This chapter explains Simple Network Management Protocol (SNMP) as implemented by the Cisco ONS 15454.

For SNMP setup information, refer to the *Cisco ONS 15454 Procedure Guide*.

Chapter topics include:

- SNMP Overview
- Basic SNMP Components
- SNMP External Interface Requirement
- SNMP Version Support
- SNMP Message Types
- SNMP Management Information Bases
- SNMP Trap Content
- SNMP Community Names
- Proxy Over Firewalls
- Remote Monitoring

Contents

- 1 SNMP Overview
  - 1.1 Figure 16-1: Basic Network Managed by SNMP
- 2 Basic SNMP Components
  - 2.1 Figure 16-2: Example of the Primary SNMP Components
  - 2.2 Figure 16-3: Agent Gathering Data from a MIB and Sending Traps to the Manager
- 3 SNMP External Interface Requirement
- 4 SNMP Version Support
- 5 SNMP Message Types
  - 5.1 Table 16-1: ONS 15454 SNMP Message Types
- 6 SNMP Management Information Bases
  - 6.1 IETF-Standard MIBs for the ONS 15454
    - 6.1.1 Table 16-2: IETF Standard MIBs Implemented in the ONS 15454 System
  - 6.2 Proprietary ONS 15454 MIBs
    - 6.2.1 Table 16-3: ONS 15454 Proprietary MIBs
  - 6.3 Generic Threshold and Performance Monitoring MIBs
    - 6.3.1 Table 16-4: cerentGenericPmThresholdTable
    - 6.3.2 Table 16-5: 32-Bit cerentGenericPmStatsCurrentTable
    - 6.3.3 Table 16-6: 32-Bit cerentGenericPmStatsIntervalTable
- 7 SNMP Trap Content
  - 7.1 Generic and IETF Traps
    - 7.1.1 Table 16-7: Supported Generic IETF Traps
  - 7.2 Variable Trap Bindings
    - 7.2.1 Table 16-8: Supported ONS 15454 SNMPv2 Trap Variable Bindings
- 8 SNMP Community Names
- 9 Proxy Over Firewalls
- 10 Remote Monitoring
  - 10.1 64-Bit RMON Monitoring over DCC
    - 10.1.1 Row Creation in MediaIndependentTable
    - 10.1.2 Row Creation in cMediaIndependentHistoryControlTable

- 10.2 HC-RMON-MIB Support
- 10.3 Ethernet Statistics RMON Group
  - 10.3.1 Row Creation in etherStatsTable
  - 10.3.2 Get Requests and GetNext Requests
  - 10.3.3 Row Deletion in etherStatsTable
  - 10.3.4 64-Bit etherStatsHighCapacityTable
- 10.4 History Control RMON Group
  - 10.4.1 History Control Table
    - 10.4.1.1 Table 16-9: RMON History Control Periods and History Categories
  - 10.4.2 Row Creation in historyControlTable
  - 10.4.3 Get Requests and GetNext Requests
  - 10.4.4 Row Deletion in historyControl Table
- 10.5 Ethernet History RMON Group
  - 10.5.1 64-Bit etherHistoryHighCapacityTable
- 10.6 Alarm RMON Group
  - 10.6.1 Alarm Table
  - 10.6.2 Row Creation in alarmTable
    - 10.6.2.1 Table 16-10: OIDs Supported in the AlarmTable
  - 10.6.3 Get Requests and GetNext Requests
  - 10.6.4 Row Deletion in alarmTable
- 10.7 Event RMON Group
  - 10.7.1 Event Table
  - 10.7.2 Log Table

SNMP Overview

SNMP is an application-layer communication protocol that allows ONS 15454 network devices to exchange management information among these systems and with other devices outside the network. Through SNMP, network administrators can manage network performance, find and solve network problems, and plan network growth. Up to 10 SNMP trap destinations and five concurrent Cisco Transport Controller (CTC) user sessions are allowed per node.

The ONS 15454 uses SNMP for asynchronous event notification to a network management system (NMS). Cisco ONS system SNMP implementation uses standard Internet Engineering Task Force (IETF) management information bases (MIBs) to convey node-level inventory, fault, and performance management information for generic DS-1, DS-3, SONET, and Ethernet read-only management. SNMP allows a generic SNMP manager such as HP OpenView Network Node Manager (NNM) or Open Systems Interconnection (OSI) NetExpert to be utilized for limited management functions.

The Cisco ONS 15454 supports SNMP Version 1 (SNMPv1) and SNMP Version 2c (SNMPv2c). These versions share many features, but SNMPv2c includes additional protocol operations and 64-bit performance monitoring support. This chapter describes both versions and gives SNMP configuration parameters for the ONS 15454.

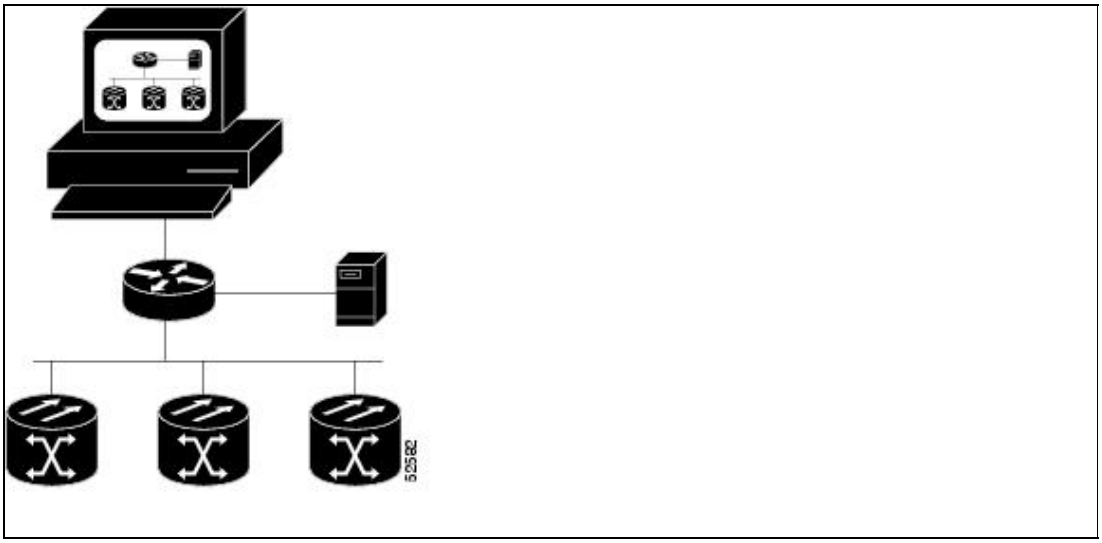
Note: In Software Release 8.0 and later, you can retrieve automatic in service (AINS) state and soak time through the SNMP and Transaction Language One (TL1) interfaces.

