Utilizing SNMP Capabilities of EMC Disk Library

A Detailed Review

Abstract

EMC® Disk Library (EDL) provides Simple Network Management Protocol (SNMP) as part of its monitoring solution. This white paper describes in some detail how customers can take advantage of SNMP for EDL.

January 2010
# Table of Contents

**Executive summary** ...................................................................................................................... 4

**Introduction** ........................................................................................................................................ 4

**Overview of SNMP** ............................................................................................................................ 5

**How EMC Disk Library can use SNMP** ............................................................................................ 10

**Conclusion** .......................................................................................................................................... 43

---

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review 3
Executive summary

EMC® Disk Library (EDL) provides Simple Network Management Protocol (SNMP) as part of its monitoring solution. This white paper describes in some detail how customers can take advantage of SNMP for EDL.

Introduction

SNMP is an industry-defined protocol for managing network devices. Most devices with an IP address have an agent running on them that will respond to SNMP commands. Many devices allow write (known as set) access so that they may be configured remotely. Having the capability to remotely monitor and (sometimes) configure network devices allows enterprises to manage very large networks with relatively few human resources. As a result, virtually all EMC customers use SNMP as part of their IT management tools.

Network administrators in an enterprise must maintain control over networks with as many as 1 million nodes. In almost all such networks, SNMP is the vehicle used to monitor (and, in many cases, manage) these devices. Figure 1 illustrates the view of a network administrator who must monitor the status of thousands of devices. Without SNMP, his task would be much more difficult. EDL offers several ways to work within this framework.

Figure 1. View of a network administrator monitoring status of thousands of devices

Figure 1 shows the view from a typical network monitoring station. Using SNMP and enterprise-level monitoring software as tools, personnel in a central location can maintain control of their network.

Audience

This white paper provides customers and system engineers with a functional description of how SNMP technology can benefit their clients and how SNMP technology can be used in an EDL environment. It assumes the reader has a basic understanding of SNMP concepts.
Overview of SNMP

This section provides background material for readers who may not be familiar with SNMP. If you are already familiar with SNMP, proceed to “Using SNMP management software” on page 15.

Definition of terms

Some of the terms used when describing SNMP are explained below for reference.

SNMP: Simple Network Management Protocol

SNMP is the protocol governing network management and the monitoring of network devices and their functions.

SNMP is a client/server—or more specifically a manager/agent—protocol. The agent (server) runs on the device being managed. The agent monitors the status of the device and reports that status to the manager.

The manager (client) runs on a separate workstation in the network. This workstation collects information from all devices being managed and consolidates it into an interface presented to the administrator. By concentrating the bulk of the processing at the management station, the agent tends to be small and easy to implement on the managed device.

SNMP is a request/response protocol with its well-known port, defined as UDP (User Datagram Protocol) port 161. SNMP uses UDP as its transport protocol because it has no need for the overhead of TCP. Reliability is not required because each request generates a response. If an SNMP application does not get a response, it simply reissues the request.

Table 1 shows the six request/response messages defined in SNMP. These are Protocol Data Units (PDUs).

Table 1. SNMP Protocol Data Units

<table>
<thead>
<tr>
<th>PDU</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetRequest</td>
<td>Manager requests an update.</td>
</tr>
<tr>
<td>GetNextRequest</td>
<td>Manager requests the next entry in a table.</td>
</tr>
<tr>
<td>GetResponse</td>
<td>Agent answers a manager request.</td>
</tr>
<tr>
<td>SetRequest</td>
<td>Manager modifies data on the managed device.</td>
</tr>
<tr>
<td>Trap</td>
<td>Agent alerts manager of an unusual event.</td>
</tr>
<tr>
<td>Walk</td>
<td>Manager issues sequential GetNext requests for an entire branch of a MIB tree.</td>
</tr>
</tbody>
</table>

These message types allow the manager to request information and—where permitted by the agent—modify that information. Note that the EMC Disk Library does not allow set commands. Therefore, SNMP is used only to monitor—not modify—EMC Disk Library.

However, fabric and network switches may be modified via SNMP. Visit the vendor website to download the MIBs that will enable this level of remote management.

Polling occurs when the management workstation issues a GetRequest on an agent at a specified periodic interval. This allows the management station to maintain a current view of network status. A shortcoming of polling is that it does not allow for real-time updates: Events are not known to the administrator until the next polling interval. To address this, SNMP uses traps. A trap is an interrupt signaled by a predefined event. When a predefined event occurs, the SNMP Agent immediately sends information to the management workstation. Traps allow the agent to inform the manager of unusual events while allowing the manager to maintain control via polling. SNMP traps are on UDP port 162. The manager sends polls on port 161 and listens for traps on port 162.
The SNMP-enabled agent on these devices is written to manage particular objects of that device. Each of these objects has a globally unique name called an Object Identifier (OID). OIDs are described in detail in the “OID: Object Identifiers” section on page 7. When the manager queries an agent, it asks for information about that agent’s OIDs. The relevant OIDs are defined for the manager in a Management Information Base (MIB), which is described next.

**MIB: Management Information Base**

A MIB is a formal description of a set of network objects that can be managed using SNMP. MIB format is defined as part of the SNMP. MIB-I refers to the initial MIB definition; MIB-II (RFC1213) refers to the current definition.

Simply put, a MIB is a text file. It uses a standard syntax to describe objects on which it is looking for information. Distributed agents will be queried by a management station that has compiled the MIB into it. The agent may or may not reply to these queries; responses generated by queries depend on how the agent was written. Think of the MIB as a “database of questions” that can be posed to distributed agents via SNMP.

It provides information such as the number of packets transmitted into and out of an interface, the number of errors that occurred sending and receiving those packets, and other useful information for spotting usage trends and potential trouble spots.

MIB1 and MIB2 are defined, standard MIBs. MIB2 is a superset of MIB1 and is the standard MIB for monitoring TCP/IP. (RFC1213 is the MIB2 MIB.) Every agent supports MIB1 or MIB2.

Figure 2 shows a graphical representation of a standard MIB, the Fibre Alliance MIB, which is used by the EMC CLARiiON® storage system. (It is not used by the EDL and is only shown here for topical interest.) The tree metaphor is used to describe the hierarchical nature of objects and their assigned names: The FA MIB represents a branch within the overall MIB tree, and each leaf defines an OID.

![Figure 2. Graphical representation of the Fibre Alliance MIB](image)
Some SNMP-enabled devices provide a private MIB in addition to MIB2. Private MIBs add monitoring capability by providing system-specific information. The Fibre Alliance MIB is an example of an additional MIB that provides information beyond that reported within MIB2.

The EMC Disk Library has its own private MIB. Most SNMP management applications include a MIB compiler. Using this compiler, an administrator needs only to get the MIB source, and then the private MIB can be added; the agent can then be queried with it.

**OID: Object Identifiers**

A MIB describes instrumentation that is supported and assigns a unique identifier string to each of these points. SNMP MIBs use a tree structure in which the managed objects are at the leaf level. These leaves are called Object Identifiers (OIDs) and they describe a path to walk from the “root” of a tree to a specific leaf. OIDs are part of the hierarchical name space that is managed by the ISO. They employ a dotted decimal notation to create their unique identifiers.

Figure 3 shows a properties screen for a specific OID. Note that:

- The OID is defined by the MIB RFC1213 (which is the MIB-II MIB).
- The OID has a unique identifier (which is registered and maintain by the ISO).
- A description of the object is included as part of the MIB’s definition.

```
Name:            sysDescr
Type:            OBJECT-TYPE
OID:             1.3.6.1.2.1.1.1.0
Full path:       iso(1).org(3).doc(6).internet(1).mgmt(2).sys(1).sysDescr(1)
Module:          RFC1213-MIB
Parent:          system
Next sibling:    sysObjectID

Description:     A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software. It is mandatory that this only contain printable ASCII characters.
```

Figure 3. Properties screen for OID .1.3.6.1.2.1.1.1.0

The OID is the object that actually has the information. When the manager queries an agent, it is looking for information on one or several OIDs. The syntax of the call looks like this:

```
C:\>snmputil get 128.221.46.22 public .1.3.6.1.2.1.1.1.0
```

In this example, the manager is querying the agent at 128.221.46.22 for contents of the specific object defined by OID .1.3.6.1.2.1.1.1.0 (which is the OID shown in the previous example).
The response received is:

Variable = system.sysDescr.0  
Value    = Linux DullsabiA 2.4.21-ipstor #1 SMP Sat Jul 12 20:16:53 EDT 2003 i686

This indicates that the device is an EMC Disk Library whose EDL server name is DullsabiA. Note that the server name is also available in object sysName, as described later.

The community string

SNMP uses the notion of communities to establish trust between managers and agents. Agents can be defined so that they grant read-only access to queries from certain communities, read/write privileges to requests from other communities, and so on. Community names are essentially role-based passwords.

For example, look at your Windows PC. You want to monitor your PC via SNMP, so you start the SNMP Service. The agent allows you to set privileges according to the community string contained in the SNMP request. If the PC in Figure 4 receives a Get request from an SNMP Manager in the “joseph” community, it will be processed; if the same request comes from the “public” community, it will be denied.

