CHAPTER 3

SNMP Commands

This chapter describes the SNMP commands available from the command line interface (CLI) on a LightStream 2020 (LS2020) enterprise ATM switch. This chapter is intended for SNMP experts who use the commands to monitor and manipulate MIB objects. If you are not familiar with SNMP, it is recommended that you do not use these commands.

In some cases, SNMP commands may be long and repetitive. You can store sequences of SNMP commands in CLI script files and execute them using the **source** command described in the *LightStream 2020 CLI Reference Manual*.

SNMP Commands Using the CLI

The CLI provides the following SNMP commands:

- **browse—**Lets you walk through the MIB and obtain any value in the MIB tree.
- **getsnmp—**Corresponds to the standard SNMP command **getrequest.**
- **getnextsnmp—**Corresponds to the standard SNMP command **getnextrequest.**
- **setsnmp—**Corresponds to the standard SNMP command **setrequest.**
- **walksnmp—**Lets you display MIB objects.

MIB Overview

The next sections include:

- A description of MIB object names and instance identifiers
- Procedures on how to monitor values of a MIB object, set a MIB object, and walk a MIB subtree

MIB Object Names

MIB object names are used to identify the MIB objects in the SNMP commands described in this chapter. MIB object names come from the standard or private MIBs that the LS2020 switch supports. For a detailed discussion of the MIB, see the *LightStream 2020 CLI Reference Manual*.

MIB Instance Identifier

In addition to a MIB object name, you must use a MIB *instance identifier* to manipulate a particular MIB object. SNMP calculates a suffix based on the hierarchy of a MIB object and adds the suffix to the MIB object name to form its instance identifier. This use of instance identifiers allows all MIB object names to be unique.

For example, the MIB object *sysDescr* has only a single value. It is identified by its MIB object name (sysDescr) followed by its instance identifier (0), resulting in *sysDescr.0*.

A MIB object with multiple instances has a unique identifier for each instance. In the following examples, the number (1, 16, 24, or 43) following the MIB object *pidName* identifies the specific instance of *pidName*:

- pidName.1
- pidName.16
- pidName.24
- pidName.43

For a further discussion of instance identifiers, see *The Simple Book: An Introduction to Management of TCP/IP-based Internets* by Marshall T. Rose, 1991, Prentice Hall, Inc. (ISBN 0-13-812611-9).

Identifying MIB Objects

If you are unsure of a MIB object name, use the **walksnmp** command with the specific object's MIB tree or subtree name as the command argument. (See "Walking a MIB Subtree" later in this chapter.) This displays all MIB objects below the specified point in the tree. Whenever the attribute MIB name or MIB address is required by a syntax statement, you must enter both the MIB object name and its instance identifier.

Monitoring the Value of a MIB Object

The procedures in this section tell you how to view the value of a particular MIB object using one of two different commands. Procedure 1 displays the value of the MIB object you specify. Procedure 2 displays the value of the MIB object following the MIB object you specify. The **getnextsnmp** command is useful if you don't know the exact structure of the MIB.

Procedure 1: Displaying the Value of a Specified MIB Object

At the cli> prompt, enter:

```
cli> getsnmp <MIB name or address>
```

where

<MIB name or address> = Name or address of the MIB object you want information on. You may enter multiple names or addresses.

Some examples of the getsnmp command are

```
getsnmp sysDescr.0
getsnmp sysDescr.0 sysObjectID.0 sysUpTime.0
getsnmp 1.3.6.1.2.1.1.1.0
```

Procedure 2: Displaying the Value of the MIB Object After a Specified MIB Object

At the cli> prompt, enter:

```
cli> getnextsnmp <MIB name or address>
```

where

<MIB name or address> = Name or address of the MIB object just before the MIB object you want to get information on. You may enter multiple names or addresses.

When you enter **getsnmp sysDescr.0**, the following information appears:

```
cli> getsnmp sysDescr.0
Name: sysDescr.0      Value: LightStream 2020 ATM Switch
cli>
```

When you enter **getnextsnmp sysDescr.0**, the following information appears:

```
cli> getnextsnmp sysDescr.0
Name: sysObjectID.0    Value: LightStream 2020 ATM Switch
cli>
```

Note When you enter the **getnextsnmp** command, the value of the MIB object that comes next in the subtree displays.

Setting a MIB Object

These steps tell you how to set the value of a MIB object using the **setsnmp** command. If you change a MIB object using this command, the change is not saved permanently to the hard disk. Instead, the changes are made to run-time memory. If you reboot the system, the changes made using this procedure are lost.At the cli> prompt, enter:

```
cli> protected
```

Step 3 Enter the protected mode password when you see the following prompt:

```
Enter password:
```

Step 4 Set the SNMP community to a read/write community by entering the following at the *cli> prompt:

```
*cli> set snmp community <community name>
```

where
<community name> = The name of the SNMP community with read/write privileges that you want to access. (A switch can have several SNMP community names with read/write privileges.) The SNMP community must be set to a read/write community before you can set the MIB object value.

Step 5 At the *cli> prompt, enter:

```
*cli> setsnmp <MIB name or address> <value>
```

where

<MIB name or address> = Name or address of the MIB object you want to change. You may enter multiple names or addresses and their corresponding values.

<value> = The value of the MIB object you want to set.

For example, you can enter

```
setsnmp chassisId 4143
```

Whenever you issue a **setsnmp** command from CLI, it is automatically followed by a **getsnmp** command. See the information displayed by the **getsnmp** command to verify that the change you made is correct. After entering the command shown above, the following would be displayed:

```
Name: chassisId.0     Value:  4143
```

Walking a MIB Subtree

To display a portion of the MIB subtree using the **walksnmp** command, you can enter the name of the subtree you want to "walk" through. This command displays all MIB objects (and their values) in that subtree. You can also use **walksnmp** to get a list of all the instances of a particular MIB object in the switch.

At the cli> prompt, enter:

```
   cli> walksnmp <MIB name or address>
```

where

<MIB name or address> = The name or address of a MIB subtree or MIB object. (You must put the name in quotes if it contains non-alphanumeric characters)

For example, to walk the System subtree, enter:

```
   cli> walksnmp system
```

or

```
   cli> walk mib2
```

or

```
   cli> walksnmp 1.3.6.1.2.1.1
```

To list the names of all the processes running in the LS2020 switch, enter:

```
   cli> walksnmp pidName
```

When you enter **walksnmp system,** the following information is displayed.

```
   cli> walksnmp system
   Name:  sysDescr.0      Value: ATM Data Switch
   Name:  sysObjectID.0   Value: SWITCH_SNMP_Agent
   Name:  sysUpTime.0     Value: 39501284
   Name:  sysContact.0    Value: Tom Matthews
   Name:  sysName.0       Value: emtb7
   Name:  sysLocation.0   Value: Cambridge
   Name:  sysServices.0   Value: 78
   cli>
```

When you enter **walksnmp pidName,** the following information is displayed:

```
Name:  pidName.8       Value: ndd
Name:  pidName.9       Value: watchdog
Name:  pidName.10      Value: mma
Name:  pidName.16      Value: kernlog
Name:  pidName.17      Value: cac
Name:  pidName.18      Value: collector
Name:  pidName.19      Value: mptmm
Name:  pidName.20      Value: npcc
Name:  pidName.21      Value: gidd
Name:  pidName.23      Value: lcc
Name:  pidName:24      Value: lcc
Name:  pidName.25      Value: lcc
Name:  pidName.33      Value: lcc
cli>
```