Note: The CERENT-MSDWDM-MIB.mib, CERENT-FC-MIB.mib, and CERENT-GENERIC-PM-MIB.mib in the CiscoV2 directory support 64-bit performance monitoring counters. The SNMPv1 MIB in the CiscoV1 directory does not contain 64-bit performance monitoring counters, but supports the lower and higher word values of the corresponding 64-bit counter. The other MIB files in the CiscoV1 and CiscoV2 directories are

identical in content and differ only in format.

Figure 16-1 illustrates the basic layout idea of an SNMP-managed network.

Figure 16-1: Basic Network Managed by SNMP

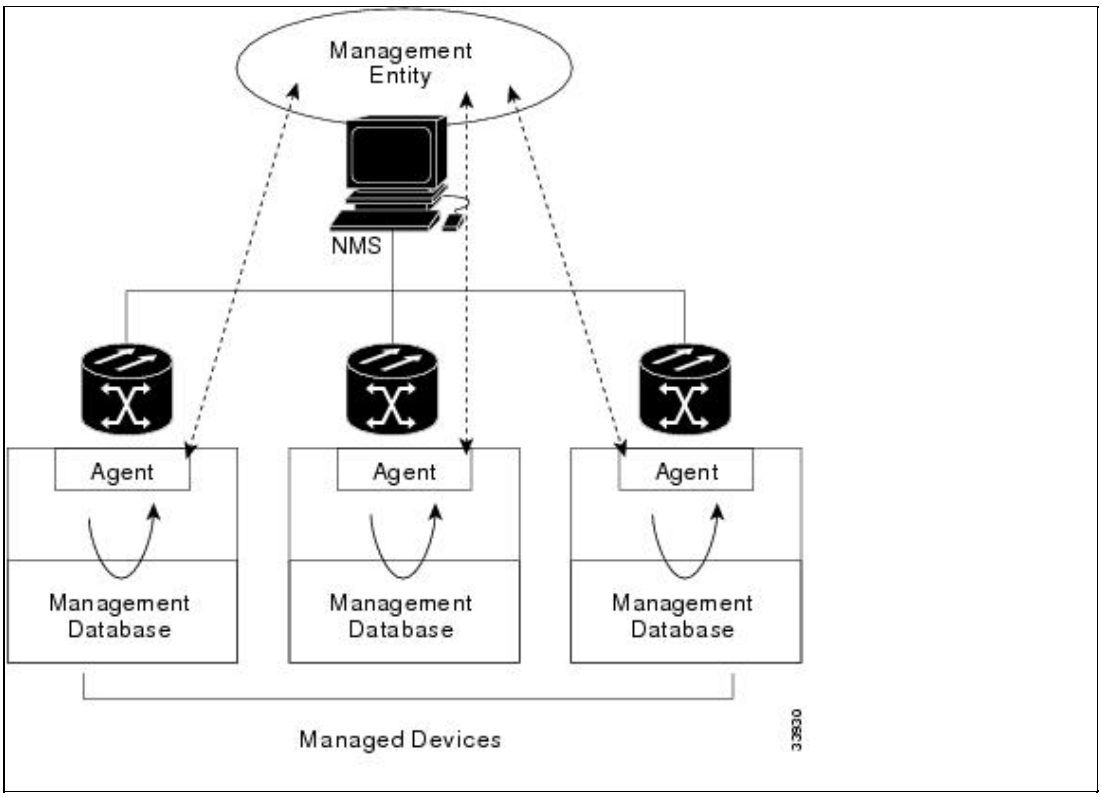


Basic SNMP Components

In general terms, an SNMP-managed network consists of a management system, agents, and managed devices.

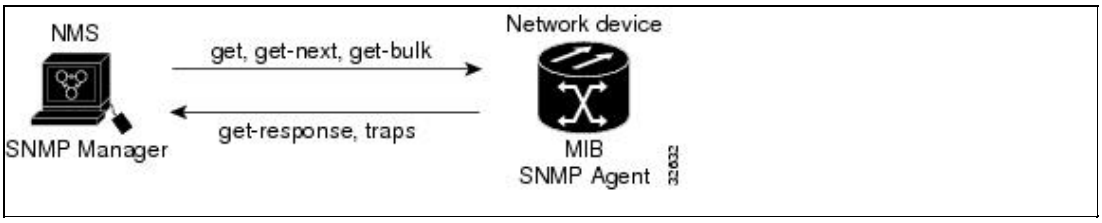
A management system such as HP OpenView executes monitoring applications and controls managed devices. Management systems execute most of the management processes and provide the bulk of memory resources used for network management. Additionally, a network might be managed by one or several management systems. Figure 16-2 illustrates the relationship between the network manager, the SNMP agent, and the managed devices.

Figure 16-2: Example of the Primary SNMP Components



An agent (such as SNMP) residing on each managed device translates local management information data-such as performance information or event and error information caught in software traps-into a readable form for the management system. Figure 16-3 illustrates SNMP agent get-requests that transport data to the network management software.

Figure 16-3: Agent Gathering Data from a MIB and Sending Traps to the Manager



The SNMP agent captures data from MIBs, which are device parameter and network data repositories, or from error or change traps.

A managed element-such as a router, access server, switch, bridge, hub, computer host, or network element (such as an ONS 15454)-is accessed through the SNMP agent. Managed devices collect and store management information, making it available through SNMP to other management systems having the same protocol compatibility.

SNMP External Interface Requirement

Since all SNMP requests come from a third-party application, the only external interface requirement is that a third-party SNMP client application should have the ability to upload RFC 3273 SNMP MIB variables in the etherStatsHighCapacityTable, etherHistoryHighCapacityTable, or mediaIndependentTable.

Adding a new para.

SNMP Version Support

The ONS 15454 supports SNMPv1 and SNMPv2c traps and get requests. The ONS 15454 SNMP MIBs define alarms, traps, and status. Through SNMP, NMS applications can query a management agent for data from functional entities such as Ethernet switches and SONET multiplexers using a supported MIB.

