

Network Working Group  
Request for Comments: 1451

J. Case  
SNMP Research, Inc.  
K. McCloghrie  
Hughes LAN Systems  
M. Rose  
Dover Beach Consulting, Inc.  
S. Waldbusser  
Carnegie Mellon University  
April 1993

Manager-to-Manager  
Management Information Base

Status of this Memo

This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1 Introduction .....	2
1.1 A Note on Terminology .....	2
2 Overview .....	3
2.1 A SNMPv2 Entity Acting in a Dual Role .....	3
2.2 Alarms, Events, and Notifications .....	3
2.3 Access Control .....	4
3 Definitions .....	6
3.1 The Alarm Group .....	7
3.1.1 Alarm-Related Notifications .....	20
3.2 The Event Group .....	21
3.3 Conformance Information .....	29
3.3.1 Compliance Statements .....	29
3.3.2 Units of Conformance .....	29
4 Acknowledgements .....	31
5 References .....	35
6 Security Considerations .....	36
7 Authors' Addresses .....	36

## 1. Introduction

A network management system contains: several (potentially many) nodes, each with a processing entity, termed an agent, which has access to management instrumentation; at least one management station; and, a management protocol, used to convey management information between the agents and management stations. Operations of the protocol are carried out under an administrative framework which defines both authentication and authorization policies.

Network management stations execute management applications which monitor and control network elements. Network elements are devices such as hosts, routers, terminal servers, etc., which are monitored and controlled through access to their management information.

Management information is viewed as a collection of managed objects, residing in a virtual information store, termed the Management Information Base (MIB). Collections of related objects are defined in MIB modules. These modules are written using a subset of OSI's Abstract Syntax Notation One (ASN.1) [1], termed the Structure of Management Information (SMI) [2].

The management protocol, version 2 of the Simple Network Management Protocol [3], provides for the exchange of messages which convey management information between the agents and the management stations, including between management stations. It is the purpose of this document to define managed objects which describe the behavior of a SNMPv2 entity acting in both a manager role and an agent role.

### 1.1. A Note on Terminology

For the purpose of exposition, the original Internet-standard Network Management Framework, as described in RFCs 1155, 1157, and 1212, is termed the SNMP version 1 framework (SNMPv1). The current framework is termed the SNMP version 2 framework (SNMPv2).

## 2. Overview

The purpose of this MIB is to provide the means for coordination between multiple management stations. That is, the means by which the controlling and monitoring functions of network management can be distributed amongst multiple management stations. Such distribution facilitates the scaling of network management solutions based on the SNMPv2 to meet the needs of very large networks, or of networks composed of multiple interconnected administrations. Specifically, this MIB provides the means for one management station to request management services from another management station.

### 2.1. A SNMPv2 Entity Acting in a Dual Role

A management station providing services to other management station(s), is a SNMPv2 entity which acts in the dual role of both manager and agent; the requests for service are received through acting in an agent role (with respect to the managed objects defined in this MIB), and the requested services are performed through acting in a manager role.

### 2.2. Alarms, Events, and Notifications

In this initial version, this MIB defines the concepts of "alarms", "events", and "notifications". Each alarm is a specific condition detected through the periodic (at a configured sampling interval) monitoring of the value of a specific management information variable. An example of an alarm condition is when the monitored variable falls outside a configured range. Each alarm condition triggers an event, and each event can cause (one or more) notifications to be reported to other management stations using the Inform-Request PDU.

Specifically, this MIB defines three MIB tables and a number of scalar objects. The three tables are: the Alarm Table, the Event Table, and the Notification Table.

### 2.3. Access Control

The Administrative Model for SNMPv2 document [4] includes an access control model, which must not be subverted by allowing access to management information variables via the Alarm table. That is, access to a monitored variable via the Alarm table must be controlled according to the identity of the management station accessing the particular entry in the Alarm table.

An entry in the Alarm table provides the means to configure the sampling of the value of a MIB variable in the MIB view associated with the specified context (which can refer to object resources that are either local or remote). The sampling is done by (conceptually or actually) issuing a SNMPv2 request to retrieve the variable's value. This request is authenticated and/or protected from disclosure according to a source party and a destination party pair which has access to the indicated context.

Thus, to provide the required access control, the initial MIB view assigned, by convention, to parties on SNMPv2 entities that implement the `snmpAlarmTable`, must include the component:

```
viewSubtree = { snmpAlarm }
viewStatus  = { excluded }
viewMask    = { 'H }
```

Then, the MIB view associated with the context, `requestContext`, accessible by a requesting management station, can be configured to include specific Alarm table entries -- the ones associated with those contexts to which the requesting management station has access.

