SNMP Operations

We've discussed how SNMP organizes information, but we've left out how we actually go about gathering management information. Now we're going to take a look under the hood to see how SNMP does its thing.

The Protocol Data Unit (PDU) is the message format that managers and agents use to send and receive information. Each of the following SNMP operations has a standard PDU format:

- get
- getnext
- getbulk (SNMPv2 and SNMPv3)
- set
- getresponse
- trap
- notification (SNMPv2 and SNMPv3)
- inform (SNMPv2 and SNMPv3)
- report (SNMPv2 and SNMPv3)

In addition to running actual command-line tools, we will also provide packet dumps of the SNMP operations. For those of you who like looking at packet dumps, this will give you an inside look at what the packet structure is for each command. The packet dumps themselves were taken using the command-line version of Ethereal (http://www.ethereal.com). Let's take a look at each operation now. All of the get and set operations were captured with the following command:

```
$ /usr/sbin/tethereal -i lo -x -V -F libpcap -f "port 161"
```

Traps and notifications were captured with this command:

```
$ /usr/sbin/tethereal -i lo -x -V -F libpcap -f "port 162"
```

The get Operation

The get request is initiated by the NMS, which sends the request to the agent. The agent receives the request and processes it to the best of its ability. Some devices that are under heavy load, such as routers, may not be able to respond to the request and will have to drop it. If the agent is successful in gathering the requested information, it sends a getresponse back to the NMS, where it is processed. This process is illustrated in Figure 2-5.

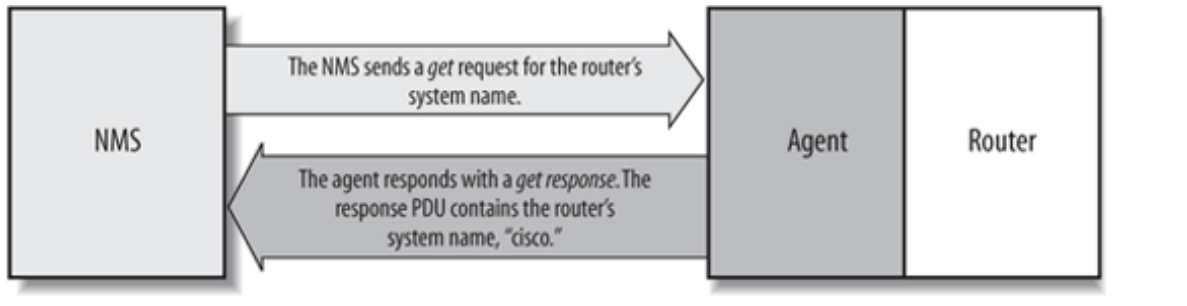

Figure 2-5. get request sequence

How did the agent know what the NMS was looking for? One of the items in the get request is a variable binding. A *variable binding*, or *varbind*, is a list of MIB objects that allows a request's recipient to see what the originator wants to know. Variable bindings can be thought of as *OID=value* pairs that make it easy for the originator (the NMS, in this case) to pick out the information it needs when the recipient fills the request and sends back a response. Let's look at this operation in action:

```
$ snmpget cisco.ora.com public .1.3.6.1.2.1.1.6.0
system.sysLocation.0 = ""
```

> **TIP**
>
> All the Unix commands presented in this chapter come from the Net-SNMP agent package (formerly the UCD-SNMP project), a freely available Unix and Windows agent. You can download the package from http://net-snmp.sourceforge.net. Appendix C summarizes the commands in this package.

Several things are going on in this example. First, we're running a command on a Unix host. The command is called snmpget. Its main job is to facilitate the gathering of management data using a get request. We've given it three arguments on the command line: the name of the device we would like to query (cisco.ora.com), the read-only community string (*public*), and the OID we would like gathered (*.1.3.6.1.2.1.1.6.0*). If we look back at Table 2-5, we see that *1.3.6.1.2.1.1* is the *system* group, but there are two more integers at the end of the OID: *.6* and *.0*. The *.6* is actually the MIB variable that we wish to query; its human-readable name is *sysLocation*. In this case, we would like to see what the system location is set to on the Cisco router. As you can see by the response (`system.sysLocation.0 = ""`), the system location on this router currently is not set to anything. Also note that the response from snmpget is in variable binding format, *OID=value*.

There is one more thing to look at. Why does the MIB variable have a *.0* tacked on the end? In SNMP, MIB objects are defined by the convention *x.y*, where *x* is the actual OID of the managed object (in our example, *1.3.6.1.2.1.1.6*) and *y* is the instance identifier. For *scalar objects* (that is, objects that aren't defined as a row in a table), *y* is always 0. In the case of a table, the instance identifier lets you select a specific row of the table; 1 is the first row, 2 is the second row, etc. For example, consider the *ifTable* object we looked at earlier in this chapter. When looking up values in *ifTable*, we would use a nonzero instance identifier to select a particular row in the table (in this case, a particular network interface).

> **TIP**
>
> Graphical NMS applications , which include most commercial packages, do not use command-line programs to retrieve management information. We use these commands to give you a feel for how the retrieval commands work and what they typically return. The information a graphical NMS retrieves and its retrieval process are identical to these command-line programs; the NMS just lets you formulate queries and displays the results using a more convenient GUI.

The get command is useful for retrieving a single MIB object at a time. Trying to manage anything in this manner can be a waste of time, though. This is where the getnext command comes in. It allows you to retrieve more than one object from a device, over a period of time.

Now let's look at an SNMP packet as seen with Ethereal's command-line tool tethereal. Given the following command:

```
$ snmpget -v 1 -c public 127.0.0.1 sysContact.0
```

we get the following two datagram traces from tethereal:

```
Frame 1 (85 bytes on wire, 85 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:15.041115000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 85 bytes
    Capture Length: 85 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
```

```
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 71
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7ca4 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: 34066 (34066), Dst Port: snmp (161)
    Source port: 34066 (34066)
    Destination port: snmp (161)
    Length: 51
    Checksum: 0xfe46 (incorrect, should be 0xbbea)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: GET (0)
    Request Id: 0x20a71b4c
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.4.0 (SNMPv2-MIB::sysContact.0)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 47 00 00 40 00 00 11 7c a4 7f 00 00 01 7f 00   .G..@...|.......
0020  00 01 85 12 00 a1 00 33 fe 46 30 29 02 01 00 04   .......3.F0)....
0030  06 70 75 62 6c 69 63 a0 1c 02 04 20 a7 1b 4c 02   .public.... ..L.
0040  01 00 02 01 00 30 0e 30 0c 06 08 2b 06 01 02 01   .....0.0...+....
0050  01 04 00 05 00                                     .....

Frame 2 (144 bytes on wire, 144 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:15.071891000
    Time delta from previous packet: 0.030776000 seconds
    Time since reference or first frame: 0.030776000 seconds
    Frame Number: 2
    Packet Length: 144 bytes
    Capture Length: 144 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
```

```
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 130
    Identification: 0x031d (797)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x794c (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34066 (34066)
    Source port: snmp (161)
    Destination port: 34066 (34066)
    Length: 110
    Checksum: 0xfe81 (incorrect, should be 0xdf61)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x20a71b4c
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.4.0 (SNMPv2-MIB::sysContact.0)
    Value: STRING: Root <root@localhost> (configure /etc/snmp/snmp.local.conf)

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 82 03 1d 40 00 00 11 79 4c 7f 00 00 01 7f 00   ....@...yL......
0020  00 01 00 a1 85 12 00 6e fe 81 30 64 02 01 00 04   .......n..0d....
0030  06 70 75 62 6c 69 63 a2 57 02 04 20 a7 1b 4c 02   .public.W.. ..L.
0040  01 00 02 01 00 30 49 30 47 06 08 2b 06 01 02 01   .....0I0G..+....
0050  01 04 00 04 3b 52 6f 6f 74 20 3c 72 6f 6f 74 40   ....;Root <root@
0060  6c 6f 63 61 6c 68 6f 73 74 3e 20 28 63 6f 6e 66   localhost> (conf
0070  69 67 75 72 65 20 2f 65 74 63 2f 73 6e 6d 70 2f   igure /etc/snmp/
0080  73 6e 6d 70 2e 6c 6f 63 61 6c 2e 63 6f 6e 66 29   snmp.local.conf)
```