Note: ONS 15454 MIB files in the CiscoV1 and CiscoV2 directories are almost identical in content except for the difference in 64-bit performance monitoring features. The CiscoV2 directory contains three MIBs with 64-bit performance monitoring counters:. CERENT-MSDWDM-MIB.mib, CERENT-FC-MIB.mib, and CERENT-GENERIC-PM-MIB.mib The CiscoV1 directory does not contain any 64-bit counters, but it does support the lower and higher word values used in 64-bit counters. The two directories also have somewhat different formats.

SNMP Message Types

The ONS 15454 SNMP agent communicates with an SNMP management application using SNMP messages. Table 16-1 describes these messages.

Table 16-1: ONS 15454 SNMP Message Types

Operation	Description
get-request	Retrieves a value from a specific variable.
get-next-request	Retrieves the value following the named variable; this operation is often used to retrieve variables from within a table. With this operation, an SNMP manager does not need to know the exact variable name. The SNMP manager searches sequentially to find the needed variable from within the MIB.
get-response	Replies to a get-request, get-next-request, get-bulk-request, or set-request sent by an NMS.
get-bulk-request	Fills the get-response with up to the max-repetition number of get-next interactions, similar to a get-next-request.
set-request	Provides remote network monitoring (RMON) MIB.
trap	Indicates that an event has occurred. An unsolicited message is sent by an SNMP agent to an SNMP manager.

SNMP Management Information Bases

A managed object, sometimes called a MIB object, is one of many specific characteristics of a managed device. The MIB consists of hierarchically organized object instances (variables) that are accessed by network-management protocols such as SNMP. Section 16.6.1 lists the IETF standard MIBs implemented in the ONS 15454 SNMP agent. Section 16.6.2 lists the proprietary MIBs implemented in the ONS 15454.

IETF-Standard MIBs for the ONS 15454

Table 16-2 lists the IETF-standard MIBs implemented in the ONS 15454 SNMP agents.

You must first compile the MIBs in Table 16-2. Compile the Table 16-3 MIBs next.

Caution! If you do not compile MIBs in the correct order, one or more might not compile correctly.

Table 16-2: IETF Standard MIBs Implemented in the ONS 15454 System

RFC[1] Number	Module Name	Title/Comments
-	IANAifType-MIB.mib	Internet Assigned Numbers Authority (IANA) ifType
1213	RFC1213-MIB-rfc1213.mib	Management Information Base for Network
1907	SNMPV2-MIB-rfc1907.mib	Management of TCP/IP-based Internets: MIB-II Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)
1253	RFC1253-MIB-rfc1253.mib	OSPF Version 2 Management Information Base
1493	BRIDGE-MIB-rfc1493.mib	Definitions of Managed Objects for Bridges (This defines MIB objects for managing MAC bridges based on the IEEE 802.1D-1990 standard between Local Area Network [LAN] segments.)
2819	RMON-MIB-rfc2819.mib	Remote Network Monitoring Management Information Base
2737	ENTITY-MIB-rfc2737.mib	Entity MIB (Version 2)
2233	IF-MIB-rfc2233.mib	Interfaces Group MIB using SNMPv2
2358	EtherLike-MIB-rfc2358.mib	Definitions of Managed Objects for the Ethernet-like Interface Types
2493	PerfHist-TC-MIB-rfc2493.mib	Textual Conventions for MIB Modules Using Performance History Based on 15 Minute Intervals
2495	DS1-MIB-rfc2495.mib	Definitions of Managed Objects for the DS1, E1, DS2 and E2 Interface Types
2496	DS3-MIB-rfc2496.mib	Definitions of Managed Object for the DS3/E3 Interface Type
2558	SONET-MIB-rfc2558.mib	Definitions of Managed Objects for the SONET/SDH Interface Type
2674	P-BRIDGE-MIB-rfc2674.mib Q-BRIDGE-MIB-rfc2674.mib	Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering and Virtual LAN Extensions
3273	HC-RMON-MIB	The MIB module for managing remote monitoring device implementations, augmenting the original RMON MIB as specified in RFC 2819 and RFC 1513 and RMON-2 MIB as specified in RFC 2021

1. RFC = Request for Comment

Proprietary ONS 15454 MIBs

Each ONS 15454 is shipped with a software CD containing applicable proprietary MIBs. Table 16-3 lists the proprietary MIBs for the ONS 15454.

Table 16-3: ONS 15454 Proprietary MIBs

MIB Number	Module Name
1	CERENT-GLOBAL-REGISTRY.mib
2	CERENT-TC.mib
3	CERENT-454.mib
4	CERENT-GENERIC.mib (not applicable to ONS 15454)
5	CISCO-SMI.mib

6	CISCO-VOA-MIB.mib
7	CERENT-MSDWDM-MIB.mib
8	CERENT-OPTICAL-MONITOR-MIB.mib
9	CERENT-HC-RMON-MIB.mib
10	CERENT-ENVMON-MIB.mib
11	CERENT-GENERIC-PM-MIB.mib

Note: If you cannot compile the proprietary MIBs correctly, log into the Technical Support Website at http://www.cisco.com/techsupport or call Cisco TAC (800) 553-2447.

Note: When SNMP indicates that a muxponder (MXP) or transponder (TXP) wavelength is unknown, it means that the corresponding card (MXP_2.5G_10E, TXP_MR_10E, MXP_2.5G_10G, TXP_MR_10G, TXP_MR_2.5G, or TXPP_MR_2.5G) works with the first tunable wavelength. For more information about MXP and TXP cards, refer to the *Cisco ONS 15454 DWDM Reference Manual*.

Generic Threshold and Performance Monitoring MIBs

A MIB called CERENT-GENERIC-PM-MIB allows network management stations (NMS) to use a single, generic MIB for accessing threshold and performance monitoring data of different interface types. The MIB is generic in the sense that it is not tied to any particular kind of interface. The MIB objects can be used to obtain threshold values, current performance monitoring (PM) counts, and historic PM statistics for each kind of monitor and any supported interval at the near end and far end.

Previously existing MIBs in the ONS 15454 system provide some of these counts. For example, SONET interface 15-minute current PM counts and historic PM statistics are available using the SONET-MIB. DS-1 and DS-3 counts and statistics are available through the DS1-MIB and DS-3 MIB respectively. The generic MIB provides these types of information and also fetches threshold values and single-day statistics. In addition, the MIB supports optics and dense wavelength division multiplexing (DWDM) threshold and performance monitoring information.