![SNMP Service Properties (Local Computer)](image)

**Figure 4. Implementing role-based SNMP security using the community string**

---

1 The SNMP is not part of the standard Windows installation. To load the SNMP Service, select Control Panel > Add/Remove Programs > Add/Remove Windows Components > Management and Monitoring Tools.
IETF: Internet Engineering Task Force
The IETF is the body that defines standard Internet operating protocols such as TCP/IP. The IETF is supervised by the Internet Society Internet Architecture Board. IETF members are drawn from the Internet Society's individual and organization membership. Standards are expressed in the form of Requests for Comments (RFCs). Their records are found at http://www.ietf.org.

NOC: Network Operations Center
An NOC is a place from which a large network is supervised, monitored, and maintained. Enterprises with large networks typically have an NOC:

- A room containing visualizations of the network or networks that are being monitored
- Workstations at which the detailed status of the network can be seen
- The necessary software to manage the networks

The NOC is the focal point for:

- Network troubleshooting
- Software distribution and updating
- Router and domain name management
- Performance monitoring
- Coordination with affiliated networks

UDP: User Datagram Protocol
UDP is a communications protocol that offers a limited amount of service when messages are exchanged between computers in a network that uses the Internet Protocol (IP). UDP is an alternative to the Transmission Control Protocol (TCP). Like TCP, UDP uses IP to get a data unit (called a datagram) from one computer to another. Unlike TCP, however, UDP does not provide the service of dividing a message into packets and reassembling it at the other end. Specifically, UDP doesn't provide sequencing of the packets in which the data arrives. This means that the application program that uses UDP must be able to make sure that the entire message has arrived and is in the right order.

Network applications that want to save processing time because they have very small data units to exchange (and therefore very little message reassembling to do) may prefer UDP to TCP. SNMP is such an application.

SNMP versions (v1, v2, v2c, and v3)
The different versions of SNMP define new versions of the protocol. Just as Internet Protocol IPv6 will offer enhancements over the current IPv4 (that is, 128-bit addressing compared to 32-bit), each later version of SNMP adds functionality to the protocol. Following is a brief description of the differences.

- **SNMPv1** is the current standard. Its security is based on communities: plain-text strings that allows access to any SNMP-based application that knows the strings to that device’s management information. Communities in v1 are either read-only, read/write, or trap.
- **SNMPv2c** may also be called v2. It uses a more robust Structure of Management Information (SMIv2), which includes additional datatypes as part of its definition.
- **SNMPv3** adds support for strong authentication and private communications between managed entities.

This discussion of SNMP for EMC Disk Library refers to the standards in place with SNMPv1.
MIB-II (or RFC 1213)
All agents implement a particular MIB called MIB-II, although some agents may also implement additional MIBs. The goal of MIB-II is to provide general management information. Refer to the “EMC Disk Library MIB” section on page 23.

NMS and Frameworks
This paper refers to Network Monitoring Stations (NMS) and Framework interchangeably. The NMS refers to the workstation where the management Framework software resides. The most popular Frameworks include HP OpenView, Tivoli NetView, CA Unicenter, and BMC Patrol.

How EMC Disk Library can use SNMP
Two ways to use SNMP in an EMC Disk Library environment are as follows:

- EMC Disk Library can be configured to send traps to an SNMP management station or to a “trap catcher.”
- The EMC Disk Library MIB can be compiled into SNMP management station software or a MIB browser in order to monitor EDL.

No SNMP Manager can modify EMC Disk Library via SNMP. In EMC Disk Library SNMP implementations, there are no set requests and no writing to agents—only monitoring is enabled.

EMC Disk Library community strings
If your EMC Disk Library is running version 3.0 or earlier software, the community string for SNMP reads (gets) is “public” and the default when adding a trap destination (see Figure 5) is also “public” but you may change the community string for EDL traps on a destination by destination basis.

If your EMC Disk Library is running version 3.1 or later software, the community string for SNMP reads (gets) is “DLmonitor” and the default when adding a trap destination is also “DLmonitor”, but you may change the community string for EDL traps on a destination-by-destination basis.

It is not possible to alter the community string for SNMP reads (gets) via the EDL Console.

The EMC Disk Library trap
An EMC Disk Library trap contains basic information about the type of event that has occurred. Any and all EDL events are candidates for SNMP traps to be sent to an SNMP management station. As mentioned earlier but worth repeating: There is no capability to issue set commands in an EMC Disk Library environment.

You can choose the severity level of traps your EDL server will send out. You do this when you configure SNMP by right-clicking the EDL server, selecting Properties, and then clicking the SNMP Maintenance tab. For example, Figure 5 indicates that informational, warning, error, and critical traps will all be sent to two SNMP management stations. Note that the severity level you select causes traps of that level as well as all higher ones to be sent. Selecting Error means that only error and critical traps would be sent; informational and warning messages would be placed in the EDL Console log but would not cause traps.
As described in the OID definition, each OID must have a unique identifier assigned by the standards body, the Internet Assigned Numbers Authority (IANA). This is accomplished by using a private branch within IANA’s managed namespace. EMC Disk Library uses “7368” as its enterprise code, so any EDL-related OIDs will be found under the branch .1.3.6.1.4.1.7368, or


Table 2 shows all possible EDL traps.

**Table 2. Listing of OIDs contained in a EMC Disk Library trap**

<table>
<thead>
<tr>
<th>OID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1.3.6.1.4.1.7368.1</td>
<td>Alarm – not used</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.2</td>
<td>Alarm – not used</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.3</td>
<td>Alarm – not used</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.4</td>
<td>Failover start trap</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.5</td>
<td>Failover recovery trap</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.6</td>
<td>Critical trap</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.7</td>
<td>Error trap</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.8</td>
<td>Warning trap</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.9</td>
<td>Informational trap</td>
</tr>
</tbody>
</table>

An EMC Disk Library trap has only one additional field, the eventlog message, which is described next\(^2\):

Graphical SNMP representations were created using MG-SOFT MIB Browser. Go to [http://www.mg-soft.com](http://www.mg-soft.com) for more information on this product.
OID: The trap’s OID (for example, 1.3.6.1.4.1.7368.9 for an informational message).

Type: STRING.

Value: A string represents the eventlog message in the following format:

\[<Severity>\] \[<EventID>\] \<EventMessage>\n
\(<Severity>\): A single character within brackets represents the severity.
- C: Critical
- E: Error
- W: Warning
- I: Informational

\(<EventID>\): A decimal number represents the eventlog ID. This is the same ID you see in the EDL Console Log.

\(<EventMessage>\): Event log message. This is the same message you see in the EDL Console Log.

Following are examples of the trap definitions from the MIB:

```snmp
criticalLogMessage TRAP-TYPE
  ENTERPRISE ipstor
  DESCRIPTION "The IPStor console will receive all log messages of critical degree if the property is set via IPStor console."
  ::= 6

errorLogMessage TRAP-TYPE
  ENTERPRISE ipstor
  DESCRIPTION "The IPStor console will receive all log messages of error degree if the property is set via IPStor console."
  ::= 7

warningLogMessage TRAP-TYPE
  ENTERPRISE ipstor
  DESCRIPTION "The IPStor console will receive all log messages of warning degree if the property is set via IPStor console."
  ::= 8

informationalLogMessage TRAP-TYPE
  ENTERPRISE ipstor
  DESCRIPTION "The IPStor console will receive all log messages of informational degree if the property is set via IPStor console."
  ::= 9
```

Inside any of these four traps, in addition to the mandatory standard SNMP trap fields, is one additional field: eventlog message. Being the same as fields in any standard traps, this field has three parts: OID, type, and value.
The following is an example of an informational trap resulting from deletion of a virtual tape drive:

Simple Network Management Protocol
Version: 1 (0)
Community: public
PDU type: TRAP-V1 (4)
Enterprise: 1.3.6.1.4.1.7368 (IPSTOR-MIB::ipstor)
Agent address: 10.3.1.156 (10.3.1.156)
Trap type: ENTERPRISE SPECIFIC (6)
Specific trap type: 9
Timestamp: 762632
Object identifier 1: 1.3.6.1.4.1.7368.9 (IPSTOR-MIB::ipstor.9)
Value: STRING: "[I] [40055] Virtual Tape Drive 109 deleted from Virtual Library 102"

Failure (simulated by disconnection) of a power supply in the EDL server (note that this is from EDL 2.0 hardware) provides another example. As shown in Figure 6, the EDL Console displays an error.

Figure 6. EDL Console event log showing hardware failure
A trap will be sent:

August 4, 2005 3:06:31 PM EDT: 10.15.40.74(10.15.40.74):
errorLogMessage ipstor([DullsabiA] [E] [11788] Appliance Hardware Problem: Error[Sensor 55: PS Redundancy (Power Supply)] Status[Redundancy Lost].)

EDL traps may originate from an event from one of three places:

- EDL software
- EDL server hardware
- CLARiiON

In all cases, the event will first be logged in the EDL Console. The ID in the console log is the EventID in the SNMP trap. If the trap is due to EDL software or hardware, the EventID may be found in the (nonexhaustive) list of EDL event codes in the EMC CLARiiON Disk Library Administrator’s Guide

EDL 1.2 software returns the EDL event number as the hex equivalent of the event code.

