files are required to download. This is not too hard of a problem, but does require a little bit of investigation.

First, list the contents of the **/usr/local/share/snmp/mibs** directory. This is used to make a comparison of which files exist already.

Second, display the contents of the vendor MIB file for a specific piece of equipment. In this case we will start with the 2900 series of Cisco Catalyst LAN Switch. View the IMPORTS SECTION. In the case of the CISCO-C2900-MIB.my file, we have the following FROM designators:

FROM SNMPv2-SMI
 FROM SNMPv2-TC

```
FROM IF-MIB
FROM CISCO-SMI
FROM RFC1213-MIB
FROM SNMPv2-CONF
```

Now comparing to the ./mib directory, we find that the following files are missing:

```
CISCO-SMI          CISCO-C2900-MIB
```

Therefore, before we can snmp the 2950, we need to install both the CISCO-SMI and CISCO-C2900-MIB files.

16.1.1.8 MIB Initial Numbering

The above discussion specified the MIBs for the Cisco 2900 Catalyst LAN Switch, but these numbers must be preceded by others that specify the start of the MIB. Naturally we know that the numeric sequence starts with:

```
.iso.org.dod.internet.          or
.1.3.6.1.
```

On reviewing the basic MIB structure we can learn that the numbers “.1.3.6.1.4.1” equates to:

```
.iso.org.dod.internet.private.enterprise
```

The first “.9” specifies that the private enterprise is Cisco.

```
.iso.org.dod.internet.private.enterprise.cisco
```

Using the snmptranslate command, and assuming that we have copied the two files that are required, we learn:

```
$ snmptranslate -m +CISCO-C2900-MIB -IR -On ciscoMgmt
.1.3.6.1.4.1.9.9
```

So we now have:

```
.iso.org.dod.internet.private.enterprise.cisco.ciscoMgmt
```

Next, issue the same request for the c2900MIBObjects numbering:

```
$ snmptranslate -m +CISCO-C2900-MIB -IR -On c2900MIBObjects
.1.3.6.1.4.1.9.9.87.1
```

Continuing, we have:

```
.iso.org.dod.internet.private.enterprise.cisco.ciscoMgmt.c2900MIBObjects
```

Recall from Section 16.1.6.4 the definition – C2900MIBObjects OBJECT IDENTIFIER ::= { cisco2900MIB 1 }. This specifies that we are the designator “1” after the “ciscoMgmt” value.

Now we have the confusion of the values of “.4.1.9.9” after the initial “.1.3.6.1”.

- .4 Standard MIB designation for a private vendor.
- .1 Standard MIB designation for a private enterprise, in our case Cisco.
- .9 Learned from the CISCO-SMI.my MIB file, it specifies the vendor as Cisco.
- .9 Learned from the CISCO-SMI.my MIB file, it is the MODULE-IDENTITY specifier.
- .87 Learned from the CISCO-C2900-MIB.my file, it specifies the MODULE-IDENTITY is ciscoMgmt
- .1 Learned from the CISCO-C2900-MIB.my file, it specifies the OBJECT-IDENTIFIER is c2900MIBObjects.

Therefore, the MIB for c2900InfoPeakBuffersUsed would be:

```
.iso.org.dod.internet.private.enterprises.9.9.cisco2900MIB.c2900MIBObjects.c2900SysInfo.c2900InfoPeakBuffersUsed
```

or

```
$ snmptranslate -m +CISCO-C2900-MIB -IR -On c2900InfoPeakBuffersUsed
.1.3.6.1.4.1.9.9.87.1.1.2
```

Now if one knew the MIB numeric format, we can translate to the word format:

```
$ snmptranslate -Td .1.3.6.1.4.1.9.9.87.1
SNMPv2-SMI::enterprises.9.9.87.1
enterprises OBJECT-TYPE
-- FROM SNMPv2-SMI, RFC1155-SMI
:= { iso(1) org(3) dod(6) internet(1) private(4) enterprises(1) 9 9 87 1 }
```

16.1.2 Making it work

After all of the above, we need to put it together and make it work. We will put several examples together to demonstrate how the process works. Our discussion will provide two different snmp agents, one for a Linux host (Red Hat 9) and for various Cisco equipment (1720, 2950, 3640, 8510).