The CERENT-GENERIC-PM-MIB is organized into three different tables:

- cerentGenericPmThresholdTable
- cerentGenericPmStatsCurrentTable
- cerentGenericPmStatsIntervalTable
- The cerentGenericPmThresholdTable is used to obtain the threshold values for the monitor types. It is indexed based on the following items:
- Interface index (cerentGenericPmThresholdIndex)
- Monitor type (cerentGenericPmThresholdMonType). The syntax of cerentGenericPmThresholdMonType is type cerentMonitorType, defined in CERENT-TC.mib.
- Location (cerentGenericPmThresholdLocation). The syntax of cerentGenericPmThresholdLocation is type cerentLocation, defined in CERENT-TC.mib.
- Time period (cerentGenericPmThresholdPeriod). The syntax of cerentGenericPmThresholdPeriod is type cerentPeriod, defined in CERENT-TC.mib.

Threshold values can be provided in 64-bit and 32-bit formats. (For more information about 64-bit counters, see the HC-RMON-MIB Support.) The 64-bit values in cerentGenericPmThresholdHCValue can be used with agents that support SNMPv2. The two 32-bit values (cerentGenericPmThresholdValue and cerentGenericPmThresholdOverFlowValue) can be used by NMSs that only support SNMPv1. The objects compiled in the cerentGenericPmThresholdTable are shown in Table 16-4.

Table 16-4: cerentGenericPmThresholdTable

Index Objects	Information Objects
cerentGenericPmThresholdIndex	cerentGenericPmThresholdValue
cerentGenericPmThresholdMonType	cerentGenericPmThresholdOverFlowValue
cerentGenericPmThresholdLocation	cerentGenericPmThresholdHCValue
cerentGenericPmThresholdPeriod	-

The second table within the MIB, cerentGenericPmStatsCurrentTable, compiles the current performance monitoring (PM) values for the monitor types. The table is indexed based on interface index (cerentGenericPmStatsCurrentIndex), monitor type (cerentGenericPmStatsCurrentMonType), location (cerentGenericPmStatsCurrentLocation) and time period (cerentGenericPmStatsCurrentPeriod). The syntax of cerentGenericPmStatsCurrentIndex is type cerentLocation, defined in CERENT-TC.mib. The syntax of cerentGenericPmStatsCurrentMonType is type cerentMonitor, defined in CERENT-TC.mib. The syntax of cerentGenericPmStatsCurrentPeriod is type cerentPeriod, defined in CERENT-TC.mib.

The cerentGenericPmStatsCurrentTable validates the current PM value using the cerentGenericPmStatsCurrentValid object and registers the number of valid intervals with historical PM statistics in the cerentGenericPmStatsCurrentValidIntervals object.

PM values are provided in 64-bit and 32-bit formats. The 64-bit values in cerentGenericPmStatsCurrentHCValue can be used with agents that support SNMPv2. The two 32-bit values (cerentGenericPmStatsCurrentValue and cerentGenericPmStatsCurrentOverFlowValue) can be used by NMS that only support SNMPv1. The cerentGenericPmStatsCurrentTable is shown in Table 16-5.

Table 16-5: 32-Bit cerentGenericPmStatsCurrentTable

Index Objects	Informational Objects
cerentGenericPmStatsCurrentIndex	cerentGenericPmStatsCurrentValue
cerentGenericPmStatsCurrentMonType	cerentGenericPmStatsCurrentOverFlowValue
cerentGenericPmStatsCurrentLocation	cerentGenericPmStatsCurrentHCValue
cerentGenericPmStatsCurrentPeriod	cerentGenericPmStatsCurrentValidData
-	cerentGenericPmStatsCurrentValidIntervals

The third table in the MIB, cerentGenericPmStatsIntervalTable, obtains historic PM values for the monitor types. It validates the current PM value in the cerentGenericPmStatsIntervalValid object. This table is indexed based on interface index (cerentGenericPmStatsIntervalIndex), monitor type (cerentGenericPMStatsIntervalMonType), location (cerentGenericPmStatsIntervalLocation), and period (cerentGenericPmStatsIntervalPeriod). The syntax of cerentGenericPmStatsIntervalIndex is type cerentLocation, defined in CERENT-TC.mib. The syntax of cerentGenericPmStatsIntervalMonType is type cerentMonitor, defined in CERENT-TC.mib. The syntax of cerentGernicPmStatsIntervalPeriod is type cerentPeriod, defined in CERENT-TC.mib.

The table provides historic PM values in 64-bit and 32-bit formats. The 64-bit values contained in the cerentGenericPmStatsIntervalHCValue table can be used with SNMPv2 agents. The two 32-bit values (cerentGenericPmStatsIntervalValue and cerentGenericPmStatsIntervalOverFlowValue) can be used by SNMPv1 NMS. The cerentGenericPmStatsIntervalTable is shown in Table 16-6.

Table 16-6: 32-Bit cerentGenericPmStatsIntervalTable

Index Objects	Informational Objects
cerentGenericPmStatsIntervalIndex	cerentGenericPmStatsIntervalValue

cerentGenericPmStatsIntervalMonType	cerentGenericPmStatsIntervalOverFlowValue
cerentGenericPmStatsIntervalLocation	cerentGenericPmStatsIntervalHCValue
cerentGenericPmStatsIntervalPeriod	cerentGenericPmStatsIntervalValidData
cerentGenericPmStatsIntervalNumber	-

SNMP Trap Content

The ONS 15454 uses SNMP traps to generate all alarms and events, such as raises and clears. The traps contain the following information:

- Object IDs that uniquely identify each event with information about the generating entity (the slot or port; synchronous transport signal [STS] and Virtual Tributary [VT]; bidirectional line switched ring [BLSR], Spanning Tree Protocol [STP], etc.).
- Severity and service effect of the alarm (critical, major, minor, or event; service-affecting or non-service-affecting).
- Date and time stamp showing when the alarm occurred.

Generic and IETF Traps

The ONS 15454 supports the generic IETF traps listed in Table 16-7.