There are two frames, each labeled appropriately. Frame 1 is initiated by the client. Frame 2 is the agent's response. Ethereal is nice in that it tells us the version of SNMP in use, and the error code (defined later in this chapter in Table 2-6 and Table 2-7). Giving the following command:

```
$ snmpget -v 2c -c public 127.0.0.1 sysContact.0
```

we see the following output from tethereal:

```
Frame 1 (85 bytes on wire, 85 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:26.129733000
    Time delta from previous packet: 0.000000000 seconds
```

```
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 85 bytes
    Capture Length: 85 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 71
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7ca4 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: 34066 (34066), Dst Port: snmp (161)
    Source port: 34066 (34066)
    Destination port: snmp (161)
    Length: 51
    Checksum: 0xfe46 (incorrect, should be 0xbb8f)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: GET (0)
    Request Id: 0x175f7f93
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.4.0 (SNMPv2-MIB::sysContact.0)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 47 00 00 40 00 00 11 7c a4 7f 00 00 01 7f 00   .G..@...|.......
0020  00 01 85 12 00 a1 00 33 fe 46 30 29 02 01 01 04   .......3.F0)....
0030  06 70 75 62 6c 69 63 a0 1c 02 04 17 5f 7f 93 02   .public....._...
0040  01 00 02 01 00 30 0e 30 0c 06 08 2b 06 01 02 01   .....0.0...+....
0050  01 04 00 05 00                                     .....

Frame 2 (144 bytes on wire, 144 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:26.129926000
    Time delta from previous packet: 0.000193000 seconds
```

```
    Time since reference or first frame: 0.000193000 seconds
    Frame Number: 2
    Packet Length: 144 bytes
    Capture Length: 144 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 130
    Identification: 0x031e (798)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x794b (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34066 (34066)
    Source port: snmp (161)
    Destination port: 34066 (34066)
    Length: 110
    Checksum: 0xfe81 (incorrect, should be 0xdf06)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x175f7f93
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.4.0 (SNMPv2-MIB::sysContact.0)
    Value: STRING: Root <root@localhost> (configure /etc/snmp/snmp.local.conf)

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 82 03 1e 40 00 00 11 79 4b 7f 00 00 01 7f 00   ....@...yK......
0020  00 01 00 a1 85 12 00 6e fe 81 30 64 02 01 01 04   .......n..0d....
0030  06 70 75 62 6c 69 63 a2 57 02 04 17 5f 7f 93 02   .public.W..._...
0040  01 00 02 01 00 30 49 30 47 06 08 2b 06 01 02 01   .....0I0G..+....
0050  01 04 00 04 3b 52 6f 6f 74 20 3c 72 6f 6f 74 40   ....;Root <root@
0060  6c 6f 63 61 6c 68 6f 73 74 3e 20 28 63 6f 6e 66   localhost> (conf
0070  69 67 75 72 65 20 2f 65 74 63 2f 73 6e 6d 70 2f   igure /etc/snmp/
0080  73 6e 6d 70 2e 6c 6f 63 61 6c 2e 63 6f 6e 66 29   snmp.local.conf)
```

The datagram traces look similar to the SNMPv1 traces. Again, we see the version of SNMP in use, namely 2C.

The getnext Operation

The getnext operation lets you issue a sequence of commands to retrieve a group of values from a MIB. In other words, for each MIB object we want to retrieve, a separate getnext request and getresponse are generated. The getnext command traverses a subtree in lexicographic order. Since an OID is a sequence of integers, it's easy for an agent to start at the root of its SMI object tree and work its way down until it finds the OID it is looking for. This form of searching is called depth-first. When the NMS receives a response from the agent for the getnext command it just issued, it issues another getnext command. It keeps doing this until the agent returns an error, signifying that the end of the MIB has been reached and there are no more objects left to get.

If we look at another example, we can see this behavior in action. This time we'll use a command called snmpwalk. This command simply facilitates the getnext procedure for us. It's invoked just like the snmpget command, except this time we specify which branch to start at (in this case, the *system* group):

```
$ snmpwalk cisco.ora.com public system

system.sysDescr.0 = "Cisco IOS Software, C2600 Software (C2600-IPBASE-M),

Version 12.3(8)T3, RELEASE SOFTWARE (fc1)

Technical Support: http://www.cisco.com/techsupport

Copyright (c) 1986-2004 by Cisco Systems, Inc.

Compiled Tue 20-Jul-04 17:03 by eaarmas"

system.sysObjectID.0 = OID: enterprises.9.1.19

system.sysUpTime.0 = Timeticks: (27210723) 3 days, 3:35:07.23

system.sysContact.0 = ""

system.sysName.0 = "cisco.ora.com"

system.sysLocation.0 = ""

system.sysServices.0 = 6
```

The getnext sequence returns seven MIB variables. Each object is part of the *system* group as it's defined in RFC 1213. We see a system object ID, the amount of time the system has been up, the contact person, etc.

Given that you've just looked up some object, how does getnext figure out which object to look up next? getnext is based on the concept of the lexicographic ordering of the MIB's object tree. This order is made much simpler because every node in the tree is assigned a number. To understand what this means, let's start at the root of the tree and walk down to the *system* node.

To get to the *system* group (OID *1.3.6.1.2.1.1*), we start at the root of the object tree and work our way down. Figure 2-6 shows the logical progression from the root of the tree all the way to the *system* group. At each node in the tree, we visit the lowest numbered branch. Thus, when we're at the root node, we start by visiting *ccitt*. This node has no nodes underneath it, so we move to the *iso* node. Since *iso* does have a child, we move to that node, *org*. The process continues until we reach the *system* node. Since each branch is made up of ascending integers (*ccitt(0) iso(1) join(2)*, for example), the agent has no problem traversing this tree structure all the way down to the *system(1)* group. If we were to continue this walk, we'd proceed to *system.1* (*system.sysLocation*), *system.2*, and the other objects in the *system* group. Next, we'd go to *interfaces(2)*, and so on.

Now let's look at what Ethereal sees. Given the following command:

```
$ snmpwalk -v 1 -c public 127.0.0.1 system
```

we get the following output from tethereal:

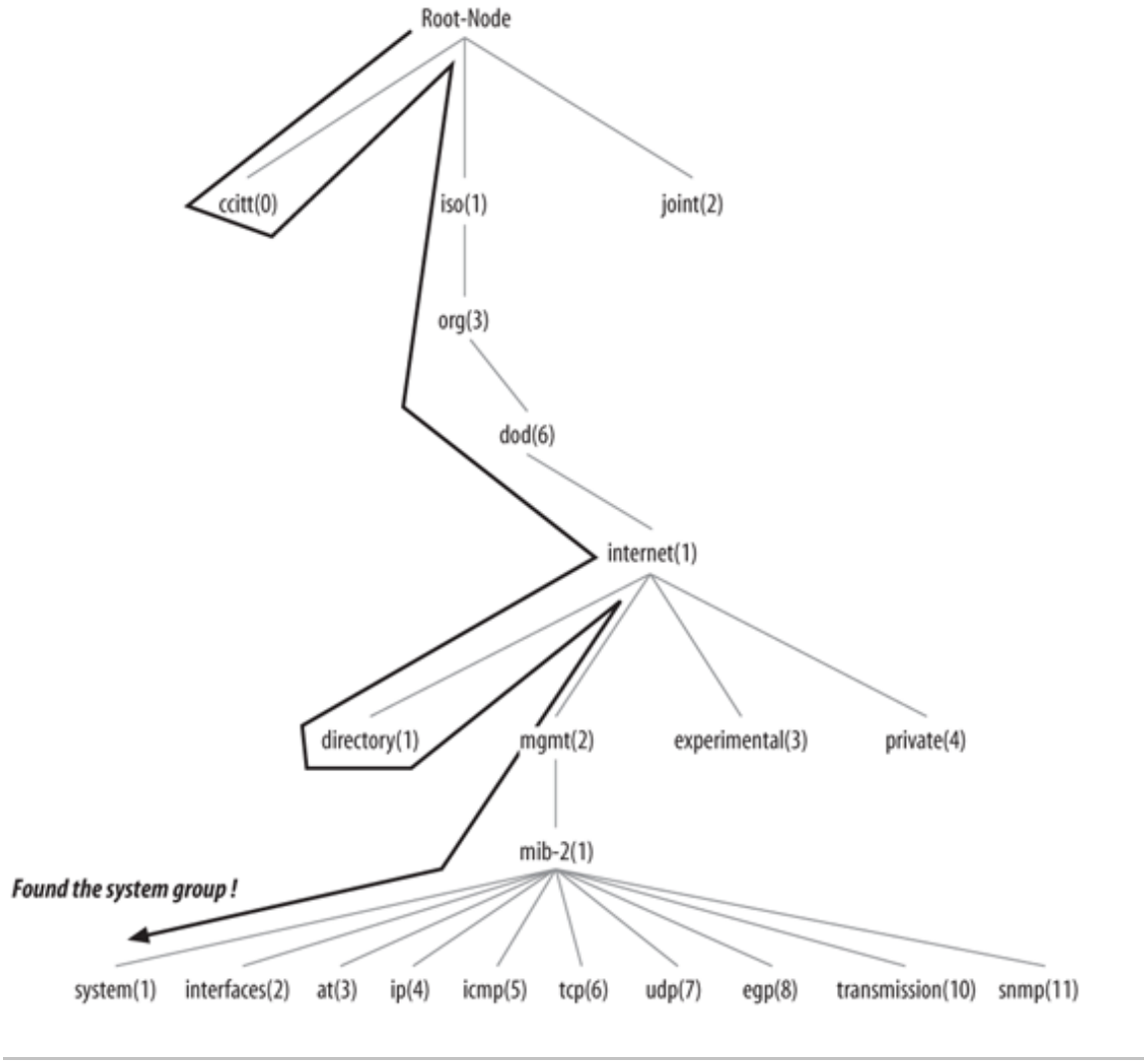

Figure 2-6. Walking the MIB tree

```
Frame 1 (82 bytes on wire, 82 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:53.598461000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 82 bytes
    Capture Length: 82 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253
(10.0.1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
```

```
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 68
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x62b0 (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: 34069 (34069), Dst Port: snmp (161)
    Source port: 34069 (34069)
    Destination port: snmp (161)
    Length: 48
    Checksum: 0x183b (incorrect, should be 0xa7f4)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: GET-NEXT (1)
    Request Id: 0x00ec4809
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1 (SNMPv2-SMI::mib-2)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 44 00 00 40 00 00 11 62 b0 0a 00 01 fd 0a 00   .D..@...b.......
0020  01 fd 85 15 00 a1 00 30 18 3b 30 26 02 01 00 04   .......0.;0&....
0030  06 70 75 62 6c 69 63 a1 19 02 04 00 ec 48 09 02   .public......H..
0040  01 00 02 01 00 30 0b 30 09 06 05 2b 06 01 02 01   .....0.0...+....
0050  05 00                                             ..

Frame 2 (160 bytes on wire, 160 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:53.598662000
    Time delta from previous packet: 0.000201000 seconds
    Time since reference or first frame: 0.000201000 seconds
    Frame Number: 2
    Packet Length: 160 bytes
    Capture Length: 160 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.
0.1.253)
    Version: 4
    Header length: 20 bytes
```

```
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 146
    Identification: 0x031f (799)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x5f43 (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34069 (34069)
    Source port: snmp (161)
    Destination port: 34069 (34069)
    Length: 126
    Checksum: 0x1889 (incorrect, should be 0x3f0a)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x00ec4809
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.1.0 (SNMPv2-MIB::sysDescr.0)
    Value: STRING: Linux mailworks.guarded.net 2.4.21-4.EL #1 Fri Oct 3 18:13:58 EDT 2003
    i686

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 92 03 1f 40 00 00 11 5f 43 0a 00 01 fd 0a 00   ....@..._C......
0020  01 fd 00 a1 85 15 00 7e 18 89 30 74 02 01 00 04   .......~..0t....
0030  06 70 75 62 6c 69 63 a2 67 02 04 00 ec 48 09 02   .public.g....H..
0040  01 00 02 01 00 30 59 30 57 06 08 2b 06 01 02 01   .....0Y0W..+....
0050  01 01 00 04 4b 4c 69 6e 75 78 20 6d 61 69 6c 77   ....KLinux mailw
0060  6f 72 6b 73 2e 67 75 61 72 64 65 64 2e 6e 65 74   orks.guarded.net
0070  20 32 2e 34 2e 32 31 2d 34 2e 45 4c 20 23 31 20    2.4.21-4.EL #1
0080  46 72 69 20 4f 63 74 20 33 20 31 38 3a 31 33 3a   Fri Oct 3 18:13:
0090  35 38 20 45 44 54 20 32 30 30 33 20 69 36 38 36   58 EDT 2003 i686

Frame 3 (85 bytes on wire, 85 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:53.682655000
    Time delta from previous packet: 0.083993000 seconds
    Time since reference or first frame: 0.084194000 seconds
    Frame Number: 3
    Packet Length: 85 bytes
    Capture Length: 85 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
```

```
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.0.
1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 71
    Identification: 0x0001 (1)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x62ac (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: 34069 (34069), Dst Port: snmp (161)
    Source port: 34069 (34069)
    Destination port: snmp (161)
    Length: 51
    Checksum: 0x183e (incorrect, should be 0x9ee5)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: GET-NEXT (1)
    Request Id: 0x00ec480a
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.1.0 (SNMPv2-MIB::sysDescr.0)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 47 00 01 40 00 00 11 62 ac 0a 00 01 fd 0a 00   .G..@...b.......
0020  01 fd 85 15 00 a1 00 33 18 3e 30 29 02 01 00 04   .......3.>0)....
0030  06 70 75 62 6c 69 63 a1 1c 02 04 00 ec 48 0a 02   .public......H..
0040  01 00 02 01 00 30 0e 30 0c 06 08 2b 06 01 02 01   .....0.0...+....
0050  01 01 00 05 00                                    .....

Frame 4 (95 bytes on wire, 95 bytes captured)
    Arrival Time: Sep 20, 2004 13:46:53.682855000
    Time delta from previous packet: 0.000200000 seconds
    Time since reference or first frame: 0.084394000 seconds
    Frame Number: 4
    Packet Length: 95 bytes
    Capture Length: 95 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
```

```
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.0.
1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 81
    Identification: 0x0320 (800)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x5f83 (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34069 (34069)
    Source port: snmp (161)
    Destination port: 34069 (34069)
    Length: 61
    Checksum: 0x1848 (incorrect, should be 0xa08c)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x00ec480a
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.2.0 (SNMPv2-MIB::sysObjectID.0)
    Value: OID: SNMPv2-SMI::enterprises.8072.3.2.10


0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 51 03 20 40 00 00 11 5f 83 0a 00 01 fd 0a 00   .Q. @..._.......
0020  01 fd 00 a1 85 15 00 3d 18 48 30 33 02 01 00 04   .......=.H03....
0030  06 70 75 62 6c 69 63 a2 26 02 04 00 ec 48 0a 02   .public.&....H..
0040  01 00 02 01 00 30 18 30 16 06 08 2b 06 01 02 01   .....0.0...+....
0050  01 02 00 06 0a 2b 06 01 04 01 bf 08 03 02 0a      .....+.........
```