16.1.2.1 Linux Host Agent (RH 9)

After we have set up the monitor, we need to configure the agent, in this case on a Linux Red Hat 9 installation.

The following actions need to be performed in order to perform an snmp operation on a Linux server. The following files were created by the snmpconf program as a basic example – and to function as in the “Keep It Simple” philosophy.

16.1.2.1.1 Linux snmpd.conf File

The following is a listing of the snmpd.conf file.

```
#####
#
# snmpd.conf
#
# - created by the snmpconf configuration program
#
```

```
#####
# SECTION: System Information Setup
#
# This section defines some of the information reported in
# the "system" mib group in the mibII tree.

# syslocation: The [typically physical] location of the system.
# Note that setting this value here means that when trying to
# perform an snmp SET operation to the sysLocation.0 variable will make
# the agent return the "notWritable" error code. IE, including
# this token in the snmpd.conf file will disable write access to
# the variable.
# arguments: location_string

syslocation "244 Lab Dallas TX"

# syscontact: The contact information for the administrator
# Note that setting this value here means that when trying to
# perform an snmp SET operation to the sysContact.0 variable will make
# the agent return the "notWritable" error code. IE, including
# this token in the snmpd.conf file will disable write access to
# the variable.
# arguments: contact_string

syscontact "Dennis Rice <drice@ourlab.com>"

#####
# SECTION: Access Control Setup
#
# This section defines who is allowed to talk to your running
# snmp agent.

# rwuser: a SNMPv3 read-write user
# arguments: user [noauth|auth|priv] [restriction_oid]

rwuser root

# rwcommunity: a SNMPv1/SNMPv2c read-write access community name
# arguments: community [default|hostname|network/bits] [oid]

rwcommunity 244Lab

#####
# SECTION: Agent Operating Mode
#
# This section defines how the agent will operate when it
# is running.

# master: Should the agent operate as a master agent or not.
# Currently, the only supported master agent type for this token
# is "agentx".
#
# arguments: (on|yes|agentx|all|off|no)

master off

# agentuser: The system user that the agent runs as.
# arguments: name|#uid
```

```
agentuser root
```

16.1.2.1.2 Linux snmp.conf File

```
#####
#
# snmp.conf
#
# - created by the snmpconf configuration program
#
#####
# SECTION: Default Authentication Options
#
# This section defines the default authentication
# information. Setting these up properly in your
# ~/.snmp/snmp.conf file will greatly reduce the amount of
# command line arguments you need to type (especially for snmpv3).

# defversion: The default snmp version number to use.
# override: with -v on the command line.
# arguments: 1|2c|3

defversion 2c

# defcommunity: The default snmpv1 and snmpv2c community name to use when needed.
# If this is specified, you don't need to include the community
# name as an argument to the snmp applications.
# override: with -c on the command line.
# arguments: communityname

defcommunity 244Lab

#####
# SECTION: Debugging output options
#
# This section allows debugging output of various kinds to
# be turned on or off.

# dodebugging: Turns debugging output on or off (0|1)
# arguments: (0|1)

dodebugging 0

# debugtokens: Debugging tokens specify which lines of debugging
# output you'd actually like to see. Each section of code is most
# likely instrumented with a particular "tag". So, to see that tag you
# would specify it here. Specifying a tag will match against all
# tags that begin with that prefix, so the tag "test" will match
# "test_function" and "test_something" and...
# There are a few special tokens as well:
# - ALL: turns on all the tokens (which generates lots of output)
# - trace: prints 'trace' lines showing source code files and
# - line numbers as they're traversed.
# - dump: Nicely breaks down packets as they're parsed or sent out.
# command line equivalent: -Dtoken[,token...]
```

```
# arguments: token[,token...]

debugtokens ALL

# dumppacket: Print packets as they are received or sent
# arguments: (1|yes|true|0|no|false)
# command line equivalent: -d

dumppacket 1
```

16.1.2.1.3 Branch Vs. Leaf Query

Recognize that a question to an agent must be directed at a “leaf”, that is, to the last element of the tree. Querying a branch that leads to the leaf does not obtain results. An alternative is to ask for all information below a specified point in the tree, thus all leafs will be queried and answers received. If we use the query statement of “snmpget”, then we are asking the answer from a specific leaf, whereas if we use the query statement of “snmpwalk”, we will obtain a list of answers from multiple leafs.