**EDL software traps**

As mentioned earlier, any message placed in the EDL Console event log is a candidate to become a trap if the level has been set appropriately. At the Informational level, you receive notification whenever someone connects to the EDL server through the console; creates a virtual tape library, drive, or tape; creates a SAN client, and so forth. Warning level indicates you’ve used up 90 percent of the CLARiiON capacity (or whatever percentage you might have configured) or the EDL server cannot reach the CLARiiON SPs over the network. Error level usually indicates some internal software problem that may need corrective action.

**EDL server hardware traps: EDL 2.0 and later hardware**

Sensors on the hardware server are polled regularly. If any sensors exceed the critical thresholds (the section “EDL hardware monitoring: EDL 2.0 and later hardware” on page 26 provides an example of all the hardware sensor information), a critical severity message will be logged and a trap sent if so configured. Loss of power supply or fan redundancy causes an error message to be logged and a trap sent if configured to do so. If the condition remains unchanged, it will not generate a log event or trap unless it continues for two hours. Therefore, you can get repeated traps every two hours in such a case.

You can check on the hardware sensors by using Telnet to connect the EDL server and running the command `sensors` or `sensors -v` to read the information via IPMI.

This information is provided for EMC field service use.

**EDL server hardware traps: EDL 1.2 hardware**

Seventeen sensors on the hardware server are polled regularly. If any sensors exceed the critical thresholds (the section “EDL hardware monitoring: EDL 1.2 hardware” on page 41 provides an example of all the hardware sensor information), a critical severity message will be logged and a trap sent if so configured. Loss of redundancy of power supply or fan redundancy causes an error message to be logged and a trap sent if configured to do so. If the condition remains unchanged, it will not generate a log event or trap unless it continues for two hours. Therefore, you can get repeated traps every two hours in such a case.

You can check on the hardware sensors by using Telnet to connect to the EDL server and running the command `sdt`.

This information is provided for EMC field service use.
**CLARiiON trap information**

If the event is a result of a CLARiiON trap received by the EDL server (the *EMC CLARiiON Disk Library Administrator’s Guide* describes how to set this up), the CLARiiON trap is converted to one of four EDL events, depending on the severity of the trap:

The CLARiiON event ID itself is decoded to produce the severity; the severity was not explicitly provided in the CLARiiON trap until FLARE® release 19. EDL decodes the severity to be backward-compatible with older FLARE code that may be used on earlier EDL releases.

- 11784 is an informational CLARiiON trap.
- 11785 is a warning CLARiiON trap.
- 11786 is an error CLARiiON trap.
- 11787 is a critical CLARiiON trap.

In turn (if so configured), EDL will send a trap for each of these events. This type of EDL trap also includes the CLARiiON event ID, as shown in the following example:

```
```

This is an example of an informational trap, EDL event 11784 decimal, which originated with CLARiiON event 41000000 hex. Note that this particular event may not actually be sent by the CLARiiON array, and therefore will not become a EDL event or EDL trap. Only those CLARiiON event IDs specified in the ConnectEMC template corresponding to the FLARE release can become CLARiiON traps.

**CLARiiON health check**

To provide additional monitoring of the CLARiiON that is part of the EDL appliance, a *health check* routine is run approximately every 15 minutes. The EDL server uses CLARiiON NaviCLI to obtain status of the CLARiiON array. If there is a fault, event number 19103 will be logged in the EDL Console and, if so configured, an SNMP trap sent. Faults are disk drive failure or loss of a power supply, for example. Loss of write cache is a second problem that is monitored. If the CLARiiON write cache is disabled, event number 19102 will be logged in the EDL Console (and an SNMP trap sent if so configured). The most common reason for write cache becoming disabled is loss of one of the mains circuits supplying power to the CLARiiON. Other than a reduction of performance, there is no ill effect.

For health check to run properly, network connectivity from the EDL server to the CLARiiON SPs is required. If your environment does not provide connectivity, event 19101 will be logged as a warning.

It is not necessary to configure the IP addresses of the CLARiiON SPs in the EDL Console. EDL software obtains the addresses by sending SCSI Inquiry commands to its various fibre devices, thereby automatically retrieving them.

**Using SNMP management software**

A number of SNMP software products are on the market. Customers can use any one of them and can also use their own software to integrate with these products. Accordingly, it is not possible to provide examples of how to incorporate EDL traps with CA Unicenter, HP OpenView, Microsoft MOM, and NetCool, to name a few of what seem to be the more popular packages. We will however provide examples of the easy-to-use Trap Console from CSCare, Inc. First, it is necessary to obtain the EDL MIB.
Obtain the EMC Disk Library MIB

EMC does not guarantee that every trap listener and SNMP management software will be able to compile the MIB. EMC personnel can request the text files from CLARiiON Engineering by opening a TS Pre-Sales Support Center call for EMC Disk Library and using the keyword MIB. Place the MIB file on the system running Trap Console.

Compile the EDL MIB

To compile the MIB into Trap Console, from the MIB Manager tab, upload the MIB file. It will appear in the list of MIBs (note IPSTOR-MIB.txt).

![Figure 7. Upload the EDL MIB into trap management software](image)

Click the Compile button and verify that it compiles without errors.
At this point, if you receive a trap from your EDL server, the trap will be listed in Trap Console’s log with the name of the trap (such as informationalLogMessage). If you get the OID (numbers separated by dots) rather than a name from the EDL MIB, you know that the MIB was not yet compiled. For example, note the informationalLogMessage and errorLogMessage.

Send a “test” trap

Although EDL Console does not have a button to send a test trap, it’s easy to generate an informational trap. Just disconnect your EDL Console from the EDL server and reconnect. You’ll get a few informational traps—as long as you’ve set the level to “informational” when configuring SNMP on the EDL server. You may generate an error trap by disconnecting from the EDL Console and entering an incorrect password or username at attempting to reconnect.

This test checks traps originating at the server per se. However, an additional test of CLARiiON traps is in order. To do this, from NaviUI, send a test trap from the callhome template in the monitor tab or log in to the EDL server and enter these commands:

```
navicli –h spa_ip_address inserttestevent
navicli –h spb_ip_address inserttestevent
```

After about 10 minutes (it takes a few minutes for the CLARiiON array to issue the trap), you should see that the EDL server has logged two CLARiiON traps in the Console log and you should also receive two SNMP traps. Failure of the console to log traps from the CLARiiON array – or inability to find the callhome template in the monitor tab of NaviUI – probably indicates that the CLARiiON has not been properly set up to forward events as SNMP traps. Refer to the EMC CLARiiON Disk Library Administrator’s Guide.
The *callhome template* installed on the CLARiiON prevents almost all informational events from being sent as traps, but it does send the test trap.

You can configure Trap Console to examine the contents of traps received and to perform specific actions. For example, if you receive a EDL trap due to a CLARiiON error or critical-level trap, you can have e-mail sent to your cell phone. Remember that the EDL traps correspond to these: 11786 and 11787. So, you can ask Trap Console to match either of those numbers in the trap text field. However, rather than use just the numbers themselves, you use the numbers with brackets around them. After all, the numbers *could* possibly be found elsewhere in the trap, but the EDL event IDs have brackets around them.

So, as shown in Figure 10, you define a rule for Trap Console to match either of the event IDs.

---

**Figure 9. Verify receipt of EDL traps**

You can configure Trap Console to examine the contents of traps received and to perform specific actions. For example, if you receive a EDL trap due to a CLARiiON error or critical-level trap, you can have e-mail sent to your cell phone. Remember that the EDL traps correspond to these: 11786 and 11787. So, you can ask Trap Console to match either of those numbers in the trap text field. However, rather than use just the numbers themselves, you use the numbers with brackets around them. After all, the numbers *could* possibly be found elsewhere in the trap, but the EDL event IDs have brackets around them.

So, as shown in Figure 10, you define a rule for Trap Console to match either of the event IDs.
Figure 10. Configuring an expression for a rule

As shown in Figure 11, this is part of a rule we’ll call “CLARiiON Error Trap” (though it’s for both CLARiiON error and critical event traps).
Figure 11. Rules defined for various trap scenarios

As Figure 12 illustrates, “Call hardware support” is the defined action. That, in turn, sends a message to the user that says “Hardware problem,” followed by the body of the trap (what Trap Console’s varval1 represents).
Figure 12. Action taken when the trap rule is satisfied

The only other detail is to specify the mail server (SMTP server) in the E-mail tab of the Preferences screen, as shown in Figure 13.
That completes the Trap Console setup. Now you can verify that traps you’ve received will be acted on as desired. To do this:

1. Click the **Logs** tab to make sure the basic trap comes through (as shown earlier in the “Send a “test” trap” section on page 17).

2. Click **SMTP Log** in the Preferences screen to make sure that a message(s) was actually sent.

**Figure 13. Specifying the SNMP server**

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
If a message was not sent, you may have to refine the rule. In addition, check the SMTP log to make sure that you have specified the correct SMTP server and that it’s not returning any errors.

Figure 14. Verify SMTP server communication

**EMC Disk Library MIB**

The “Compile the EDL MIB” section on page 16 describes how to obtain the EDL MIB. It may be compiled into your SNMP management software or MIB walker. The MIB contains EDL-specific objects. Remember that EDL also responds with various required RFC1213 objects.