To save space, we included only the first four frames, which are the first two getnext operations. As before, frames 1 and 3 are client requests and frames 2 and 4 are the agent's responses. Now let's look at SNMPv2's datagram traces (again we kept it short):

```
Frame 1 (82 bytes on wire, 82 bytes captured)
    Arrival Time: Sep 20, 2004 13:47:06.413352000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
```

```
    Frame Number: 1
    Packet Length: 82 bytes
    Capture Length: 82 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.0.
1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 68
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x62b0 (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: 34069 (34069), Dst Port: snmp (161)
    Source port: 34069 (34069)
    Destination port: snmp (161)
    Length: 48
    Checksum: 0x183b (incorrect, should be 0xa1af)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: GET-NEXT (1)
    Request Id: 0x75cb182f
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1 (SNMPv2-SMI::mib-2)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 44 00 00 40 00 00 11 62 b0 0a 00 01 fd 0a 00   .D..@...b.......
0020  01 fd 85 15 00 a1 00 30 18 3b 30 26 02 01 01 04   .......0.;0&....
0030  06 70 75 62 6c 69 63 a1 19 02 04 75 cb 18 2f 02   .public....u../.
0040  01 00 02 01 00 30 0b 30 09 06 05 2b 06 01 02 01   .....0.0...+....
0050  05 00                                             ..

Frame 2 (160 bytes on wire, 160 bytes captured)
    Arrival Time: Sep 20, 2004 13:47:06.413554000
    Time delta from previous packet: 0.000202000 seconds
```

```
    Time since reference or first frame: 0.000202000 seconds
    Frame Number: 2
    Packet Length: 160 bytes
    Capture Length: 160 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.0.
1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 146
    Identification: 0x0342 (834)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x5f20 (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34069 (34069)
    Source port: snmp (161)
    Destination port: 34069 (34069)
    Length: 126
    Checksum: 0x1889 (incorrect, should be 0x38c5)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x75cb182f
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.1.0 (SNMPv2-MIB::sysDescr.0)
    Value: STRING: Linux mailworks.guarded.net 2.4.21-4.EL #1 Fri Oct 3 18:13:58 EDT 2003
    i686

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 92 03 42 40 00 00 11 5f 20 0a 00 01 fd 0a 00   ...B@..._ ......
0020  01 fd 00 a1 85 15 00 7e 18 89 30 74 02 01 01 04   .......~..0t....
0030  06 70 75 62 6c 69 63 a2 67 02 04 75 cb 18 2f 02   .public.g..u../.
0040  01 00 02 01 00 30 59 30 57 06 08 2b 06 01 02 01   .....0Y0W..+....
0050  01 01 00 04 4b 4c 69 6e 75 78 20 6d 61 69 6c 77   ....KLinux mailw
0060  6f 72 6b 73 2e 67 75 61 72 64 65 64 2e 6e 65 74   orks.guarded.net
0070  20 32 2e 34 2e 32 31 2d 34 2e 45 4c 20 23 31 20    2.4.21-4.EL #1
```

```
0080  46 72 69 20 4f 63 74 20 33 20 31 38 3a 31 33 3a   Fri Oct 3 18:13:
0090  35 38 20 45 44 54 20 32 30 30 33 20 69 36 38 36   58 EDT 2003 i686

Frame 3 (85 bytes on wire, 85 bytes captured)
    Arrival Time: Sep 20, 2004 13:47:06.495596000
    Time delta from previous packet: 0.082042000 seconds
    Time since reference or first frame: 0.082244000 seconds
    Frame Number: 3
    Packet Length: 85 bytes
    Capture Length: 85 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.0.
1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 71
    Identification: 0x0001 (1)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x62ac (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: 34069 (34069), Dst Port: snmp (161)
    Source port: 34069 (34069)
    Destination port: snmp (161)
    Length: 51
    Checksum: 0x183e (incorrect, should be 0x98a0)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: GET-NEXT (1)
    Request Id: 0x75cb1830
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.1.0 (SNMPv2-MIB::sysDescr.0)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 47 00 01 40 00 00 11 62 ac 0a 00 01 fd 0a 00   .G..@...b.......
0020  01 fd 85 15 00 a1 00 33 18 3e 30 29 02 01 01 04   .......3.>0)....
```

```
0030  06 70 75 62 6c 69 63 a1 1c 02 04 75 cb 18 30 02   .public....u..0.
0040  01 00 02 01 00 30 0e 30 0c 06 08 2b 06 01 02 01   .....0.0...+....
0050  01 01 00 05 00                                     .....

Frame 4 (95 bytes on wire, 95 bytes captured)
    Arrival Time: Sep 20, 2004 13:47:06.495794000
    Time delta from previous packet: 0.000198000 seconds
    Time since reference or first frame: 0.082442000 seconds
    Frame Number: 4
    Packet Length: 95 bytes
    Capture Length: 95 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 10.0.1.253 (10.0.1.253), Dst Addr: 10.0.1.253 (10.0.
1.253)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 81
    Identification: 0x0343 (835)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x5f60 (correct)
    Source: 10.0.1.253 (10.0.1.253)
    Destination: 10.0.1.253 (10.0.1.253)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34069 (34069)
    Source port: snmp (161)
    Destination port: 34069 (34069)
    Length: 61
    Checksum: 0x1848 (incorrect, should be 0x9a47)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x75cb1830
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.2.0 (SNMPv2
-MIB::sysObjectID.0)
    Value: OID: SNMPv2-SMI::enterprises.8072.3.2.10

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
```

```
0010  00 51 03 43 40 00 00 11 5f 60 0a 00 01 fd 0a 00   .Q.C@..._'......

0020  01 fd 00 a1 85 15 00 3d 18 48 30 33 02 01 01 04   .......=.H03....

0030  06 70 75 62 6c 69 63 a2 26 02 04 75 cb 18 30 02   .public.&..u..0.

0040  01 00 02 01 00 30 18 30 16 06 08 2b 06 01 02 01   .....0.0...+....

0050  01 02 00 06 0a 2b 06 01 04 01 bf 08 03 02 0a      .....+.........
```

The getbulk Operation

SNMPv2 defines the getbulk operation, which allows a management application to retrieve a large section of a table at once. The standard get operation can attempt to retrieve more than one MIB object at once, but message sizes are limited by the agent's capabilities. If the agent can't return all the requested responses, it returns an error message with no data. The getbulk operation, on the other hand, tells the agent to send back as much of the response as it can. This means that incomplete responses are possible. Two fields must be set when issuing a getbulk command: nonrepeaters and max-repetitions. Nonrepeaters tell the getbulk command that the first *N* objects can be retrieved with a simple getnext operation. max-repetitions tells the getbulk command to attempt up to *M* getnext operations to retrieve the remaining objects. Figure 2-7 shows the getbulk command sequence.

In Figure 2-7, we're requesting three bindings: *sysDescr*, *ifInOctets*, and *ifOutOctets*. The total number of variable bindings that we've requested is given by the formula *N* + *(M * R)*, where *N* is the number of nonrepeaters (i.e., scalar objects in the request—in this case, 1 because *sysDescr* is the only scalar object), *M* is max-repetitions (in this

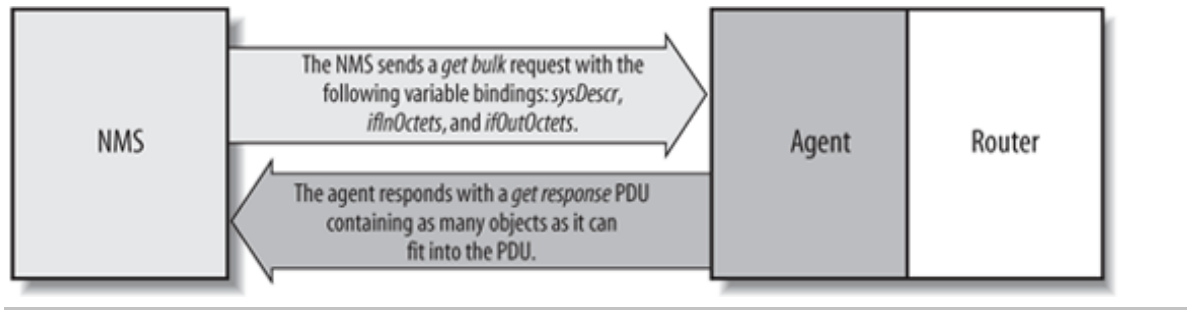

Figure 2-7. getbulk request sequence

case, we've set it arbitrarily to 3), and *R* is the number of nonscalar objects in the request (in this case, 2 because *ifInOctets* and *ifOutOctets* are both nonscalar). Plugging in the numbers from this example, we get 1 + (3 * 2) = 7, which is the total number of variable bindings that can be returned by this getbulk request.