While one is first learning about how to use snmp, it is better to use the **snmpwalk** command. As an example, the following is the response on a system when using the configuration files immediately above. (The output has been highly edited for space purposes.)

```
# snmpwalk localhost -c ricepad system
2004-04-25 15:45:16
Sending 42 bytes to 127.0.0.1
2004-04-25 15:45:16 0000: 30 28 02 01 01 04 07 72 69 63 65 70 61 64 A1 1A 0(. . . . .ricepad..
2004-04-25 15:45:16 0016: 02 04 05 55 7D 4E 02 01 00 02 01 00 30 0C 30 0A ...U}N.....0.0.
2004-04-25 15:45:16 0032: 06 06 2B 06 01 02 01 01 05 00 ...+.....
2004-04-25 15:45:16
2004-04-25 15:45:16
Received 101 bytes from 127.0.0.1
2004-04-25 15:45:16 0000: 30 63 02 01 01 04 07 72 69 63 65 70 61 64 A2 55 0c.....ricepad.U
2004-04-25 15:45:16 0016: 02 04 05 55 7D 4E 02 01 00 02 01 00 30 47 30 45 ...U}N.....0G0E
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 01 00 04 39 4C 69 6E 75 ..+.....9Linu
2004-04-25 15:45:16 0048: 78 20 66 72 69 65 64 20 32 2E 34 2E 32 30 2D 36 x fried 2.4.20-6
2004-04-25 15:45:16 0064: 20 23 31 20 54 68 75 20 46 65 62 20 32 37 20 31 #1 Thu Feb 27 1
2004-04-25 15:45:16 0080: 30 3A 30 31 3A 31 39 20 45 53 54 20 32 30 30 33 0:01:19 EST 2003
2004-04-25 15:45:16 0096: 20 69 36 38 36 i686
2004-04-25 15:45:16
SNMPv2-MIB::sysDescr.0 = STRING: Linux fried 2.4.20-6 #1 Thu Feb 27 10:01:19 EST 2003 i686
2004-04-25 15:45:16
Sending 44 bytes to 127.0.0.1
2004-04-25 15:45:16 0000: 30 2A 02 01 01 04 07 72 69 63 65 70 61 64 A1 1C 0*.....ricepad..
2004-04-25 15:45:16 0016: 02 04 05 55 7D 50 02 01 00 02 01 00 30 0E 30 0C ...U}P.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 02 00 05 00 ..+.....
2004-04-25 15:45:16
2004-04-25 15:45:16
Received 47 bytes from 127.0.0.1
2004-04-25 15:45:16 0000: 30 2D 02 01 01 04 07 72 69 63 65 70 61 64 A2 1F 0-.....ricepad..
2004-04-25 15:45:16 0016: 02 04 05 55 7D 50 02 01 00 02 01 00 30 11 30 0F ...U}P.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 03 00 43 03 00 FE 8B ..+.....C....
2004-04-25 15:45:16
SNMPv2-MIB::sysUpTime.0 = 65163
2004-04-25 15:45:16
Sending 44 bytes to 127.0.0.1
2004-04-25 15:45:16 0000: 30 2A 02 01 01 04 07 72 69 63 65 70 61 64 A1 1C 0*.....ricepad..
2004-04-25 15:45:16 0016: 02 04 05 55 7D 51 02 01 00 02 01 00 30 0E 30 0C ...U}Q.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 03 00 05 00 ..+.....
2004-04-25 15:45:16
2004-04-25 15:45:16
Received 57 bytes from 127.0.0.1
2004-04-25 15:45:16 0000: 30 37 02 01 01 04 07 72 69 63 65 70 61 64 A2 29 07.....ricepad.)