**Walking the MIB**

Upon installing your EDL, you will want to specify values for the RFC1213 required objects that may be altered for your environment. Enter values for SysLocation and SysContact on the SNMP Maintenance tab. These settings are shown in Figure 15.
Figure 14. Specify SysContact and SysLocation via EDL Console

If the MIB is walked, you will see the following objects (these are just the first few objects in the RFC1213 tree):

sysDescr.0:--> EMC CLARiiON Disk Library
sysObjectID.0:-->.iso.org.dod.internet.private.enterprises.ipstor.5.10
sysUpTime.0:-->0 hours, 12 minutes, 24 seconds.
sysContact.0:-->John Doe
sysName.0:-->DullsabiA
sysLocation.0:-->Engineering Lab A

EDL-specific MIB

Through SNMP you can read many EDL-specific objects, including:

- The WWPNs of the various fibre ports
- The virtual tape libraries that have been created—their names, number of tape drives, tape drive types, and so on
- The virtual tapes you’ve created, including barcodes, amount of data written, and location (which virtual libraries they reside in)
- Names of SAN clients you’ve created, their WWPNs, and which virtual tape drives and libraries have been assigned to them

In short, you can use SNMP to develop a tree structure just like the EDL Console provides.

The information contained in these responses can be filtered by sophisticated NMS software. For example, an NMS workstation could be set up to send an alert to key personnel when connUnitStatus (OID
1.3.6.1.3.94.1.6.1.6) returns a value of either 4 (warning) or 5 (failed). In this way, the FA MIB and its sub-agents on CLARiiON CX series storage systems allow network administrators to monitor CLARiiONs within their larger network monitoring framework.

This is the benefit of the MIB:

Customers may monitor EMC Disk Libraries from within their management framework via SNMP queries.

### Table 3. Examples of OIDs contained in an EMC Disk Library MIB

<table>
<thead>
<tr>
<th>OID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1.3.6.1.2.1.1.4</td>
<td>sysContact</td>
</tr>
<tr>
<td>.1.3.6.1.2.1.1.5</td>
<td>sysName</td>
</tr>
<tr>
<td>.1.3.6.1.2.1.1.6</td>
<td>sysLocation</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.3.3.8.1.1</td>
<td>vlibNumber (total number of virtual tape libraries)</td>
</tr>
<tr>
<td>.1.3.6.1.4.1.7368.3.3.8.3.1</td>
<td>vtapeNumber (total number of virtual tapes)</td>
</tr>
</tbody>
</table>

Further information on certain EDL-specific SNMP objects is provided in the next section.

**EDL throughput counters**

SNMP objects are provided so that you can check on the throughput of the EDL fibre ports. Following is an example from walking the MIB.

Values are cumulative megabytes since the EDL server booted. You would have to calculate throughput using your SNMP management software or custom application by reading the appropriate values every 10 seconds or so, calculating the MB transferred over that time, and then dividing by the number of seconds to determine throughput.

These values are available starting with EDL 2.0. The following example is also taken with EDL 2.0 hardware (the EDL 2.0 server has 12 fibre ports or adapters, whereas prior EDL servers have eight ports). Note also that throughput figures are only available for “target” ports. Additionally, port numbering starts at 1 whereas EDL documentation and the EDL Console number ports starting at zero. For example, only ports 1 and 3 (by default target ports) are showing data transferred.

```
adapterMBRead.1:-->0
adapterMBRead.2:-->404363
adapterMBRead.3:-->0
adapterMBRead.4:-->351602
adapterMBRead.5:-->0
adapterMBRead.6:-->0
adapterMBRead.7:-->0
adapterMBRead.8:-->0
adapterMBRead.9:-->0
adapterMBRead.10:-->0
adapterMBRead.11:-->0
adapterMBRead.12:-->0
adapterMBWritten.1:-->0
adapterMBWritten.2:-->405181
adapterMBWritten.3:-->0
adapterMBWritten.4:-->356903
adapterMBWritten.5:-->0
adapterMBWritten.6:-->0
adapterMBWritten.7:-->0
adapterMBWritten.8:-->0
adapterMBWritten.9:-->0
adapterMBWritten.10:-->0
adapterMBWritten.11:-->0
```
Throughput counters may be used to determine whether there are bottlenecks in your SAN environment. No more than 200 MB/s can be sustained on a given target port for either reads or writes, so if you have 190 MB/s through one target port but only 20 MB/s through the others, you may want to consider reassigning (in EDL) or rezoning (in the fabric switch) backup servers so they are more evenly balanced across the EDL target ports.

**EDL hardware monitoring: EDL 2.0 and later hardware**

It is possible to monitor the health of the EDL server hardware. Following is the entire tree of server hardware objects (for EDL 2.0 server hardware). Note that this example is for the case of a power supply lacking mains power, as in a trap described in the “CLARiiON health” section on page 15. The particular object of interest is sensorStatus.55, which is reported as “Redundancy Lost,” meaning there is no power to one of the power supplies. The name of the value is given by sensorName.55, “PS Redundancy (Power Supply).” As shown next, many other server hardware items can be monitored such as power supply voltages, fan speeds, CPU temperature, and so on.

EDL 3.0 and 4.0 hardware have similar sensors. Refer to the MIB for your particular EDL version.

```
sensorIndex.1:--->1
sensorIndex.2:--->2
sensorIndex.3:--->3
sensorIndex.4:--->4
sensorIndex.5:--->5
sensorIndex.6:--->6
sensorIndex.7:--->7
sensorIndex.8:--->8
sensorIndex.9:--->9
sensorIndex.10:--->10
sensorIndex.11:--->11
sensorIndex.12:--->12
sensorIndex.13:--->13
sensorIndex.14:--->14
sensorIndex.15:--->15
sensorIndex.16:--->16
sensorIndex.17:--->17
sensorIndex.18:--->18
sensorIndex.19:--->19
sensorIndex.20:--->20
sensorIndex.21:--->21
sensorIndex.22:--->22
sensorIndex.23:--->23
sensorIndex.24:--->24
sensorIndex.25:--->25
sensorIndex.26:--->26
sensorIndex.27:--->27
sensorIndex.28:--->28
sensorIndex.29:--->29
sensorIndex.30:--->30
sensorIndex.31:--->31
sensorIndex.32:--->32
sensorIndex.33:--->33
sensorIndex.34:--->34
sensorIndex.35:--->35
sensorIndex.36:--->36
```
Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
sensorName.38:-->FAN 1 RPM (Fan)
sensorName.39:-->FAN 2 RPM (Fan)
sensorName.40:-->FAN 3 RPM (Fan)
sensorName.41:-->FAN 4 RPM (Fan)
sensorName.42:-->FAN 5 RPM (Fan)
sensorName.43:-->FAN 6 RPM (Fan)
sensorName.44:-->FAN 7 RPM (Fan)
sensorName.45:-->FAN 8 RPM (Fan)
sensorName.46:-->Status (Processor)
sensorName.47:-->Status (Processor)
sensorName.48:-->Status (Power Supply)
sensorName.49:-->Status (Power Supply)
sensorName.50:-->VRM (Power Supply)
sensorName.51:-->VRM (Power Supply)
sensorName.52:-->OS Watchdog (Watchdog 2)
sensorName.53:-->SEL (Event Logging Disabled)
sensorName.54:-->Intrusion (Platform Chassis Int
sensorName.55:-->PS Redundancy (Power Supply)
sensorName.56:-->Fan Redundancy (Fan)
sensorStatus.1:-->OK
sensorStatus.2:-->OK
sensorStatus.3:-->OK
sensorStatus.4:-->OK
sensorStatus.5:-->OK
sensorStatus.6:-->OK
sensorStatus.7:-->OK
sensorStatus.8:-->OK
sensorStatus.9:-->OK
sensorStatus.10:-->OK
sensorStatus.11:-->OK
sensorStatus.12:-->OK
sensorStatus.13:-->OK
sensorStatus.14:-->OK
sensorStatus.15:-->OK
sensorStatus.16:-->State Deasserted
sensorStatus.17:-->State Deasserted
sensorStatus.18:-->State Deasserted
sensorStatus.19:-->State Deasserted
sensorStatus.20:-->State Deasserted
sensorStatus.21:-->State Deasserted
sensorStatus.22:-->State Deasserted
sensorStatus.23:-->State Deasserted
sensorStatus.24:-->State Deasserted
sensorStatus.25:-->State Deasserted
sensorStatus.26:-->OK
sensorStatus.27:-->OK
sensorStatus.28:-->OK
sensorStatus.29:-->OK
sensorStatus.30:-->OK
sensorStatus.31:-->Unknown
sensorStatus.32:-->OK
sensorStatus.33:-->OK
sensorStatus.34:-->OK
sensorStatus.35:-->OK
sensorStatus.36:-->OK
sensorStatus.37:-->OK
sensorStatus.38:-->OK
sensorStatus.39:-->OK
sensorStatus.40:-->OK
sensorStatus.41:-->OK
sensorStatus.42:-->OK
sensorStatus.43:-->OK
sensorStatus.44:-->OK
sensorStatus.45:-->OK
sensorStatus.46:-->OK
sensorStatus.47:-->OK
sensorStatus.48:-->OK
sensorStatus.49:-->Unknown
sensorStatus.50:-->OK
sensorStatus.51:-->OK
sensorStatus.52:-->OK
sensorStatus.53:-->Unknown
sensorStatus.54:-->OK
sensorStatus.55:-->Redundancy Lost
sensorStatus.56:-->Fully Redundant
sensorReading.1:-->38.00 Degrees C
sensorReading.2:-->39.00 Degrees C
sensorReading.3:-->23.00 Degrees C
sensorReading.4:-->33.00 Degrees C
sensorReading.5:-->32.00 Degrees C
sensorReading.6:-->40.00 Degrees C
sensorReading.7:-->40.00 Degrees C
sensorReading.8:-->50.00 Degrees C
sensorReading.9:-->50.00 Degrees C
sensorReading.10:-->23.00 Degrees C
sensorReading.11:-->40.00 Degrees C
sensorReading.12:-->40.00 Degrees C
sensorReading.13:-->40.00 Degrees C
sensorReading.14:-->40.00 Degrees C
sensorReading.15:-->3.15 Volts
sensorReading.16:-->3.10 Volts
sensorReading.17:-->3.10 Volts
sensorReading.18:-->3.10 Volts
sensorReading.19:-->3.10 Volts
sensorReading.20:-->3.10 Volts
sensorReading.21:-->3.10 Volts
sensorReading.22:-->3.10 Volts
sensorReading.23:-->3.10 Volts
sensorReading.24:-->3.10 Volts
sensorReading.25:-->3.10 Volts
sensorReading.26:-->3.10 Volts
sensorReading.27:-->3.10 Volts
sensorReading.28:-->3.10 Volts
sensorReading.29:-->3.10 Volts
sensorReading.30:-->3.10 Volts
sensorReading.31:-->3.10 Volts
sensorReading.32:-->4875.00 RPM
sensorReading.33:-->4875.00 RPM
sensorReading.34:-->4875.00 RPM
sensorReading.35:-->4950.00 RPM
sensorReading.36:-->4875.00 RPM
sensorReading.37:-->4875.00 RPM
sensorReading.38:-->1800.00 RPM
sensorReading.39:-->1800.00 RPM