The Net-SNMP package comes with a command for issuing getbulk queries . If we execute this command using all the parameters previously discussed, it will look like the following:

```
$ snmpbulkget -v2c -c public -Cn1 -Cr3 linux.ora.com sysDescr ifInOctets ifOutOctets

system.sysDescr.0 = " Linux snort 2.4.7-10 #1 Thu Sep 6 17:27:27 EDT 2001

i686 unknown "

interfaces.ifTable.ifEntry.ifInOctets.1 = 70840

interfaces.ifTable.ifEntry.ifOutOctets.1 = 70840

interfaces.ifTable.ifEntry.ifInOctets.2 = 143548020

interfaces.ifTable.ifEntry.ifOutOctets.2 = 111725152

interfaces.ifTable.ifEntry.ifInOctets.3 = 0

interfaces.ifTable.ifEntry.ifOutOctets.3 = 0
```

Since getbulk is an SNMPv2 command, you have to tell snmpbulkget to use an SNMPv2 PDU with the -v2c option. nonrepeaters and max-repetitions are set with the -Cn1 and -Cr3 options. This sets nonrepeaters to 1 and max-repetitions to 3. Notice that the command returned seven variable bindings: one for *sysDescr* and three each for *ifInOctets* and *ifOutOctets*.

Now let's look at a trace. If we use the following command:

```
$ ./snmpbulkget -v2c -Cn1 -Cr2 127.0.0.1 -c public sysDescr sysContact
```

we get the following trace:

```
Frame 1 (97 bytes on wire, 97 bytes captured)
    Arrival Time: Sep 20, 2004 20:24:19.106374000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 97 bytes
    Capture Length: 97 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 83
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7c98 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: 34193 (34193), Dst Port: snmp (161)
    Source port: 34193 (34193)
    Destination port: snmp (161)
    Length: 63
    Checksum: 0xfe52 (incorrect, should be 0x0c90)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: GETBULK (5)
    Request Id: 0x0f15c607
    Non-repeaters: 1
    Max repetitions: 2
    Object identifier 1: 1.3.6.1.2.1.1.1 (SNMPv2-MIB::sysDescr)
    Value: NULL
    Object identifier 2: 1.3.6.1.2.1.1.4 (SNMPv2-MIB::sysContact)
    Value: NULL

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 53 00 00 40 00 00 11 7c 98 7f 00 00 01 7f 00   .S..@...|.......
0020  00 01 85 91 00 a1 00 3f fe 52 30 35 02 01 01 04   .......?.R05....
```

```
0030  06 70 75 62 6c 69 63 a5 28 02 04 0f 15 c6 07 02   .public.(.......
0040  01 01 02 01 02 30 1a 30 0b 06 07 2b 06 01 02 01   .....0.0...+....
0050  01 01 05 00 30 0b 06 07 2b 06 01 02 01 01 04 05   ....0...+.......
0060  00                                                .

Frame 2 (211 bytes on wire, 211 bytes captured)
    Arrival Time: Sep 20, 2004 20:24:19.151924000
    Time delta from previous packet: 0.045550000 seconds
    Time since reference or first frame: 0.045550000 seconds
    Frame Number: 2
    Packet Length: 211 bytes
    Capture Length: 211 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 197
    Identification: 0x0052 (82)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7bd4 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34193 (34193)
    Source port: snmp (161)
    Destination port: 34193 (34193)
    Length: 177
    Checksum: 0xfec4 (incorrect, should be 0x47bb)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: RESPONSE (2)
    Request Id: 0x0f15c607
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.1.0 (SNMPv2-MIB::sysDescr.0)
    Value: STRING: Linux mailworks.guarded.net 2.4.21-4.EL #1 Fri Oct 3 18:13:58 EDT 2003
    i686
    Object identifier 2: 1.3.6.1.2.1.1.4.0 (SNMPv2-MIB::sysContact.0)
    Value: STRING: "kjs@guarded.net"
```

```
    Object identifier 3: 1.3.6.1.2.1.1.5.0 (SNMPv2-MIB::sysName.0)
    Value: STRING: box


0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 c5 00 52 40 00 00 11 7b d4 7f 00 00 01 7f 00   ...R@...{.......
0020  00 01 00 a1 85 91 00 b1 fe c4 30 81 a6 02 01 01   ..........0.....
0030  04 06 70 75 62 6c 69 63 a2 81 98 02 04 0f 15 c6   ..public........
0040  07 02 01 00 02 01 00 30 81 89 30 57 06 08 2b 06   .......0..0W..+.
0050  01 02 01 01 01 00 04 4b 4c 69 6e 75 78 20 6d 61   .......KLinux ma
0060  69 6c 77 6f 72 6b 73 2e 67 75 61 72 64 65 64 2e   ilworks.guarded.
0070  6e 65 74 20 32 2e 34 2e 32 31 2d 34 2e 45 4c 20   net 2.4.21-4.EL
0080  23 31 20 46 72 69 20 4f 63 74 20 33 20 31 38 3a   #1 Fri Oct 3 18:
0090  31 33 3a 35 38 20 45 44 54 20 32 30 30 33 20 69   13:58 EDT 2003 i
00a0  36 38 36 30 1d 06 08 2b 06 01 02 01 01 04 00 04   6860...+........
00b0  11 22 6b 6a 73 40 67 75 61 72 64 65 64 2e 6e 65   ."kjs@guarded.ne
00c0  74 22 30 0f 06 08 2b 06 01 02 01 01 05 00 04 03   t"0...+.........
00d0  62 6f 78                                         box
```

The set Operation

The set command is used to change the value of a managed object or to create a new row in a table. Objects that are defined in the MIB as read-write or read-only can be altered or created using this command. It is possible for an NMS to set more than one object at a time.

Figure 2-8 shows the set request sequence. It's similar to the other commands we've seen so far, but it actually changes something in the device's configuration as opposed to just retrieving a response to a query. Let's look at the set command in action. The following example queries the *sysLocation* variable and sets it to a value:

```
$ snmpget cisco.ora.com public system.sysLocation.0
system.sysLocation.0 = ""
$ snmpset
 cisco.ora.com private system.sysLocation.0 s "Atlanta, GA"
system.sysLocation.0 = "Atlanta, GA"
$ snmpget cisco.ora.com public system.sysLocation.0
system.sysLocation.0 = "Atlanta, GA"
```

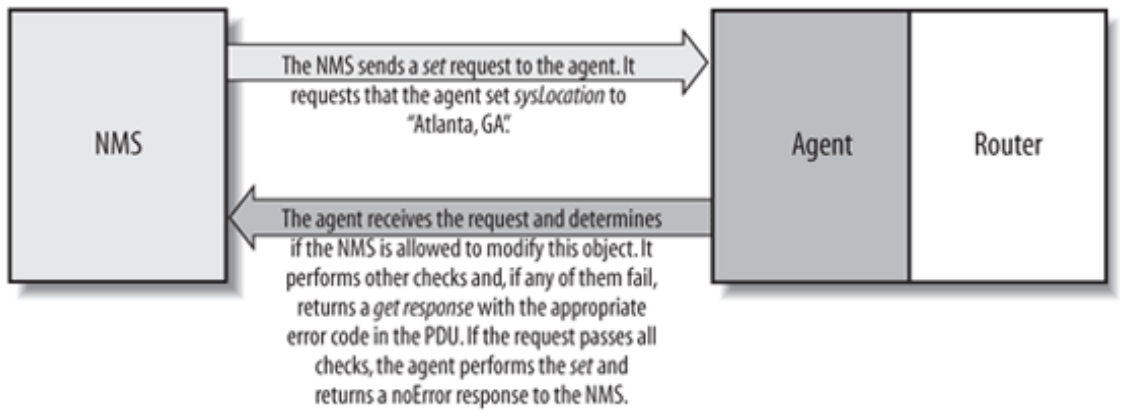

Figure 2-8. set request sequence

The first command is the familiar get command, which displays the current value of *sysLocation*. In one of the previous examples, we saw that it was undefined; this is still the case. The second command is snmpset. For this command, we supply the hostname, the read-write community string (*private*), and the variable we want to set (*system.sysLocation.0*), together with its new value (`s "Atlanta, GA"`). The `s` tells snmpset that we want to set the value of *sysLocation* to a string, and `"Atlanta, GA"` is the new value itself. How do we know that *sysLocation* requires a string value? The definition of *sysLocation* in RFC 1213 looks like this:

```
sysLocation OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The physical location of this node (e.g., 'telephone closet,
         3rd floor')."
    ::= { system 6 }
```

The `SYNTAX` for *sysLocation* is `DisplayString (SIZE (0..255))`, which means that it's a string with a maximum length of 255 characters. The snmpset command succeeds and reports the new value of *sysLocation*. But just to confirm, we run a final snmpget, which tells us that the set actually took effect. It is possible to set more than one object at a time, but if any of the sets fail, they all fail (i.e., no values are changed). This behavior is intended.