Table 16-7: Supported Generic IETF Traps

Trap	From RFC No. MIB	Description
coldStart	RFC1907-MIB	Agent up, cold start.
warmStart	RFC1907-MIB	Agent up, warm start.
authenticationFailure	RFC1907-MIB	Community string does not match.
newRoot	RFC1493/ BRIDGE-MIB	Sending agent is the new root of the spanning tree.
topologyChange	RFC1493/ BRIDGE-MIB	A port in a bridge has changed from Learning to Forwarding or Forwarding to Blocking.
entConfigChange	RFC2737/ ENTITY-MIB	The entLastChangeTime value has changed.
dsx1LineStatusChange	RFC2495/ DS1-MIB	The value of an instance of dsx1LineStatus has changed. The trap can be used by an NMS to trigger polls. When the line status change results from a higher-level line status change (for example, a DS-3), no traps for the DS-1 are sent.
dsx3LineStatusChange	RFC2496/ DS3-MIB	The value of an instance of dsx3LineStatus has changed. This trap can be used by an NMS to trigger polls. When the line status change results in a lower-level line status change (for example, a DS-1), no traps for the lower-level are sent.
risingAlarm	RFC2819/ RMON-MIB	The SNMP trap that is generated when an alarm entry crosses the rising threshold and the entry generates an event that is configured for sending SNMP traps.
fallingAlarm	RFC2819/ RMON-MIB	The SNMP trap that is generated when an alarm entry crosses the falling threshold and the entry generates an event that is configured for sending SNMP traps.

Variable Trap Bindings

Each SNMP trap contains variable bindings that are used to create the MIB tables. ONS 15454 traps and variable bindings are listed in Table 16-8. For each group (such as Group A), all traps within the group are associated with all of its variable bindings.

Table 16-8: Supported ONS 15454 SNMPv2 Trap Variable Bindings

Group	Trap Name(s) Associated with	Variable Binding Number	SNMPv2 Variable Bindings	Description
A	dsx1LineStatusChange (from RFC 2495.)	(1)	dsx1LineStatus	This variable indicates the line status of the interface. It contains loopback, failure, received alarm and transmitted alarm information.
		(2)	dsx1LineStatusLastChange	The value of MIB II's sysUpTime object at the time this DS1 entered its current line status state. If the current state was entered prior to the last proxy-agent reinitialization, the value of this object is zero.
		(3)	cerent454NodeTime	The time that an event occurred.
		(4)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(5)	snmpTrapAddress	The address of the SNMP trap.
B	dsx3LineStatusChange (from	(1)	dsx3LineStatus	This variable indicates the line status of the interface. It contains loopback state information and

				failure state information.
		(2)	dsx3LineStatusLastChange	The value of MIB II's sysUpTime object at the time this DS3/E3 entered its current line status state. If the current state was entered prior to the last reinitialization of the proxy-agent, then the value is zero.
		(3)	cerent454NodeTime	The time that an event occurred.
		(4)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(5)	snmpTrapAddress	The address of the SNMP trap.
C	coldStart (from RFC 1907)	(1)	cerent454NodeTime	The time that the event occurred.
	warmStart (from RFC 1907)	(2)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
	newRoot (from RFC)	(3)	snmpTrapAddress	The address of the SNMP trap.
	topologyChange (from RFC)		-	-
	entConfigChange (from RFC 2737)		-	-
	authenticationFailure (from RFC 1907)		-	-
		(1)	alarmIndex	This variable uniquely identifies each entry in the alarm table. When

				an alarm in the table clears, the alarm indexes change for each alarm listed.
		(2)	alarmVariable	The object identifier of the variable being sampled.
		(3)	alarmSampleType	The method of sampling the selected variable and calculating the value to be compared against the thresholds.
		(4)	alarmValue	The value of the statistic during the last sampling period.
		(5)	alarmRisingThreshold	When the current sampled value is greater than or equal to this threshold, and the value at the last sampling interval was less than this threshold, a single event is generated. A single event is also generated if the first sample after this entry is greater than or equal to this threshold.
		(6)	cerent454NodeTime	The time that an event occurred.
		(7)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(8)	snmpTrapAddress	The address of the SNMP trap.
		(1)	alarmIndex	This variable uniquely identifies each entry in the alarm table. When an alarm in the table

				clears, the alarm indexes change for each alarm listed.
		(2)	alarmVariable	The object identifier of the variable being sampled.
		(3)	alarmSampleType	The method of sampling the selected variable and calculating the value to be compared against the thresholds.
		(4)	alarmValue	The value of the statistic during the last sampling period.
		(5)	alarmFallingThreshold	When the current sampled value is less than or equal to this threshold, and the value at the last sampling interval was greater than this threshold, a single event is generated. A single is also generated if the first sample after this entry is less than or equal to this threshold.
		(6)	cerent454NodeTime	The time that an event occurred.
		(7)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(8)	snmpTrapAddress	The address of the SNMP trap.
		(1)	cerent454NodeTime	The time that an event occurred.
		(2)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and

				Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(3)	cerent454AlarmObjectType	The entity that raised the alarm. The NMS should use this value to decide which table to poll for further information about the alarm.
		(4)	cerent454AlarmObjectIndex	Every alarm is raised by an object entry in a specific table. This variable is the index of objects in each table; if the alarm is interface-related, this is the index of the interface in the interface table.
		(5)	cerent454AlarmSlotNumber	The slot of the object that raised the alarm. If a slot is not relevant to the alarm, the slot number is zero.
		(6)	cerent454AlarmPortNumber	The port of the object that raised the alarm. If a port is not relevant to the alarm, the port number is zero.
		(7)	cerent454AlarmLineNumber	The object line that raised the alarm. If a line is not relevant to the alarm, the line number is zero.
		(8)	cerent454AlarmObjectName	The TL1-style user-visible name that uniquely identifies an object in the system.
		(9)	cerent454AlarmAdditionalInfo	Additional information for the alarm object. In the current version of the MIB, this object contains provisioned

				description for alarms that are external to the NE. If there is no additional information, the value is zero.
		(10)	snmpTrapAddress	The address of the SNMP trap.
F	performanceMonitor ThresholdCrossingAlert (from CERENT-454-mib)	(1)	cerent454NodeTime	The time that an event occurred.
		(2)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(3)	cerent454AlarmObjectType	The entity that raised the alarm. The NMS should use this value to decide which table to poll for further information about the alarm.
		(4)	cerent454AlarmObjectIndex	Every alarm is raised by an object entry in a specific table. This variable is the index of objects in each table; if the alarm is interface-related, this is the index of the interface in the interface table.
		(5)	cerent454AlarmSlotNumber	The slot of the object that raised the alarm. If a slot is not relevant to the alarm, the slot number is zero.
		(6)	cerent454AlarmPortNumber	The port of the object that raised the alarm. If a port is not relevant to the alarm, the port number is zero.
		(7)	cerent454AlarmLineNumber	The object line that raised the alarm. If a