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
sensorReading.40: --> 1800.00 RPM
sensorReading.41: --> 1800.00 RPM
sensorReading.42: --> 1800.00 RPM
sensorReading.43: --> 1800.00 RPM
sensorReading.44: --> 1800.00 RPM
sensorReading.45: --> 1800.00 RPM
sensorReading.46: -->
sensorReading.47: -->
sensorReading.48: -->
sensorReading.49: -->
sensorReading.50: -->
sensorReading.51: -->
sensorReading.52: -->
sensorReading.53: -->
sensorReading.54: -->
sensorReading.55: -->
sensorReading.56: -->
sensorLowerNonCritical.1: --> 10.00 Degrees C
sensorLowerNonCritical.2: --> 10.00 Degrees C
sensorLowerNonCritical.3: --> 8.00 Degrees C
sensorLowerNonCritical.4: --> 8.00 Degrees C
sensorLowerNonCritical.5: --> 8.00 Degrees C
sensorLowerNonCritical.6: --> 9.00 Degrees C
sensorLowerNonCritical.7: --> 9.00 Degrees C
sensorLowerNonCritical.8: --> 10.00 Degrees C
sensorLowerNonCritical.9: --> 10.00 Degrees C
sensorLowerNonCritical.10: --> 8.00 Degrees C
sensorLowerNonCritical.11: --> 8.00 Degrees C
sensorLowerNonCritical.12: --> 8.00 Degrees C
sensorLowerNonCritical.13: --> 9.00 Degrees C
sensorLowerNonCritical.14: --> 9.00 Degrees C
sensorLowerNonCritical.15: --> 2.94 Volts
sensorLowerNonCritical.16: -->
sensorLowerNonCritical.17: -->
sensorLowerNonCritical.18: -->
sensorLowerNonCritical.19: -->
sensorLowerNonCritical.20: -->
sensorLowerNonCritical.21: -->
sensorLowerNonCritical.22: -->
sensorLowerNonCritical.23: -->
sensorLowerNonCritical.24: -->
sensorLowerNonCritical.25: -->
sensorLowerNonCritical.26: --> 2.94 Volts
sensorLowerNonCritical.27: -->
sensorLowerNonCritical.28: -->
sensorLowerNonCritical.29: -->
sensorLowerNonCritical.30: -->
sensorLowerNonCritical.31: -->
sensorLowerNonCritical.32: --> 1875.00 RPM
sensorLowerNonCritical.33: --> 1875.00 RPM
sensorLowerNonCritical.34: --> 1875.00 RPM
sensorLowerNonCritical.35: --> 1875.00 RPM
sensorLowerNonCritical.36: --> 1875.00 RPM
sensorLowerNonCritical.37: --> 1875.00 RPM
sensorLowerNonCritical.38: --> 2400.00 RPM
sensorLowerNonCritical.39: --> 2400.00 RPM
sensorLowerNonCritical.40: --> 2400.00 RPM
sensorLowerNonCritical.41:-->2400.00 RPM
sensorLowerNonCritical.42:-->1050.00 RPM
sensorLowerNonCritical.43:-->1050.00 RPM
sensorLowerNonCritical.44:-->1050.00 RPM
sensorLowerNonCritical.45:-->1050.00 RPM
sensorLowerNonCritical.46:-->
sensorLowerNonCritical.47:-->
sensorLowerNonCritical.48:-->
sensorLowerNonCritical.49:-->
sensorLowerNonCritical.50:-->
sensorLowerNonCritical.51:-->
sensorLowerNonCritical.52:-->
sensorLowerNonCritical.53:-->
sensorLowerNonCritical.54:-->
sensorLowerNonCritical.55:-->
sensorLowerNonCritical.56:-->
sensorUpperNonCritical.1:-->85.00 Degrees C
sensorUpperNonCritical.2:-->85.00 Degrees C
sensorUpperNonCritical.3:-->42.00 Degrees C
sensorUpperNonCritical.4:-->67.00 Degrees C
sensorUpperNonCritical.5:-->57.00 Degrees C
sensorUpperNonCritical.6:-->78.00 Degrees C
sensorUpperNonCritical.7:-->78.00 Degrees C
sensorUpperNonCritical.8:-->85.00 Degrees C
sensorUpperNonCritical.9:-->85.00 Degrees C
sensorUpperNonCritical.10:-->42.00 Degrees C
sensorUpperNonCritical.11:-->67.00 Degrees C
sensorUpperNonCritical.12:-->57.00 Degrees C
sensorUpperNonCritical.13:-->78.00 Degrees C
sensorUpperNonCritical.14:-->78.00 Degrees C
sensorUpperNonCritical.15:-->2.55 Volts
sensorUpperNonCritical.16:-->
sensorUpperNonCritical.17:-->
sensorUpperNonCritical.18:-->
sensorUpperNonCritical.19:-->
sensorUpperNonCritical.20:-->
sensorUpperNonCritical.21:-->
sensorUpperNonCritical.22:-->
sensorUpperNonCritical.23:-->
sensorUpperNonCritical.24:-->
sensorUpperNonCritical.25:-->
sensorUpperNonCritical.26:-->2.55 Volts
sensorUpperNonCritical.27:-->
sensorUpperNonCritical.28:-->
sensorUpperNonCritical.29:-->
sensorUpperNonCritical.30:-->
sensorUpperNonCritical.31:-->
sensorUpperNonCritical.32:-->19125.00 RPM
sensorUpperNonCritical.33:-->19125.00 RPM
sensorUpperNonCritical.34:-->19125.00 RPM
sensorUpperNonCritical.35:-->19125.00 RPM
sensorUpperNonCritical.36:-->19125.00 RPM
sensorUpperNonCritical.37:-->19125.00 RPM
sensorUpperNonCritical.38:-->19125.00 RPM
sensorUpperNonCritical.39:-->19125.00 RPM
sensorUpperNonCritical.40:-->19125.00 RPM
sensorUpperNonCritical.41:-->19125.00 RPM

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
sensorUpperNonCritical.42:-->19125.00 RPM
sensorUpperNonCritical.43:-->19125.00 RPM
sensorUpperNonCritical.44:-->19125.00 RPM
sensorUpperNonCritical.45:-->19125.00 RPM
sensorUpperNonCritical.46:-->  
sensorUpperNonCritical.47:-->  
sensorUpperNonCritical.48:-->  
sensorUpperNonCritical.49:-->  
sensorUpperNonCritical.50:-->  
sensorUpperNonCritical.51:-->  
sensorUpperNonCritical.52:-->  
sensorUpperNonCritical.53:-->  
sensorUpperNonCritical.54:-->  
sensorUpperNonCritical.55:-->  
sensorUpperNonCritical.56:-->  
sensorLowerCritical.1:-->5.00 Degrees C
sensorLowerCritical.2:-->5.00 Degrees C
sensorLowerCritical.3:-->3.00 Degrees C
sensorLowerCritical.4:-->3.00 Degrees C
sensorLowerCritical.5:-->3.00 Degrees C
sensorLowerCritical.6:-->4.00 Degrees C
sensorLowerCritical.7:-->4.00 Degrees C
sensorLowerCritical.8:-->5.00 Degrees C
sensorLowerCritical.9:-->5.00 Degrees C
sensorLowerCritical.10:-->3.00 Degrees C
sensorLowerCritical.11:-->3.00 Degrees C
sensorLowerCritical.12:-->3.00 Degrees C
sensorLowerCritical.13:-->4.00 Degrees C
sensorLowerCritical.14:-->4.00 Degrees C
sensorLowerCritical.15:-->2.64 Volts
sensorLowerCritical.16:-->  
sensorLowerCritical.17:-->  
sensorLowerCritical.18:-->  
sensorLowerCritical.19:-->  
sensorLowerCritical.20:-->  
sensorLowerCritical.21:-->  
sensorLowerCritical.22:-->  
sensorLowerCritical.23:-->  
sensorLowerCritical.24:-->  
sensorLowerCritical.25:-->  
sensorLowerCritical.26:-->2.64 Volts
sensorLowerCritical.27:-->  
sensorLowerCritical.28:-->  
sensorLowerCritical.29:-->  
sensorLowerCritical.30:-->  
sensorLowerCritical.31:-->  
sensorLowerCritical.32:-->1575.00 RPM
sensorLowerCritical.33:-->1575.00 RPM
sensorLowerCritical.34:-->1575.00 RPM
sensorLowerCritical.35:-->1575.00 RPM
sensorLowerCritical.36:-->1575.00 RPM
sensorLowerCritical.37:-->1575.00 RPM
sensorLowerCritical.38:-->2025.00 RPM
sensorLowerCritical.39:-->2025.00 RPM
sensorLowerCritical.40:-->2025.00 RPM
sensorLowerCritical.41:-->2025.00 RPM
sensorLowerCritical.42:-->900.00 RPM