2004-04-25 15:45:16 0016: 02 04 05 55 7D 51 02 01 00 02 01 00 30 1B 30 19 ...U}Q.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 04 00 04 0D 22 44 65 6E ..+....."Den
2004-04-25 15:45:16 0048: 6E 69 73 20 52 69 63 65 22 nis Rice"
2004-04-25 15:45:16
SNMPv2-MIB::sysContact.0 = STRING: "Dennis Rice"
2004-04-25 15:45:16
Sending 44 bytes to 127.0.0.1
2004-04-25 15:45:16 0000: 30 2A 02 01 01 04 07 72 69 63 65 70 61 64 A1 1C 0*.....ricepad..
```

```

2004-04-25 15:45:16 0016: 02 04 05 55 7D 52 02 01 00 02 01 00 30 0E 30 0C ...U}R.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 04 00 05 00 ...+.....
2004-04-25 15:45:16
2004-04-25 15:45:16
Received 49 bytes from 127.0.0.1
2004-04-25 15:45:16 0000: 30 2F 02 01 01 04 07 72 69 63 65 70 61 64 A2 21 0/.....ricepad.!
2004-04-25 15:45:16 0016: 02 04 05 55 7D 52 02 01 00 02 01 00 30 13 30 11 ...U}R.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 05 00 04 05 66 72 69 65 ...+.....frie
2004-04-25 15:45:16 0048: 64 d
2004-04-25 15:45:16
SNMPv2-MIB::sysName.0 = STRING: fried
2004-04-25 15:45:16
Sending 44 bytes to 127.0.0.1
2004-04-25 15:45:16 0000: 30 2A 02 01 01 04 07 72 69 63 65 70 61 64 A1 1C 0*.....ricepad..
2004-04-25 15:45:16 0016: 02 04 05 55 7D 53 02 01 00 02 01 00 30 0E 30 0C ...U}S.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 05 00 05 00 ...+.....
2004-04-25 15:45:16
2004-04-25 15:45:16
Received 58 bytes from 127.0.0.1
2004-04-25 15:45:16 0000: 30 38 02 01 01 04 07 72 69 63 65 70 61 64 A2 2A 08.....ricepad.*
2004-04-25 15:45:16 0016: 02 04 05 55 7D 53 02 01 00 02 01 00 30 1C 30 1A ...U}S.....0.0.
2004-04-25 15:45:16 0032: 06 08 2B 06 01 02 01 01 06 00 04 0E 22 72 69 63 ...+....."ric
2004-04-25 15:45:16 0048: 65 70 61 64 20 68 6F 6D 65 22 epad home"
2004-04-25 15:45:16
SNMPv2-MIB::sysLocation.0 = STRING: "ricepad home"
2004-04-25 15:45:16
Sending 46 bytes to 127.0.0.1
2004-04-25 15:45:16 0000: 30 2C 02 01 01 04 07 72 69 63 65 70 61 64 A1 1E 0,.....ricepad..
2004-04-25 15:45:16 0016: 02 04 05 55 7D 56 02 01 00 02 01 00 30 10 30 0E ...U}V.....0.0.
2004-04-25 15:45:16 0032: 06 0A 2B 06 01 02 01 01 09 01 02 01 05 00 ...+.....
2004-04-25 15:45:16
2004-04-25 15:45:16
Received 52 bytes from 127.0.0.1
2004-04-25 15:45:16 0000: 30 32 02 01 01 04 07 72 69 63 65 70 61 64 A2 24 02.....ricepad.$
2004-04-25 15:45:16 0016: 02 04 05 55 7D 56 02 01 00 02 01 00 30 16 30 14 ...U}V.....0.0.
2004-04-25 15:45:16 0032: 06 0A 2B 06 01 02 01 01 09 01 02 02 06 06 2B 06 ...+.....+.
2004-04-25 15:45:16 0048: 01 06 03 01 ....
2004-04-25 15:45:16
SNMPv2-MIB::sysORID.2 = OID: SNMPv2-MIB::snmpMIB
2004-04-25 15:45:16