				line is not relevant to the alarm, the line number is zero.
		(8)	cerent454AlarmObjectName	The TL1-style user-visible name that uniquely identifies an object in the system.
		(9)	cerent454ThresholdMonitorType	This object indicates the type of metric being monitored.
		(10)	cerent454ThresholdLocation	Indicates whether the event occurred at the near or far end.
		(11)	cerent454ThresholdPeriod	Indicates the sampling interval period.
		(12)	cerent454ThresholdSetValue	The value of this object is the threshold provisioned by the NMS.
		(13)	cerent454ThresholdCurrentValue	-
		(14)	cerent454ThresholdDetectType	-
		(15)	snmpTrapAddress	The address of the SNMP trap.
G	All other traps (from CERENT-454-MIB) not listed above	(1)	cerent454NodeTime	The time that an event occurred.
		(2)	cerent454AlarmState	The alarm severity and service-affecting status. Severities are Minor, Major, and Critical. Service-affecting statuses are Service-Affecting and Non-Service Affecting.
		(3)	cerent454AlarmObjectType	The entity that raised the alarm. The NMS should use this value to decide which table to poll for further information about the alarm.
		(4)	cerent454AlarmObjectIndex	Every alarm is raised by an object entry in a specific table. This variable is the index of objects in each

				table; if the alarm is interface-related, this is the index of the interface in the interface table.
		(5)	cerent454AlarmSlotNumber	The slot of the object that raised the alarm. If a slot is not relevant to the alarm, the slot number is zero.
		(6)	cerent454AlarmPortNumber	The port of the object that raised the alarm. If a port is not relevant to the alarm, the port number is zero.
		(7)	cerent454AlarmLineNumber	The object line that raised the alarm. If a line is not relevant to the alarm, the line number is zero.
		(8)	cerent454AlarmObjectName	The TL1-style user-visible name that uniquely identifies an object in the system.
		(9)	snmpTrapAddress	The address of the SNMP trap.

SNMP Community Names

Community names are used to group SNMP trap destinations. All ONS 15454 trap destinations can be provisioned as part of SNMP communities in CTC. When community names are assigned to traps, the ONS 15454 treats the request as valid if the community name matches one that is provisioned in CTC. In this case, all agent-managed MIB variables are accessible to that request. If the community name does not match the provisioned list, SNMP drops the request.

Proxy Over Firewalls

SNMP and NMS applications have traditionally been unable to cross firewalls used for isolating security risks inside or from outside networks. CTC enables network operations centers (NOCs) to access performance monitoring data such as RMON statistics or autonomous messages across firewalls by using an SNMP proxy element installed on a firewall.

The application-level proxy transports SNMP protocol data units (PDU) between the NMS and NEs, allowing requests and responses between the NMS and NEs and forwarding NE autonomous messages to the NMS. The proxy agent requires little provisioning at the NOC and no additional provisioning at the NEs.

The firewall proxy is intended for use in a gateway network element-end network element (GNE-ENE) topology with many NEs through a single NE gateway. Up to 64 SNMP requests (such as get, getnext, or getbulk) are supported at any time behind single or multiple firewalls. The proxy interoperates with common

NMS such as HP OpenView.

For security reasons, the SNMP proxy feature must be enabled at all receiving and transmitting NEs to function. For instructions to do this, refer to the *Cisco ONS 15454 Procedure Guide*.

Remote Monitoring

The ONS 15454 incorporates RMON to allow network operators to monitor Ethernet card performance and events. The RMON thresholds are user-provisionable in CTC. Refer to the *Cisco ONS 15454 Procedure Guide* for instructions.

Note: Typical RMON operations, other than threshold provisioning, are invisible to the CTC user.

ONS 15454 system RMON is based on the IETF-standard MIB RFC 2819 and includes the following five groups from the standard MIB: Ethernet Statistics, History Control, Ethernet History, Alarm, and Event.

Certain statistics measured on the ML card are mapped to standard MIB if one exists else mapped to a non standard MIB variable. The naming convention used by the standarad/non-standard MIB is not the same as the statistics variable used by the card. Hence when these statistics are obtained via get-reques/get-next-request/SNMP Trap they don't match the name used on the card or as seen by CTC/TL1.

- For ex: STATS_MediaIndStatsRxFramesTooLong stats is mapped to cMediaIndependentInFramesTooLong variable in CERENT MIB. STATS_RxTotalPkts is mapped to mediaIndependentInPkts in HC-RMON-rfc3273.mib

64-Bit RMON Monitoring over DCC

The ONS 15454 DCC is implemented over the IP protocol, which is not compatible with Ethernet. The system builds Ethernet equipment History and Statistics tables using high data level control (HDLC) statistics that are gathered over the data communications channel (DCC) that is running point-to-point protocol (PPP). RMON DCC monitors the health of remote DCC connections for IP and Ethernet.

RMON DCC contains two MIBS for DCC interfaces. They are:

- cMediaIndependentTable-standard, RFC3273; the proprietary extension of the HC-RMON MIB used for reporting statistics
- cMediaIndependentHistoryTable-proprietary MIB used to support history

Row Creation in MediaIndependentTable

The SetRequest PDU contains all needed values to activate a row of the mediaIndependentTable in a single operation as well as assign the status variable to createRequest (2). In order to create the row and status, the SetRequest PDU for entry creation must have a value of zero for each of the object IDs. That is, all object IDs (OIDs) should be of the type OID.0.

In order to create a row, the SetRequest PDU should contain the following:

- mediaIndependentDataSource and its desired value
- mediaIndependentOwner and its desired value (up to 32 characters)
- mediaIndependentStatus with a value of createRequest (2)

The mediaIndependentTable creates a row if the SetRequest PDU is valid according to these rules. The

SNMP agent decides the value of mediaIndependentIndex when the row is created, and a value can change if an Ethernet interface is added or deleted. The values are not sequentially allotted or contiguously numbered. The newly created row will have an mediaIndependentTable value of valid (1). If the row already exists, or if the SetRequest PDU values are insufficient or do not make sense, the SNMP agent returns an error code.

Note: mediaIndependentTable entries are not preserved if the SNMP agent is restarted.