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
sensorLowerCritical.43:-->900.00 RPM
sensorLowerCritical.44:-->900.00 RPM
sensorLowerCritical.45:-->900.00 RPM
sensorLowerCritical.46:-->
sensorLowerCritical.47:-->
sensorLowerCritical.48:-->
sensorLowerCritical.49:-->
sensorLowerCritical.50:-->
sensorLowerCritical.51:-->
sensorLowerCritical.52:-->
sensorLowerCritical.53:-->
sensorLowerCritical.54:-->
sensorLowerCritical.55:-->
sensorUpperCritical.1:-->90.00 Degrees C
sensorUpperCritical.2:-->90.00 Degrees C
sensorUpperCritical.3:-->47.00 Degrees C
sensorUpperCritical.4:-->72.00 Degrees C
sensorUpperCritical.5:-->62.00 Degrees C
sensorUpperCritical.6:-->88.00 Degrees C
sensorUpperCritical.7:-->88.00 Degrees C
sensorUpperCritical.8:-->90.00 Degrees C
sensorUpperCritical.9:-->90.00 Degrees C
sensorUpperCritical.10:-->47.00 Degrees C
sensorUpperCritical.11:-->72.00 Degrees C
sensorUpperCritical.12:-->62.00 Degrees C
sensorUpperCritical.13:-->88.00 Degrees C
sensorUpperCritical.14:-->88.00 Degrees C
sensorUpperCritical.15:-->2.62 Volts
sensorUpperCritical.16:-->
sensorUpperCritical.17:-->
sensorUpperCritical.18:-->
sensorUpperCritical.19:-->
sensorUpperCritical.20:-->
sensorUpperCritical.21:-->
sensorUpperCritical.22:-->
sensorUpperCritical.23:-->
sensorUpperCritical.24:-->
sensorUpperCritical.25:-->
sensorUpperCritical.26:-->2.62 Volts
sensorUpperCritical.27:-->
sensorUpperCritical.28:-->
sensorUpperCritical.29:-->
sensorUpperCritical.30:-->
sensorUpperCritical.31:-->
sensorUpperCritical.32:-->19125.00 RPM
sensorUpperCritical.33:-->19125.00 RPM
sensorUpperCritical.34:-->19125.00 RPM
sensorUpperCritical.35:-->19125.00 RPM
sensorUpperCritical.36:-->19125.00 RPM
sensorUpperCritical.37:-->19125.00 RPM
sensorUpperCritical.38:-->19125.00 RPM
sensorUpperCritical.39:-->19125.00 RPM
sensorUpperCritical.40:-->19125.00 RPM
sensorUpperCritical.41:-->19125.00 RPM
sensorUpperCritical.42:-->19125.00 RPM
sensorUpperCritical.43:-->19125.00 RPM
sensorUpperCritical.44: --> 19125.00 RPM
sensorUpperCritical.45: --> 19125.00 RPM
sensorUpperCritical.46: --> 
sensorUpperCritical.47: --> 
sensorUpperCritical.48: --> 
sensorUpperCritical.49: --> 
sensorUpperCritical.50: --> 
sensorUpperCritical.51: --> 
sensorUpperCritical.52: --> 
sensorUpperCritical.53: --> 
sensorUpperCritical.54: --> 
sensorUpperCritical.55: --> 
sensorUpperCritical.56: --> 

sensorLowerNonRecoverable.1: --> -128.00 Degrees C 
sensorLowerNonRecoverable.2: --> -128.00 Degrees C 
sensorLowerNonRecoverable.3: --> -128.00 Degrees C 
sensorLowerNonRecoverable.4: --> -128.00 Degrees C 
sensorLowerNonRecoverable.5: --> -128.00 Degrees C 
sensorLowerNonRecoverable.6: --> -128.00 Degrees C 
sensorLowerNonRecoverable.7: --> -128.00 Degrees C 
sensorLowerNonRecoverable.8: --> -128.00 Degrees C 
sensorLowerNonRecoverable.9: --> -128.00 Degrees C 
sensorLowerNonRecoverable.10: --> -128.00 Degrees C 
sensorLowerNonRecoverable.11: --> -128.00 Degrees C 
sensorLowerNonRecoverable.12: --> -128.00 Degrees C 
sensorLowerNonRecoverable.13: --> -128.00 Degrees C 
sensorLowerNonRecoverable.14: --> -128.00 Degrees C 
sensorLowerNonRecoverable.15: --> 0.00 Volts 
sensorLowerNonRecoverable.16: --> 
sensorLowerNonRecoverable.17: --> 
sensorLowerNonRecoverable.18: --> 
sensorLowerNonRecoverable.19: --> 
sensorLowerNonRecoverable.20: --> 
sensorLowerNonRecoverable.21: --> 
sensorLowerNonRecoverable.22: --> 
sensorLowerNonRecoverable.23: --> 
sensorLowerNonRecoverable.24: --> 
sensorLowerNonRecoverable.25: --> 
sensorLowerNonRecoverable.26: --> 0.00 Volts 
sensorLowerNonRecoverable.27: --> 
sensorLowerNonRecoverable.28: --> 
sensorLowerNonRecoverable.29: --> 
sensorLowerNonRecoverable.30: --> 
sensorLowerNonRecoverable.31: --> 
sensorLowerNonRecoverable.32: --> 0.00 RPM 
sensorLowerNonRecoverable.33: --> 0.00 RPM 
sensorLowerNonRecoverable.34: --> 0.00 RPM 
sensorLowerNonRecoverable.35: --> 0.00 RPM 
sensorLowerNonRecoverable.36: --> 0.00 RPM 
sensorLowerNonRecoverable.37: --> 0.00 RPM 
sensorLowerNonRecoverable.38: --> 0.00 RPM 
sensorLowerNonRecoverable.39: --> 0.00 RPM 
sensorLowerNonRecoverable.40: --> 0.00 RPM 
sensorLowerNonRecoverable.41: --> 0.00 RPM 
sensorLowerNonRecoverable.42: --> 0.00 RPM 
sensorLowerNonRecoverable.43: --> 0.00 RPM 
sensorLowerNonRecoverable.44: --> 0.00 RPM
sensorLowerNonRecoverable.45:-->0.00 RPM
sensorLowerNonRecoverable.46:--> 
sensorLowerNonRecoverable.47:--> 
sensorLowerNonRecoverable.48:--> 
sensorLowerNonRecoverable.49:--> 
sensorLowerNonRecoverable.50:--> 
sensorLowerNonRecoverable.51:--> 
sensorLowerNonRecoverable.52:--> 
sensorLowerNonRecoverable.53:--> 
sensorLowerNonRecoverable.54:--> 
sensorLowerNonRecoverable.55:--> 
sensorLowerNonRecoverable.56:--> 
sensorUpperNonRecoverable.1:-->127.00 Degrees C
sensorUpperNonRecoverable.2:-->127.00 Degrees C
sensorUpperNonRecoverable.3:-->127.00 Degrees C
sensorUpperNonRecoverable.4:-->127.00 Degrees C
sensorUpperNonRecoverable.5:-->127.00 Degrees C
sensorUpperNonRecoverable.6:-->127.00 Degrees C
sensorUpperNonRecoverable.7:-->127.00 Degrees C
sensorUpperNonRecoverable.8:-->127.00 Degrees C
sensorUpperNonRecoverable.9:-->127.00 Degrees C
sensorUpperNonRecoverable.10:-->127.00 Degrees C
sensorUpperNonRecoverable.11:-->127.00 Degrees C
sensorUpperNonRecoverable.12:-->127.00 Degrees C
sensorUpperNonRecoverable.13:-->127.00 Degrees C
sensorUpperNonRecoverable.14:-->127.00 Degrees C
sensorUpperNonRecoverable.15:-->3.29 Volts
sensorUpperNonRecoverable.16:--> 
sensorUpperNonRecoverable.17:--> 
sensorUpperNonRecoverable.18:--> 
sensorUpperNonRecoverable.19:--> 
sensorUpperNonRecoverable.20:--> 
sensorUpperNonRecoverable.21:--> 
sensorUpperNonRecoverable.22:--> 
sensorUpperNonRecoverable.23:--> 
sensorUpperNonRecoverable.24:--> 
sensorUpperNonRecoverable.25:--> 
sensorUpperNonRecoverable.26:-->3.29 Volts
sensorUpperNonRecoverable.27:--> 
sensorUpperNonRecoverable.28:--> 
sensorUpperNonRecoverable.29:--> 
sensorUpperNonRecoverable.30:--> 
sensorUpperNonRecoverable.31:--> 
sensorUpperNonRecoverable.32:-->19125.00 RPM
sensorUpperNonRecoverable.33:-->19125.00 RPM
sensorUpperNonRecoverable.34:-->19125.00 RPM
sensorUpperNonRecoverable.35:-->19125.00 RPM
sensorUpperNonRecoverable.36:-->19125.00 RPM
sensorUpperNonRecoverable.37:-->19125.00 RPM
sensorUpperNonRecoverable.38:-->19125.00 RPM
sensorUpperNonRecoverable.39:-->19125.00 RPM
sensorUpperNonRecoverable.40:-->19125.00 RPM
sensorUpperNonRecoverable.41:-->19125.00 RPM
sensorUpperNonRecoverable.42:-->19125.00 RPM
sensorUpperNonRecoverable.43:-->19125.00 RPM
sensorUpperNonRecoverable.44:-->19125.00 RPM
sensorUpperNonRecoverable.45:-->19125.00 RPM
Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review 36
Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review