```

Observing the above, we can ask several specific questions, knowing the correct leaf to ask.

```

# snmpwalk localhost -c ricepad sysContact.0
Sending 44 bytes to 127.0.0.1
2004-04-25 16:05:02 0000: 30 2A 02 01 01 04 07 72 69 63 65 70 61 64 A0 1C 0*.....ricepad..
2004-04-25 16:05:02 0016: 02 04 61 1B 73 0C 02 01 00 02 01 00 30 0E 30 0C ...a.s.....0.0.
2004-04-25 16:05:02 0032: 06 08 2B 06 01 02 01 01 04 00 05 00 ...+.....
2004-04-25 16:05:02
2004-04-25 16:05:02
Received 57 bytes from 127.0.0.1
2004-04-25 16:05:02 0000: 30 37 02 01 01 04 07 72 69 63 65 70 61 64 A2 29 07.....ricepad.)
2004-04-25 16:05:02 0016: 02 04 61 1B 73 0C 02 01 00 02 01 00 30 1B 30 19 ...a.s.....0.0.
2004-04-25 16:05:02 0032: 06 08 2B 06 01 02 01 01 04 00 04 0D 22 44 65 6E ...+....."Den
2004-04-25 16:05:02 0048: 6E 69 73 20 52 69 63 65 22 nis Rice"
2004-04-25 16:05:02
SNMPv2-MIB::sysContact.0 = STRING: "Dennis Rice"

```

As one becomes more proficient, additional questions may be generated.

16.1.2.2 Configuring Cisco Host Agent⁴

In order to enable Cisco equipment to act as an agent, the following actions need to be performed.

16.1.2.2.1 Cisco 1720 Router

After telneting into the 1720 router, issue the following commands to configure it to support snmp:

```

> enable
password your-password
# config t

```

⁴ Cisco – How to Configure SNMP Community Strings

```
(config)# snmp-server community 244Lab RW
(config)# snmp-server contact Dennis Rice
(config)# snmp-server location 244 Lab Rack 203 PRTR01
```

With the above commands, we have specified the following:

1. Community Name as 244Lab and given it read-write privileges. We could alternatively specified **RO**, which would have specified read-only privileges.
2. System Contact as Dennis Rice. This could also include additional information up to 255 characters in length.
3. System Location as 244Lab with equipment located in rack 203, labeled PRTR01.

16.1.2.2.2 Cisco Catalyst 2950 LAN Switch

After telneting into the 2950 switch, issue the following commands to configure it to support snmp:

```
> enable
password your-password
# config t
(config)# snmp-server community 244Lab RW
(config)# snmp-server contact Dennis Rice
(config)# snmp-server location 244 Lab Rack 203 PSW01
```

16.1.2.2.3 Cisco 3600 Series Router

After telneting into the 3640 router, issue the following commands to configure it to support snmp:

```
> enable
password your-password
# config t
(config)# snmp-server community 244Lab RW
(config)# snmp-server contact Dennis Rice
(config)# snmp-server location 244 Lab Rack 201 WRTR01
```

16.1.2.2.4 Cisco Catalyst 8500 Series ATM Switch (?)

After telneting into the 2950 switch, issue the following commands to configure it to support snmp:

```
> enable
password your-password
# config t
(config)# snmp-server community 244Lab RW
(config)# snmp-server contact Dennis Rice
(config)# snmp-server location 244 Lab Rack 201 PSW01
```

16.1.2.3 snmp Data Collection from Cisco

After the Cisco equipment has been configured, the following commands may be issued:

\$ snmpwalk 1720a -c 244Lab system

(It is assumed that an entry in the `/etc/hosts` file has been made for the 1720a router.)