The mediaIndependentTable deletes a row if the SetRequest PDU contains a mediaIndependentStatus with a value of invalid (4). The varbind's OID instance value identifies the row for deletion. You can recreate a deleted row in the table if desired.

Row Creation in cMediaIndependentHistoryControlTable

SNMP row creation and deletion for the cMediaIndependentHistoryControlTable follows the same processes as for the MediaIndependentTable; only the variables differ. In order to create a row, the SetRequest PDU should contain the following:

- cMediaIndependentHistoryControlDataSource and its desired value
- cMediaIndependentHistoryControlOwner and its desired value
- cMediaIndependentHistoryControlStatus with a value of createRequest (2)

HC-RMON-MIB Support

For the ONS 15454, the implementation of the high-capacity remote monitoring information base (HC-RMON-MIB, or RFC 3273) enables 64-bit support of existing RMON tables. This support is provided with the etherStatsHighCapacityTable and the etherHistoryHighCapacityTable. An additional table, the mediaIndependentTable, and an additional object, hcRMONCapabilities, are also added for this support. All of these elements are accessible by any third-party SNMP client should have the ability to upload RFC 3273 SNMP MIB variables in the etherStatsHighCapacityTable, etherHistoryHighCapacityTable, or mediaIndependentTable.

Ethernet Statistics RMON Group

The Ethernet Statistics group contains the basic statistics monitored for each subnetwork in a single table called the etherStatsTable.

Row Creation in etherStatsTable

The SetRequest PDU for creating a row in this table contains all needed values to activate a table row in a single operation as well as assign the status variable to createRequest. The SetRequest PDU OID) entries must have an instance value, or type OID, of 0.

In order to create a row, the SetRequest PDU should contain the following:

- The etherStatsDataSource and its desired value
- The etherStatsOwner and its desired value (up to 32 characters)
- The etherStatsStatus with a value of createRequest (2)

The etherStatsTable creates a row if the SetRequest PDU is valid according to these rules. The SNMP agent decides the value of etherStatsIndex when the row is created and this value changes when an Ethernet interface is added or deleted; it is not sequentially allotted or contiguously numbered. A newly created row will have an etherStatsStatus value of valid (1). If the etherStatsTable row already exists, or if the SetRequest PDU values are insufficient or do not make sense, the SNMP agent returns an error code.

Note: EtherStatsTable entries are not preserved if the SNMP agent is restarted.

Get Requests and GetNext Requests

Get requests and getNext requests for the etherStatsMulticastPkts and etherStatsBroadcastPkts columns return a value of zero because the variables are not supported by ONS 15454 Ethernet cards.

Row Deletion in etherStatsTable

To delete a row in the etherStatsTable, the SetRequest PDU should contain an etherStatsStatus "invalid" value (4). The OID marks the row for deletion. If required, a deleted row can be recreated.

64-Bit etherStatsHighCapacityTable

The Ethernet statistics group contains 64-bit statistics in the etherStatsHighCapacityTable, which provides 64-bit RMON support for the HC-RMON-MIB. The etherStatsHighCapacityTable is an extension of the etherStatsTable that adds 16 new columns for performance monitoring data in 64-bit format. There is a one-to-one relationship between the etherStatsTable and etherStatsHighCapacityTable when rows are created or deleted in either table.

History Control RMON Group

The History Control group defines sampling functions for one or more monitor interfaces in the historyControlTable. The values in this table, as specified in RFC 2819, are derived from the historyControlTable and etherHistoryTable.

History Control Table

The RMON is sampled at one of four possible intervals. Each interval, or period, contains specific history values called buckets. Table 16-9 lists the four sampling periods and corresponding buckets.

The historyControlTable maximum row size is determined by multiplying the number of ports on a card by the number of sampling periods. For example, an ONS 15454 E100 card contains 24 ports, which multiplied by periods allows 96 rows in the table. An E1000 card contains 14 ports, which multiplied by four periods allows 56 table rows.

Table 16-9: RMON History Control Periods and History Categories

Sampling Periods (historyControlValue Variable)	Total Values, or Buckets (historyControl Variable)
15 minutes	32
24 hours	7
1 minute	60
60 minutes	24

Row Creation in historyControlTable

To activate a historyControlTable row, the SetRequest PDU must contain all needed values and have a status variable value of 2 (createRequest). All OIDs in the SetRequest PDU should be type OID.0 for entry creation.

To create a SetRequest PDU for the historyControlTable, the following values are required:

- The historyControlDataSource and its desired value

- The historyControlBucketsRequested and it desired value
- The historyControlInterval and its desired value
- The historyControlOwner and its desired value
- The historyControlStatus with a value of createRequest (2)

The historyControlBucketsRequested OID value is ignored because the number of buckets allowed for each sampling period, based upon the historyControlInterval value, is already fixed as listed in Table 16-9.

The historyControlInterval value cannot be changed from the four allowed choices. If you use another value, the SNMP agent selects the closest smaller time period from the set buckets. For example, if the set request specifies a 25-minute interval, this falls between the 15-minute (32 bucket) variable and the 60-minute (24 bucket) variable. The SNMP agent automatically selects the lower, closer value, which is 15 minutes, so it allows 32 buckets.

If the SetRequest PDU is valid, a historyControlTable row is created. If the row already exists, or if the SetRequest PDU values do not make sense or are insufficient, the SNMP agent does not create the row and returns an error code.

Get Requests and GetNext Requests

These PDUs are not restricted.

Row Deletion in historyControl Table

To delete a row from the table, the SetRequest PDU should contain a historyControlStatus value of 4 (invalid). A deleted row can be recreated.

Ethernet History RMON Group

The ONS 15454 implements the etherHistoryTable as defined in RFC 2819. The group is created within the bounds of the historyControlTable and does not deviate from the RFC in its design.

64-Bit etherHistoryHighCapacityTable

64-bit Ethernet history for the HC-RMON-MIB is implemented in the etherHistoryHighCapacityTable, which is an extension of the etherHistoryTable. The etherHistoryHighCapacityTable adds four columns for 64-bit performance monitoring data. These two tables have a one-to-one relationship. Adding or deleting a row in one table will effect the same change in the other.

Alarm RMON Group

The Alarm group consists of the alarmTable, which periodically compares sampled values with configured thresholds and raises an event if a threshold is crossed. This group requires the implementation of the event group, which follows this section.

Alarm Table

The NMS uses the alarmTable to determine and provision network performance alarmable thresholds.