sensorMinReading.47:
sensorMinReading.48:
sensorMinReading.49:
sensorMinReading.50:
sensorMinReading.51:
sensorMinReading.52:
sensorMinReading.53:
sensorMinReading.54:
sensorMinReading.55:
sensorMinReading.56:
sensorMaxReading.1:
127.00 Degrees C
sensorMaxReading.2:
127.00 Degrees C
csensorMaxReading.3:
127.00 Degrees C
csensorMaxReading.4:
127.00 Degrees C
csensorMaxReading.5:
127.00 Degrees C
csensorMaxReading.6:
127.00 Degrees C
csensorMaxReading.7:
127.00 Degrees C
csensorMaxReading.8:
127.00 Degrees C
csensorMaxReading.9:
127.00 Degrees C
csensorMaxReading.10:
127.00 Degrees C
csensorMaxReading.11:
127.00 Degrees C
csensorMaxReading.12:
127.00 Degrees C
csensorMaxReading.13:
127.00 Degrees C
csensorMaxReading.14:
127.00 Degrees C
csensorMaxReading.15:
3.29 Volts
csensorMaxReading.16:

csensorMaxReading.17:

csensorMaxReading.18:

csensorMaxReading.19:

csensorMaxReading.20:

csensorMaxReading.21:

csensorMaxReading.22:

csensorMaxReading.23:

csensorMaxReading.24:

csensorMaxReading.25:

csensorMaxReading.26:
3.29 Volts
csensorMaxReading.27:

csensorMaxReading.28:

csensorMaxReading.29:

csensorMaxReading.30:

csensorMaxReading.31:

csensorMaxReading.32:
19125.00 RPM
csensorMaxReading.33:
19125.00 RPM
csensorMaxReading.34:
19125.00 RPM
csensorMaxReading.35:
19125.00 RPM
csensorMaxReading.36:
19125.00 RPM
csensorMaxReading.37:
19125.00 RPM
csensorMaxReading.38:
19125.00 RPM
csensorMaxReading.39:
19125.00 RPM
csensorMaxReading.40:
19125.00 RPM
csensorMaxReading.41:
19125.00 RPM
csensorMaxReading.42:
19125.00 RPM
csensorMaxReading.43:
19125.00 RPM
csensorMaxReading.44:
19125.00 RPM
csensorMaxReading.45:
19125.00 RPM
csensorMaxReading.46:

csensorMaxReading.47:
sensorMaxReading.48:-->  
sensorMaxReading.49:-->  
sensorMaxReading.50:-->  
sensorMaxReading.51:-->  
sensorMaxReading.52:-->  
sensorMaxReading.53:-->  
sensorMaxReading.54:-->  
sensorMaxReading.55:-->  
sensorMaxReading.56:-->  
sensorNormalMin.1:--> 11.00 Degrees C  
sensorNormalMin.2:--> 11.00 Degrees C  
sensorNormalMin.3:--> 11.00 Degrees C  
sensorNormalMin.4:--> 11.00 Degrees C  
sensorNormalMin.5:--> 11.00 Degrees C  
sensorNormalMin.6:--> 11.00 Degrees C  
sensorNormalMin.7:--> 11.00 Degrees C  
sensorNormalMin.8:--> 11.00 Degrees C  
sensorNormalMin.9:--> 11.00 Degrees C  
sensorNormalMin.10:--> 11.00 Degrees C  
sensorNormalMin.11:--> 11.00 Degrees C  
sensorNormalMin.12:--> 11.00 Degrees C  
sensorNormalMin.13:--> 11.00 Degrees C  
sensorNormalMin.14:--> 11.00 Degrees C  
sensorNormalMin.15:--> 1.79 Volts  
sensorNormalMin.16:-->  
sensorNormalMin.17:-->  
sensorNormalMin.18:-->  
sensorNormalMin.19:-->  
sensorNormalMin.20:-->  
sensorNormalMin.21:-->  
sensorNormalMin.22:-->  
sensorNormalMin.23:-->  
sensorNormalMin.24:-->  
sensorNormalMin.25:-->  
sensorNormalMin.26:--> 1.79 Volts  
sensorNormalMin.27:-->  
sensorNormalMin.28:-->  
sensorNormalMin.29:-->  
sensorNormalMin.30:-->  
sensorNormalMin.31:-->  
sensorNormalMin.32:--> 10425.00 RPM  
sensorNormalMin.33:--> 10425.00 RPM  
sensorNormalMin.34:--> 10425.00 RPM  
sensorNormalMin.35:--> 10425.00 RPM  
sensorNormalMin.36:--> 10425.00 RPM  
sensorNormalMin.37:--> 10425.00 RPM  
sensorNormalMin.38:--> 10425.00 RPM  
sensorNormalMin.39:--> 10425.00 RPM  
sensorNormalMin.40:--> 10425.00 RPM  
sensorNormalMin.41:--> 10425.00 RPM  
sensorNormalMin.42:--> 10425.00 RPM  
sensorNormalMin.43:--> 10425.00 RPM  
sensorNormalMin.44:--> 10425.00 RPM  
sensorNormalMin.45:--> 10425.00 RPM  
sensorNormalMin.46:-->  
sensorNormalMin.47:-->  
sensorNormalMin.48:-->  

Utilizing SNMP Capabilities of EMC Disk Library  
A Detailed Review 38
sensorNormalMin.49:-->50.00 Degrees C
sensorNormalMin.50:-->50.00 Degrees C
sensorNormalMin.51:-->50.00 Degrees C
sensorNormalMin.52:-->50.00 Degrees C
sensorNormalMin.53:-->50.00 Degrees C
sensorNormalMin.54:-->50.00 Degrees C
sensorNormalMin.55:-->50.00 Degrees C
sensorNormalMin.56:-->50.00 Degrees C
sensorNormalReading.1:-->50.00 Degrees C
sensorNormalReading.2:-->50.00 Degrees C
sensorNormalReading.3:-->23.00 Degrees C
sensorNormalReading.4:-->40.00 Degrees C
sensorNormalReading.5:-->40.00 Degrees C
sensorNormalReading.6:-->40.00 Degrees C
sensorNormalReading.7:-->40.00 Degrees C
sensorNormalReading.8:-->50.00 Degrees C
sensorNormalReading.9:-->50.00 Degrees C
sensorNormalReading.10:-->23.00 Degrees C
sensorNormalReading.11:-->40.00 Degrees C
sensorNormalReading.12:-->40.00 Degrees C
sensorNormalReading.13:-->40.00 Degrees C
sensorNormalReading.14:-->40.00 Degrees C
sensorNormalReading.15:-->3.10 Volts
sensorNormalReading.16:-->3.10 Volts
sensorNormalReading.17:-->3.10 Volts
sensorNormalReading.18:-->3.10 Volts
sensorNormalReading.19:-->3.10 Volts
sensorNormalReading.20:-->3.10 Volts
sensorNormalReading.21:-->3.10 Volts
sensorNormalReading.22:-->3.10 Volts
sensorNormalReading.23:-->3.10 Volts
sensorNormalReading.24:-->3.10 Volts
sensorNormalReading.25:-->3.10 Volts
sensorNormalReading.26:-->3.10 Volts
sensorNormalReading.27:-->3.10 Volts
sensorNormalReading.28:-->3.10 Volts
sensorNormalReading.29:-->3.10 Volts
sensorNormalReading.30:-->3.10 Volts
sensorNormalReading.31:-->3.10 Volts
sensorNormalReading.32:-->1800.00 RPM
sensorNormalReading.33:-->1800.00 RPM
sensorNormalReading.34:-->1800.00 RPM
sensorNormalReading.35:-->1800.00 RPM
sensorNormalReading.36:-->1800.00 RPM
sensorNormalReading.37:-->1800.00 RPM
sensorNormalReading.38:-->1800.00 RPM
sensorNormalReading.39:-->1800.00 RPM
sensorNormalReading.40:-->1800.00 RPM
sensorNormalReading.41:-->1800.00 RPM
sensorNormalReading.42:-->1800.00 RPM
sensorNormalReading.43:-->1800.00 RPM
sensorNormalReading.44:-->1800.00 RPM
sensorNormalReading.45:-->1800.00 RPM
sensorNormalReading.46:-->1800.00 RPM
sensorNormalReading.47:-->1800.00 RPM
sensorNormalReading.48:-->1800.00 RPM
sensorNormalReading.49:-->1800.00 RPM