It's time for more tethereal output. With the following set command:

```
$ snmpset -v 1 -c private 127.0.0.1 sysContact.0s \ snmp@vagrant.org
```

we get the following output:

```
Frame 1 (89 bytes on wire, 89 bytes captured)
    Arrival Time: Sep 20, 2004 14:25:01.895097000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 89 bytes
    Capture Length: 89 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 75
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7ca0 (correct)
    Source: 127.0.0.1 (127.0.0.1)
```

```
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: 34102 (34102), Dst Port: snmp (161)
    Source port: 34102 (34102)
    Destination port: snmp (161)
    Length: 55
    Checksum: 0xfe4a (incorrect, should be 0xc029)
Simple Network Management Protocol
    Version: 1 (0)
    Community: private
    PDU type: SET (3)
    Request Id: 0x1df8e7e6
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.5.0 (SNMPv2-MIB::sysName.0)
    Value: STRING: box

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 4b 00 00 40 00 00 11 7c a0 7f 00 00 01 7f 00   .K..@...|.......
0020  00 01 85 36 00 a1 00 37 fe 4a 30 2d 02 01 00 04   ...6...7.J0-....
0030  07 70 72 69 76 61 74 65 a3 1f 02 04 1d f8 e7 e6   .private........
0040  02 01 00 02 01 00 30 11 30 0f 06 08 2b 06 01 02   ......0.0...+...
0050  01 01 05 00 04 03 62 6f 78                        ......box

Frame 2 (89 bytes on wire, 89 bytes captured)
    Arrival Time: Sep 20, 2004 14:25:01.902787000
    Time delta from previous packet: 0.007690000 seconds
    Time since reference or first frame: 0.007690000 seconds
    Frame Number: 2
    Packet Length: 89 bytes
    Capture Length: 89 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 75
    Identification: 0x0004 (4)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7c9c (correct)
    Source: 127.0.0.1 (127.0.0.1)
```

```
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34102 (34102)
    Source port: snmp (161)
    Destination port: 34102 (34102)
    Length: 55
    Checksum: 0xfe4a (incorrect, should be 0xc129)
Simple Network Management Protocol
    Version: 1 (0)
    Community: private
    PDU type: RESPONSE (2)
    Request Id: 0x1df8e7e6
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.5.0 (SNMPv2-MIB::sysName.0)
    Value: STRING: box

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 4b 00 04 40 00 00 11 7c 9c 7f 00 00 01 7f 00   .K..@...|.......
0020  00 01 00 a1 85 36 00 37 fe 4a 30 2d 02 01 00 04   .....6.7.J0-....
0030  07 70 72 69 76 61 74 65 a2 1f 02 04 1d f8 e7 e6   .private........
0040  02 01 00 02 01 00 30 11 30 0f 06 08 2b 06 01 02   ......0.0...+...
0050  01 01 05 00 04 03 62 6f 78                        ......box
```

The SNMPv2 set traces are as follows:

```
Frame 1 (89 bytes on wire, 89 bytes captured)
    Arrival Time: Sep 20, 2004 14:25:12.926493000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 89 bytes
    Capture Length: 89 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 75
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
```

```
    Protocol: UDP (0x11)
    Header checksum: 0x7ca0 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: 34102 (34102), Dst Port: snmp (161)
    Source port: 34102 (34102)
    Destination port: snmp (161)
    Length: 55
    Checksum: 0xfe4a (incorrect, should be 0x726b)
Simple Network Management Protocol
    Version: 2C (1)
    Community: private
    PDU type: SET (3)
    Request Id: 0x34df1dbe
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.5.0 (SNMPv2-MIB::sysName.0)
    Value: STRING: box

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 4b 00 00 40 00 00 11 7c a0 7f 00 00 01 7f 00   .K..@...|.......
0020  00 01 85 36 00 a1 00 37 fe 4a 30 2d 02 01 01 04   ...6...7.J0-....
0030  07 70 72 69 76 61 74 65 a3 1f 02 04 34 df 1d be   .private....4...
0040  02 01 00 02 01 00 30 11 30 0f 06 08 2b 06 01 02   ......0.0...+...
0050  01 01 05 00 04 03 62 6f 78                        ......box

Frame 2 (89 bytes on wire, 89 bytes captured)
    Arrival Time: Sep 20, 2004 14:25:12.989438000
    Time delta from previous packet: 0.062945000 seconds
    Time since reference or first frame: 0.062945000 seconds
    Frame Number: 2
    Packet Length: 89 bytes
    Capture Length: 89 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 75
    Identification: 0x0005 (5)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
```

```
    Protocol: UDP (0x11)
    Header checksum: 0x7c9b (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 34102 (34102)
    Source port: snmp (161)
    Destination port: 34102 (34102)
    Length: 55
    Checksum: 0xfe4a (incorrect, should be 0x736b)
Simple Network Management Protocol
    Version: 2C (1)
    Community: private
    PDU type: RESPONSE (2)
    Request Id: 0x34df1dbe
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.5.0 (SNMPv2-MIB::sysName.0)
    Value: STRING: box


0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 4b 00 05 40 00 00 11 7c 9b 7f 00 00 01 7f 00   .K..@...|.......
0020  00 01 00 a1 85 36 00 37 fe 4a 30 2d 02 01 01 04   .....6.7.J0-....
0030  07 70 72 69 76 61 74 65 a2 1f 02 04 34 df 1d be   .private....4...
0040  02 01 00 02 01 00 30 11 30 0f 06 08 2b 06 01 02   ......0.0...+...
0050  01 01 05 00 04 03 62 6f 78                        ......box
```

get, getnext, getbulk, and set Error Responses

Error responses help you determine whether your get or set request was processed correctly by the agent. The get, getnext, getbulk, and set operations can return the error responses shown in Table 2-6. The error status for each error is shown in parentheses.

Table 2-6. SNMPv1 error messages

SNMPv1 error message	Description
noError(0)	There was no problem performing the request.
tooBig(1)	The response to your request was too big to fit into one response.
noSuchName(2)	An agent was asked to get or set an OID that it can't find; i.e., the OID doesn't exist.
badValue(3)	A read-write or write-only object was set to an inconsistent value.
readOnly(4)	This error is generally not used. The `noSuchName` error is equivalent to this one.
genErr(5)	This is a catchall error. If an error occurs for which none of the previous messages is appropriate, a `genErr` is issued.

The SNMPv1 error messages are not very robust. In an attempt to fix this problem, SNMPv2 defines additional error responses that are valid for get, set, getnext, and getbulk operations, provided that both the agent and the NMS support SNMPv2. These responses are listed in Table 2-7.