The response will list:

1. Unit Description
2. System ObjectID:
3. System Uptime:
4. System Contact: Dennis Rice
5. System Name: 1720a
6. System Location: 244 Lab Rk 203 PRTR01
7. System Services:
8. System OR Last Change:

Now that you have a basic operational snmp agent and monitor, you have the duty of doing additional research as to all of the MIB information that is available. You even have the opportunity to figure out what it means.

16.2 snmp Analysis Programs

Several programs are available to analyze the data received from snmp, which perform the function of creating various graphics and analysis.

16.2.1 nino

Nino, which stands for “nino is not openview”, is very much an “OpenView” look-alike program. It has the ability to display various graphics indicating host performance and network traffic.

16.2.2 Multi-Router Traffic Grapher⁵

The **Multi-Router Traffic Grapher**, or **mrtg**, uses snmp to create a graphic output for analysis. By using specific commands and collecting the data over a period of time, the program is able to plot the performance of a system on the screen of a monitor.

⁵ <http://people.ee.ethz.ch/~oetiker/webtools/mrtg/>

16.2.3 OpenNMS ⁶

OpenNMS is an open source equivalent to HP OpenView. OpenView is an excellent program designed to run on various Unix and MS Windows systems that produces a graphical display of a network. If a specific system has a problem, the color of the system will change from green (everything working ok), to yellow (non-service effecting problem), to red (system outage or major problem). The administrator may then click on the object and query its status through mouse clicks rather than having to issue commands and interpret the results. So what is the big difference between the two – HP OpenView has a fairly high price tag.

16.X Commands Used in this Chapter

chkconfig	Checks for specified service operation.
cp	Copies a file.
service	Used to restart a specified service.
snmpconf	Used to create a new snmp.conf, snmpd.conf, or snmptrap.conf file.
snmpget	Used to obtain an agent's snmp value.
snmpgetnext	Used to obtain the next snmp value from an agent.
snmpset	Used to set an snmp value on an agent.
snmptable	Used to obtain a set of snmp values in as a formatted table.
snmptranslate	Used to translate a MIB between the verbal and numeric format.
snmpwalk	Used to obtain a set of snmp values from an agent.

16.X Chapter Review Questions

⁶ <http://www.opennms.org/>

Chapter Index

A		N	
Application		net-snmp	
net-snmp	3	Configuration Files	4
C		O	
Cisco		Object Identifier	29
1720 snmp agent	47	Object Identifier	3
2950 snmp agent	48	OpenNMS	50
3600 snmp agent	48	S	
8510 snmp agent	48	Simple Network Monitoring Protocol	3
D		snmp	
Directory		agent	3
/etc/snmp	4, 12	Cisco 1720 Router	47
/usr/bin	4	Cisco 2950 Switch	48
/usr/local/share/snmp/mibs	25, 38,	Cisco 3600 Router	48
41		Cisco 8510 ATM Switch	48
/usr/share/snmp	12	Linux RH9	43
F		commands	28
File		Configuring Cisco agent	47
/etc/hosts	29, 49	Making it work	43
/etc/snmp/snmpd.conf	29	mib files \	25
M		monitor	3
man		snmp.conf File Example	23
snmpcmd	28	snmpconf script	5, 12
snmpgetnext	32	snmpd.conf File Example	9
snmpset	36	trap	3
snmptable	34	snmp Analysis Programs	49
snmptranslate	30	snmp Command	
snmpwalk	33	Community designator	29
mib	3	Input Options	29
MIB assignment	26	Output Options	29
MIB File		Table Format Options	29
BEGIN	38	snmptrap	37
IMPORTS	39	U	
MODULE INENTITY	39	URL	
OBJECT IDENTIFIER	40	net-snmp.org	3
OBJECT TYPES	40	Utility	
MIB Files		chkconfig	3
MIB Groups	40	nino	49
Required Vendor MIB Files	41	snmpget	31
MIB Numbering	42	snmpgetnext	32
MIBs		snmpset	36
Structure	26	snmptable	34
Multi-Router Traffic Grapher	49	snmptranslate	30
		snmpwalk	33