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
sensorNormalReading.50:--> 
sensorNormalReading.51:--> 
sensorNormalReading.52:--> 
sensorNormalReading.53:--> 
sensorNormalReading.54:--> 
sensorNormalReading.55:--> 
sensorNormalReading.56:--> 
sensorNormalMax.1:-->69.00 Degrees C
sensorNormalMax.2:-->69.00 Degrees C
sensorNormalMax.3:-->69.00 Degrees C
sensorNormalMax.4:-->69.00 Degrees C
sensorNormalMax.5:-->69.00 Degrees C
sensorNormalMax.6:-->69.00 Degrees C
sensorNormalMax.7:-->69.00 Degrees C
sensorNormalMax.8:-->69.00 Degrees C
sensorNormalMax.9:-->69.00 Degrees C
sensorNormalMax.10:-->69.00 Degrees C
sensorNormalMax.11:-->69.00 Degrees C
sensorNormalMax.12:-->69.00 Degrees C
sensorNormalMax.13:-->69.00 Degrees C
sensorNormalMax.14:-->69.00 Degrees C
sensorNormalMax.15:-->2.54 Volts
sensorNormalMax.16:--> 
sensorNormalMax.17:--> 
sensorNormalMax.18:--> 
sensorNormalMax.19:--> 
sensorNormalMax.20:--> 
sensorNormalMax.21:--> 
sensorNormalMax.22:--> 
sensorNormalMax.23:--> 
sensorNormalMax.24:--> 
sensorNormalMax.25:--> 
sensorNormalMax.26:-->2.54 Volts
sensorNormalMax.27:--> 
sensorNormalMax.28:--> 
sensorNormalMax.29:--> 
sensorNormalMax.30:--> 
sensorNormalMax.31:--> 
sensorNormalMax.32:-->14775.00 RPM
sensorNormalMax.33:-->14775.00 RPM
sensorNormalMax.34:-->14775.00 RPM
sensorNormalMax.35:-->14775.00 RPM
sensorNormalMax.36:-->14775.00 RPM
sensorNormalMax.37:-->14775.00 RPM
sensorNormalMax.38:-->14775.00 RPM
sensorNormalMax.39:-->14775.00 RPM
sensorNormalMax.40:-->14775.00 RPM
sensorNormalMax.41:-->14775.00 RPM
sensorNormalMax.42:-->14775.00 RPM
sensorNormalMax.43:-->14775.00 RPM
sensorNormalMax.44:-->14775.00 RPM
sensorNormalMax.45:-->14775.00 RPM
sensorNormalMax.46:--> 
sensorNormalMax.47:--> 
sensorNormalMax.48:--> 
sensorNormalMax.49:--> 
sensorNormalMax.50:--> 

Utilizing SNMP Capabilities of EMC Disk Library
A Detailed Review
EDL hardware monitoring: EDL 1.2 hardware

Following is the entire tree of server hardware objects for EDL 1.2 server hardware. As shown, several server hardware items can be monitored, such as power supply voltages, fan speeds, CPU temperature, and so on. You need to compile other MIBs into your SNMP management software, SUPERMICRO-HEALTH-MIB.txt and SUPERMICRO-SMI.txt.

```plaintext
smHealthMonitorName.1:-->CPU1 Fan Speed
smHealthMonitorName.2:-->CPU2 Fan Speed
smHealthMonitorName.3:-->Chassis Fan 1 Speed
smHealthMonitorName.4:-->Chassis Fan 2 Speed
smHealthMonitorName.5:-->Chassis Fan 3 Speed
smHealthMonitorName.6:-->Chassis Fan 4 Speed
smHealthMonitorName.7:-->+12V Voltage
smHealthMonitorName.8:-->+5V Voltage
smHealthMonitorName.9:-->+3.3V Voltage
smHealthMonitorName.10:-->3.3Vsb Voltage
smHealthMonitorName.11:-->Vccp Voltage
smHealthMonitorName.12:-->-12V Voltage
smHealthMonitorName.13:-->CPU1 Temperature
smHealthMonitorName.14:-->CPU2 Temperature
smHealthMonitorName.15:-->System Temperature
smHealthMonitorName.16:-->Chassis Intrusion
smHealthMonitorName.17:-->Power Supply Failure
```

```plaintext
smHealthMonitorType.1:-->0
smHealthMonitorType.2:-->0
smHealthMonitorType.3:-->0
smHealthMonitorType.4:-->0
smHealthMonitorType.5:-->0
smHealthMonitorType.6:-->0
smHealthMonitorType.7:-->1
smHealthMonitorType.8:-->1
smHealthMonitorType.9:-->1
smHealthMonitorType.10:-->1
smHealthMonitorType.11:-->1
smHealthMonitorType.12:-->1
smHealthMonitorType.13:-->2
smHealthMonitorType.14:-->2
smHealthMonitorType.15:-->2
smHealthMonitorType.16:-->3
smHealthMonitorType.17:-->3
```

```plaintext
smHealthMonitorReading.1:-->5273
smHealthMonitorReading.2:-->4963
smHealthMonitorReading.3:-->3422
smHealthMonitorReading.4:-->3538
smHealthMonitorReading.5:-->3499
smHealthMonitorReading.6:-->3461
smHealthMonitorReading.7:-->11916
smHealthMonitorReading.8:-->4945
```
smHealthMonitorReading.9:-->3296
smHealthMonitorReading.10:-->3312
smHealthMonitorReading.11:-->1440
smHealthMonitorReading.12:-->-12130
smHealthMonitorReading.13:-->31
smHealthMonitorReading.14:-->30
smHealthMonitorReading.15:-->42
smHealthMonitorReading.16:-->0
smHealthMonitorReading.17:-->0
smHealthMonitorHighLimit.7:-->13740
smHealthMonitorHighLimit.8:-->5752
smHealthMonitorHighLimit.9:-->3792
smHealthMonitorHighLimit.10:-->3792
smHealthMonitorHighLimit.11:-->1856
smHealthMonitorHighLimit.12:-->10185
smHealthMonitorHighLimit.13:-->78
smHealthMonitorHighLimit.14:-->78
smHealthMonitorHighLimit.15:-->65
smHealthMonitorLowLimit.1:-->1430
smHealthMonitorLowLimit.2:-->1430
smHealthMonitorLowLimit.3:-->88
smHealthMonitorLowLimit.4:-->88
smHealthMonitorLowLimit.5:-->88
smHealthMonitorLowLimit.6:-->88
smHealthMonitorLowLimit.7:-->10153
smHealthMonitorLowLimit.8:-->4247
smHealthMonitorLowLimit.9:-->2800
smHealthMonitorLowLimit.10:-->2800
smHealthMonitorLowLimit.11:-->768
smHealthMonitorLowLimit.12:-->13821
smHealthMonitorMaxReading.1:-->168750
smHealthMonitorMaxReading.2:-->168750
smHealthMonitorMaxReading.3:-->21093
smHealthMonitorMaxReading.4:-->21093
smHealthMonitorMaxReading.5:-->21093
smHealthMonitorMaxReading.6:-->21093
smHealthMonitorMaxReading.7:-->15504
smHealthMonitorMaxReading.8:-->6854
smHealthMonitorMaxReading.9:-->4080
smHealthMonitorMaxReading.10:-->4080
smHealthMonitorMaxReading.11:-->4080
smHealthMonitorMaxReading.12:-->6137
smHealthMonitorMaxReading.13:-->127
smHealthMonitorMaxReading.14:-->127
smHealthMonitorMaxReading.15:-->127
smHealthMonitorMaxReading.16:-->1
smHealthMonitorMaxReading.17:-->1
smHealthMonitorMinReading.1:-->0
smHealthMonitorMinReading.2:-->0
smHealthMonitorMinReading.3:-->0
smHealthMonitorMinReading.4:-->0
smHealthMonitorMinReading.5:-->0
smHealthMonitorMinReading.6:-->0
smHealthMonitorMinReading.7:-->0
smHealthMonitorMinReading.8:-->0
smHealthMonitorMinReading.9:-->0
smHealthMonitorMinReading.10:-->0
Discovering EDL servers

To determine that a particular device that responds to SNMP RFC1213 reads is an EDL server, read the object sysDesc object, which returns “EMC CLARiiON Disk Library”.

Active Engine Failover configurations

EDL models with Active Engine Failover have two EDL servers, each of which responds separately to SNMP get (read) requests. Similarly, each can be configured to send SNMP traps. SNMP management software must therefore be able to receive traps from either IP address. (And the trap will appear to come from the “service” IP address.)

Conclusion

SNMP is the standard means of monitoring networked devices in large enterprises. Its ease of use and extensive set of tools and standards ensures that it will be increasingly adopted. As more and more EMC Disk Libraries are deployed in enterprise environments, we will see greater interest in our capability to use SNMP as a management tool.