Table 2-7. SNMPv2 error messages

SNMPv2 error message	Description
noAccess(6)	A set to an inaccessible variable was attempted. This typically occurs when the variable has an `ACCESS` type of `not-accessible`.
wrongType(7)	An object was set to a type that is different from its definition. This error will occur if you try to set an object that is of type `INTEGER` to a string, for example.
wrongLength(8)	An object's value was set to something other than what it calls for. For instance, a string can be defined to have a maximum character size. This error occurs if you try to set a string object to a value that exceeds its maximum length.
wrongEncoding(9)	A set operation was attempted using the wrong encoding for the object being set.

wrongValue(10)	A variable was set to a value it doesn't understand. This can occur when a read-write is defined as an enumeration, and you try to set it to a value that is not one of the enumerated types.
noCreation(11)	You tried to set a nonexistent variable or create a variable that doesn't exist in the MIB.
inconsistentValue	A MIB variable is in an inconsistent state and is not accepting any set requests.
resourceUnavailable(13)	No system resources are available to perform a set.
commitFailed(14)	This error is a catchall for set failures.
undoFailed(15)	A set failed and the agent was unable to roll back all the previous sets up until the point of failure.
authorizationError(16)	An SNMP command could not be authenticated; in other words, someone has supplied an incorrect community string.
notWritable(17)	A variable will not accept a set, even though it is supposed to.
inconsistentName(18)	You attempted to set a variable, but that attempt failed because the variable was in some kind of inconsistent state.

SNMP Traps

A trap is a way for an agent to tell the NMS that something bad has happened. In "Managers and Agents" in Chapter 1, we explored the notion of traps at a general level; now we'll look at them in a bit more detail. Figure 2-9 shows the trap-generation sequence .

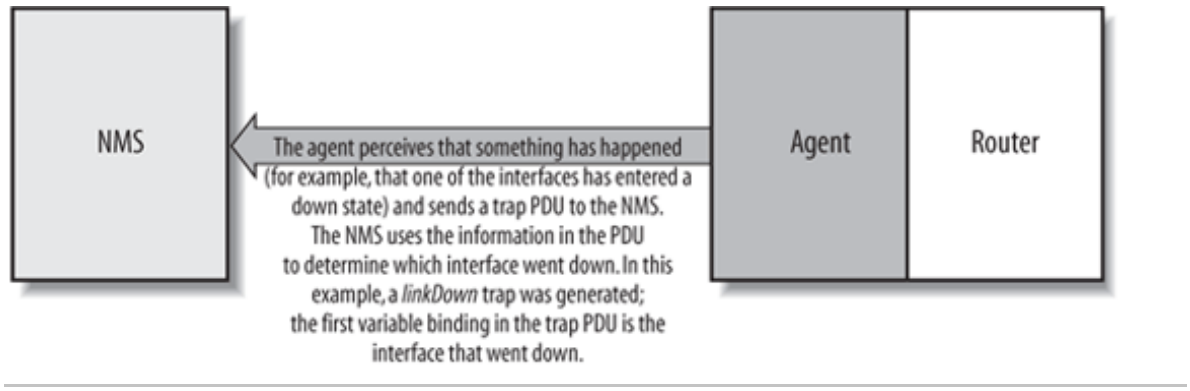

Figure 2-9. Trap-generation sequence

The trap originates from the agent and is sent to the trap destination, as configured within the agent itself. The trap destination is typically the IP address of the NMS. No acknowledgment is sent from the NMS to the agent, so the agent has no way of knowing if the trap makes it to the NMS. Since SNMP uses UDP, and since traps are designed to report problems with your network, traps are especially prone to getting lost and not making it to their destinations. However, the fact that traps can get lost doesn't make them any less useful; in a well-planned environment, they are an integral part of network management. It's better for your equipment to try to tell you that something is wrong, even if the message may never reach you, than simply to give up and let you guess what happened. Here are a few situations that a trap might report:

- A network interface on the device (where the agent is running) has gone down.
- A network interface on the device (where the agent is running) has come back up.
- An incoming call to a modem rack was unable to establish a connection to a modem.
- The fan on a switch or router has failed.

When an NMS receives a trap, it needs to know how to interpret it; that is, it needs to know what the trap means and how to interpret the information it carries. A trap is first identified by its generic trap number. There are seven generic trap numbers (0-6), shown in Table 2-8. Generic trap 6 is a special catchall category for "enterprise-specific " traps, which are traps defined by vendors or users that fall outside of the six generic trap categories. Enterprise-specific traps are further identified by an enterprise ID (i.e., an object ID somewhere in the *enterprises* branch of the MIB tree, *iso.org.dod.internet.private.enterprises*) and a specific trap number chosen by the enterprise that defined the trap. Thus, the object ID of an enterprise-specific trap is *enterprise-id.specific-trap-number*. For example, when Cisco defines special traps for its private MIBs, it places them all in its enterprise-specific MIB tree (*iso.org.dod.internet.private.enterprises.cisco*). As we'll see in Chapter 9, you are free to define your own enterprise-specific traps ; the only requirement is that you register your own enterprise number with IANA.

A trap is usually packed with information. As you'd expect, this information is in the form of MIB objects and their values; as mentioned earlier, these object-value pairs are known as variable bindings . For the generic traps 0 through 5, knowledge of what the trap contains is generally built into the NMS software or trap receiver. The variable bindings contained by an enterprise-specific trap are determined by whomever defined the trap. For example, if a modem in a modem rack fails, the rack's agent may send a trap to the NMS informing it of the failure. The trap will most likely be an enterprise-specific trap defined by the rack's manufacturer; the trap's contents are up to the manufacturer, but it will probably contain enough information to let you determine exactly what failed (for example, the position of the modem card in the rack and the channel on the modem card).

Table 2-8. Generic traps

Generic trap name and number	Definition
coldStart (0)	Indicates that the agent has rebooted. All management variables will be reset; specifically, `Counters` and `Gauges` will be reset to zero (0). One nice thing about the `coldStart` trap is that it can be used to determine when new hardware is added to the network. When a device is powered on, it sends this trap to its trap destination. If the trap destination is set correctly (i.e., to the IP address of your NMS), the NMS can receive the trap and determine whether it needs to manage the device.
warmStart (1)	Indicates that the agent has reinitialized itself. None of the management variables will be reset.
linkDown (2)	Sent when an interface on a device goes down. The first variable binding identifies the index in the *interfaces* table for the interface that went down.
linkUp (3)	Sent when an interface on a device comes back up. The first variable binding identifies which interface came back up.
authenticationFailure (4)	Indicates that someone has tried to query your agent with an incorrect community string; useful in determining if someone is trying to gain unauthorized access to one of your devices.
egpNeighborLoss (5)	Indicates that an EGP neighbor has gone down.
enterpriseSpecific (6)	Indicates that the trap is enterprise-specific. SNMP vendors and users define their own traps under the *private-enterprise* branch of the SMI *object* tree. To process this trap properly, the NMS has to decode the specific trap number that is part of the SNMP message.

In Chapter 1, we mentioned that RFC 1697 is the RDBMS MIB. One of the traps defined by this MIB is *rdbmsOutOfSpace*:

```
rdbmsOutOfSpace TRAP-TYPE
    ENTERPRISE  rdbmsTraps
    VARIABLES   { rdbmsSrvInfoDiskOutOfSpaces }
    DESCRIPTION
        "An rdbmsOutOfSpace trap signifies that one of the database
         servers managed by this agent has been unable to allocate
         space for one of the databases managed by this agent. Care
         should be taken to avoid flooding the network with these traps."
    ::= 2
```

The enterprise is *rdbmsTraps* and the specific trap number is 2. This trap has one variable binding, *rdbmsSrvInfoDiskOutOfSpaces*. If we look elsewhere in the MIB, we will find that this variable is a scalar object. Its definition is:

```
rdbmsSrvInfoDiskOutOfSpaces OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The total number of times the server has been unable to obtain
         disk space that it wanted, since server startup. This would be
         inspected by an agent on receipt of an rdbmsOutOfSpace trap."
    ::= { rdbmsSrvInfoEntry  9 }
```

The `DESCRIPTION` for this object indicates why the note about taking care to avoid flooding the network (in the `DESCRIPTION` text for the `TRAP-TYPE`) is so important. Every time the RDBMS is unable to allocate space for the database, the agent will send a trap. A busy (and full) database could end up sending this trap thousands of times a day.