Row Creation in alarmTable

To create a row in the alarmTable, all OIDs in the SetRequest PDU should be type OID.0. The table has a maximum number of 256 rows.

To create a SetRequest PDU for the alarmTable, the following values are required:

- The alarmInterval and its desired value
- The alarmVariable and its desired value
- The alarmSampleType and its desired value
- The alarmStartupAlarm and its desired value
- The alarmOwner and its desired value
- The alarmStatus with a value of createRequest (2)

If the SetRequest PDU is valid, a historyControlTable row is created. If the row already exists, or if the SetRequest PDU values do not make sense or are insufficient, the SNMP agent does not create the row and returns an error code.

In addition to the required values, the following restrictions must be met in the SetRequest PDU:

- The alarmOwner is a string of length 32 characters.
- The alarmRisingEventIndex always takes value 1.
- The alarmFallingEventIndex always takes value 2.
- The alarmStatus has only two values supported in SETs: createRequest (2) and invalid (4).
- The AlarmVariable is of the type OID.ifIndex, where ifIndex gives the interface this alarm is created on and OID is one of the OIDs supported in Table 16-10.

Table 16-10: OIDs Supported in the AlarmTable

No.	Column Name	OID	Status
1	ifInOctets	{1.3.6.1.2.1.2.2.1.10}	-
2	IfInUcastPkts	{1.3.6.1.2.1.2.2.1.11}	-
3	ifInMulticastPkts	{1.3.6.1.2.1.31.1.1.1.2}	Unsupported in E100/E1000
4	ifInBroadcastPkts	{1.3.6.1.2.1.31.1.1.1.3}	Unsupported in E100/E1000
5	ifInDiscards	{1.3.6.1.2.1.2.2.1.13}	Unsupported in E100/E1000
6	ifInErrors	{1.3.6.1.2.1.2.2.1.14}	-
7	ifOutOctets	{1.3.6.1.2.1.2.2.1.16}	-
8	ifOutUcastPkts	{1.3.6.1.2.1.2.2.1.17}	-
9	ifOutMulticastPkts	{1.3.6.1.2.1.31.1.1.1.4}	Unsupported in E100/E1000
10	ifOutBroadcastPkts	{1.3.6.1.2.1.31.1.1.1.5}	Unsupported in E100/E1000
11	ifOutDiscards	{1.3.6.1.2.1.2.2.1.19}	Unsupported in E100/E1000
12	Dot3StatsAlignmentErrors	{1.3.6.1.2.1.10.7.2.1.2}	-
13	Dot3StatsFCSErrors	{1.3.6.1.2.1.10.7.2.1.3}	-
14	Dot3StatsSingleCollisionFrames	{1.3.6.1.2.1.10.7.2.1.4}	-
15	Dot3StatsMultipleCollisionFrames	{1.3.6.1.2.1.10.7.2.1.5}	-
16	Dot3StatsDeferredTransmissions	{1.3.6.1.2.1.10.7.2.1.7}	-
17	Dot3StatsLateCollisions	{1.3.6.1.2.1.10.7.2.1.8}	-
18	Dot3StatsExcessiveCollisions	{13.6.1.2.1.10.7.2.1.9}	-
19	Dot3StatsFrameTooLong	{1.3.6.1.2.1.10.7.2.1.13}	-
20	Dot3StatsCarrierSenseErrors	{1.3.6.1.2.1.10.7.2.1.11}	Unsupported in E100/E1000
21	Dot3StatsSQETestErrors	{1.3.6.1.2.1.10.7.2.1.6}	Unsupported in E100/E1000
22	etherStatsUndersizePkts	{1.3.6.1.2.1.16.1.1.1.9}	-
23	etherStatsFragments	{1.3.6.1.2.1.16.1.1.1.11}	-

24	etherStatsPkts64Octets	{1.3.6.1.2.1.16.1.1.1.14}	-
25	etherStatsPkts65to127Octets	{1.3.6.1.2.1.16.1.1.1.15}	-
26	etherStatsPkts128to255Octets	{1.3.6.1.2.1.16.1.1.1.16}	-
27	etherStatsPkts256to511Octets	{1.3.6.1.2.1.16.1.1.1.17}	-
28	etherStatsPkts512to1023Octets	{1.3.6.1.2.1.16.1.1.1.18}	-
29	etherStatsPkts1024to1518Octets	{1.3.6.1.2.1.16.1.1.1.19}	-
30	EtherStatsBroadcastPkts	{1.3.6.1.2.1.16.1.1.1.6}	-
31	EtherStatsMulticastPkts	{1.3.6.1.2.1.16.1.1.1.7}	-
32	EtherStatsOversizePkts	{1.3.6.1.2.1.16.1.1.1.10}	-
33	EtherStatsJabbers	{1.3.6.1.2.1.16.1.1.1.12}	-
34	EtherStatsOctets	{1.3.6.1.2.1.16.1.1.1.4}	-
35	EtherStatsCollisions	{1.3.6.1.2.1.16.1.1.1.13}	-
36	EtherStatsCollisions	{1.3.6.1.2.1.16.1.1.1.8}	-
37	EtherStatsDropEvents	{1.3.6.1.2.1.16.1.1.1.3}	Unsupported in E100/E1000 and G1000

Get Requests and GetNext Requests

These PDUs are not restricted.

Row Deletion in alarmTable

To delete a row from the table, the SetRequest PDU should contain an alarmStatus value of 4 (invalid). A deleted row can be recreated.

Note: Entries in the alarmTable are preserved if the SNMP agent is restarted.

Event RMON Group

The Event group controls event generation and notification. It consists of two tables: the eventTable, which is a read-only list of events to be generated, and the logTable, which is a writable set of data describing a logged event. The ONS 15454 implements the logTable as specified in RFC 2819.

Event Table

The eventTable is read-only and unprovisionable. The table contains one row for rising alarms and another for falling ones. This table has the following restrictions:

- The eventType is always log-and-trap (4).
- The eventCommunity value is always a zero-length string, indicating that this event causes the trap to be despatched to all provisioned destinations.
- The eventOwner column value is always "monitor."
- The eventStatus column value is always valid(1).

Log Table

The logTable is implemented exactly as specified in RFC 2819. The logTable is based upon data that is locally cached in a controller card. If there is a controller card protection switch, the existing logTable is cleared and a new one is started on the newly active controller card. The table contains as many rows as provided by the alarm controller.