Some commercial RDBMS vendors, such as Oracle, provide an SNMP agent with their database engines. Agents such as these typically have functionality above and beyond that found in the RDBMS MIB.

Now let's look at an Ethereal trace of an SNMPv1 trap. Given the following command:[*]

```
$ snmptrap -v 1 -c public .1.3.6.1.4.1.2789.2005 127.0.0.1 6\ 2476317 '' .1.3.6.1.4.
1.2789.2005.1 s "WWW Server Has Been\ Restarted"
```

Ethereal gives us the following trace:

```
Frame 1 (135 bytes on wire, 135 bytes captured)
    Arrival Time: Sep 20, 2004 14:38:40.191174000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 135 bytes
    Capture Length: 135 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 121
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7c72 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
```

```
User Datagram Protocol, Src Port: 34108 (34108), Dst Port: snmptrap (162)
    Source port: 34108 (34108)
    Destination port: snmptrap (162)
    Length: 101
    Checksum: 0xfe78 (incorrect, should be 0xf82d)
Simple Network Management Protocol
    Version: 1 (0)
    Community: public
    PDU type: TRAP-V1 (4)
    Enterprise: 1.3.6.1.4.1.2789.2005 (SNMPv2-SMI::enterprises.2789.2005)
    Agent address: 127.0.0.1 (127.0.0.1)
    Trap type: ENTERPRISE SPECIFIC (6)
    Specific trap type: 2476317
    Timestamp: 181730327
    Object identifier 1: 1.3.6.1.4.1.2789.2005.1 (SNMPv2-SMI::enterprises.2789.2005.1)
    Value: STRING: "WWW Server Has Been Restarted"

0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010  00 79 00 00 40 00 00 11 7c 72 7f 00 00 01 7f 00   .y..@...|r......
0020  00 01 85 3c 00 a2 00 65 fe 78 30 5b 02 01 00 04   ...<...e.x0[....
0030  06 70 75 62 6c 69 63 a4 4e 06 09 2b 06 01 04 01   .public.N..+....
0040  95 65 8f 55 40 04 7f 00 00 01 02 01 06 02 03 25   .e.U@.........%
0050  c9 1d 43 04 0a d4 fc 17 30 2d 30 2b 06 0a 2b 06   ..C.....0-0+..+.
0060  01 04 01 95 65 8f 55 01 04 1d 57 57 57 20 53 65   ....e.U...WWW Se
0070  72 76 65 72 20 48 61 73 20 42 65 65 6e 20 52 65   rver Has Been Re
0080  73 74 61 72 74 65 64                              started
```

We have only one frame since the agent initiated the trap. An SNMPv1 trap is sent from the agent and is not acknowledged by the receiver in any way, so the agent sends it and forgets about it. This is why we see just the single trace.

SNMP Notification

In an effort to standardize the PDU format of SNMPv1 traps (recall that SNMPv1 traps have a different PDU format from get and set), SNMPv2 defines a `NOTIFICATION-TYPE`. The PDU format for `NOTIFICATION-TYPE` is identical to that for get and set. RFC 2863 redefines the *linkDown* generic notification type like so:

```
linkDown NOTIFICATION-TYPE
    OBJECTS { ifIndex, ifAdminStatus, ifOperStatus }
    STATUS  current
    DESCRIPTION
        "A linkDown trap signifies that the SNMPv2 entity, acting in an
         agent role, has detected that the ifOperStatus object for one
         of its communication links left the down state and transitioned
         into some other state (but not into the notPresent state). This
         other state is indicated by the included value of ifOperStatus."
    ::= { snmpTraps 3 }
```

The list of bindings is called `OBJECTS` rather than `VARIABLES`, but little else has changed. The first object is the specific interface (*ifIndex*) that transitioned from the *linkDown* condition to some other condition. The OID for this trap is *1.3.6.1.6.3.1.1.5.3*, or *iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snmpMIBObjects.snmpTraps.linkDown*.

Let's look at how to create an SNMP notification:

```
$ snmptrap -v2c -c public 127.0.0.1 '' .1.3.6.1.6.3.1.1.5.3 ifIndex i 2 ifAdminStatus
i 1 ifOperStatus i 1
```

The datagram trace is as follows:

```
Frame 1 (162 bytes on wire, 162 bytes captured)
    Arrival Time: Sep 20, 2004 14:38:53.846768000
    Time delta from previous packet: 0.000000000 seconds
    Time since reference or first frame: 0.000000000 seconds
    Frame Number: 1
    Packet Length: 162 bytes
    Capture Length: 162 bytes
Ethernet II, Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00
    Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 127.0.0.1 (127.0.0.1), Dst Addr: 127.0.0.1 (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..0. = ECN-Capable Transport (ECT): 0
        .... ...0 = ECN-CE: 0
    Total Length: 148
    Identification: 0x0000 (0)
    Flags: 0x04
        .1.. = Don't fragment: Set
        ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 0
    Protocol: UDP (0x11)
    Header checksum: 0x7c57 (correct)
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
User Datagram Protocol, Src Port: 34108 (34108), Dst Port: snmptrap (162)
    Source port: 34108 (34108)
    Destination port: snmptrap (162)
    Length: 128
    Checksum: 0xfe93 (incorrect, should be 0x76ba)
Simple Network Management Protocol
    Version: 2C (1)
    Community: public
    PDU type: TRAP-V2 (7)
    Request Id: 0x6737908a
    Error Status: NO ERROR (0)
    Error Index: 0
    Object identifier 1: 1.3.6.1.2.1.1.3.0 (SNMPv2-MIB::sysUpTime.0)
    Value: Timeticks: (181731693) 21 days, 0:48:36.93
    Object identifier 2: 1.3.6.1.6.3.1.1.4.1.0 (SNMPv2-MIB::snmpTrapOID.0)
    Value: OID: IF-MIB::linkDown
```

```
    Object identifier 3: 1.3.6.1.2.1.2.2.1.1 (IF-MIB::ifIndex)

    Value: INTEGER: 2

    Object identifier 4: 1.3.6.1.2.1.2.2.1.7 (IF-MIB::ifAdminStatus)

    Value: INTEGER: up(1)

    Object identifier 5: 1.3.6.1.2.1.2.2.1.8 (IF-MIB::ifOperStatus)

    Value: INTEGER: up(1)


0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.

0010  00 94 00 00 40 00 00 11 7c 57 7f 00 00 01 7f 00   ....@...|W......

0020  00 01 85 3c 00 a2 00 80 fe 93 30 76 02 01 01 04   ...<......0v....

0030  06 70 75 62 6c 69 63 a7 69 02 04 67 37 90 8a 02   .public.i..g7...

0040  01 00 02 01 00 30 5b 30 10 06 08 2b 06 01 02 01   .....0[0...+....

0050  01 03 00 43 04 0a d5 01 6d 30 17 06 0a 2b 06 01   ...C....m0...+..

0060  06 03 01 01 04 01 00 06 09 2b 06 01 06 03 01 01   .........+......

0070  05 03 30 0e 06 09 2b 06 01 02 01 02 02 01 01 02   ..0...+.........

0080  01 02 30 0e 06 09 2b 06 01 02 01 02 02 01 07 02   ..0...+.........

0090  01 01 30 0e 06 09 2b 06 01 02 01 02 02 01 08 02   ..0...+.........

00a0  01 01                                            ..
```

SNMP inform

SNMPv2 provides an inform mechanism , which allows for acknowledged sending of traps. This operation can be useful when the need arises for more than one NMS in the network. When an inform is sent, the receiver sends a response to the sender acknowledging receipt of the event. This behavior is similar to that of the get and set requests. Note that an SNMP inform can be used to send SNMPv2 traps to an NMS. If you use an inform for this purpose, the agent will be notified when the NMS receives the trap.

SNMP report

The report operation was defined in the draft version of SNMPv2 but was never implemented. It is now part of the SNMPv3 standard and is intended to allow SNMP engines to communicate with each other (mainly to report problems with processing SNMP messages).