In particular, to provide a `requestContext` with access to the sampling context `sampleContext`, the following family of view subtrees would be included for the `requestContext` on the SNMPv2 entity acting in a dual role:

```
{ snmpAlarmEntry WILDCARD sampleContext }
```

Which would be configured in the party MIB [5] as:

```
contextIdentity = { requestContext }
contextViewIndex = { ViewIndex }
```

```
viewIndex      = { ViewIndex }
viewSubtree    = { snmpAlarmEntry 0 sampleContext }
viewStatus     = { included }
viewMask       = { 'FFEF'H } -- specifies wildcard for column
```

## 3. Definitions

```
SNMPv2-M2M-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,  
    Integer32, Counter32, snmpModules  
        FROM SNMPv2-SMI  
    DisplayString, InstancePointer, RowStatus, TimeStamp  
        FROM SNMPv2-TC  
    MODULE-COMPLIANCE, OBJECT-GROUP  
        FROM SNMPv2-CONF  
    contextIdentity  
        FROM SNMPv2-PARTY-MIB;
```

```
snmpM2M MODULE-IDENTITY
```

```
    LAST-UPDATED "9304010000Z"  
    ORGANIZATION "IETF SNMPv2 Working Group"  
    CONTACT-INFO  
        "  
            Steven Waldbusser
```

```
        Postal: Carnegie Mellon University  
                4910 Forbes Ave  
                Pittsburgh, PA 15213
```

```
        Tel: +1 412 268 6628  
        Fax: +1 412 268 4987
```

```
        E-mail: waldbusser@cmu.edu"
```

```
DESCRIPTION
```

```
    "The Manager-to-Manager MIB module."  
    ::= { snmpModules 2 }
```

```
snmpM2MObjects OBJECT IDENTIFIER ::= { snmpM2M 1 }
```

```
-- the alarm group
--
-- a collection of objects allowing the description and
-- configuration of threshold alarms from a SNMPv2 entity
-- acting in a dual role.

snmpAlarm      OBJECT IDENTIFIER ::= { snmpM2MObjects 1 }

-- This Alarm mechanism periodically takes statistical samples
-- from variables available via SNMPv2 and compares them to
-- thresholds that have been configured. The alarm table
-- stores configuration entries that each define a variable,
-- polling period, and threshold parameters. If a sample is
-- found to cross the threshold values, an event is generated.
-- Only variables that resolve to an ASN.1 primitive type of
-- INTEGER (Integer32, Counter32, Gauge32, TimeTicks,
-- Counter64, or UInteger32) may be monitored in this way.
--
-- This function has a hysteresis mechanism to limit the
-- generation of events. This mechanism generates one event
-- as a threshold is crossed in the appropriate direction. No
-- more events are generated for that threshold until the
-- opposite threshold is crossed.
--
-- In the case of sampling a deltaValue, an entity may
-- implement this mechanism with more precision if it takes a
-- delta sample twice per period, each time comparing the sum
-- of the latest two samples to the threshold. This allows
-- the detection of threshold crossings that span the sampling
-- boundary. Note that this does not require any special
-- configuration of the threshold value. It is suggested that
-- entities implement this more precise algorithm.
--
```

**snmpAlarmNextIndex OBJECT-TYPE**

SYNTAX INTEGER (0..65535)

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The index number of the next appropriate unassigned entry in the snmpAlarmTable. The value 0 indicates that no unassigned entries are available.

A management station should create new entries in the snmpAlarmTable using this algorithm: first, issue a management protocol retrieval operation to determine the value of snmpAlarmNextIndex; and, second, issue a management protocol set operation to create an instance of the snmpAlarmStatus object setting its value to 'createAndGo' or 'createAndWait' (as specified in the description of the RowStatus textual convention)."

::= { snmpAlarm 1 }

**snmpAlarmTable OBJECT-TYPE**

SYNTAX SEQUENCE OF SnmpAlarmEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"A list of snmpAlarm entries."

::= { snmpAlarm 2 }

**snmpAlarmEntry OBJECT-TYPE**

SYNTAX SnmpAlarmEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"A list of parameters that set up a periodic sampling query to check for alarm conditions. The contextIdentity included in the INDEX clause is the context to which the sampling queries are directed."

INDEX { contextIdentity, snmpAlarmIndex }

::= { snmpAlarmTable 1 }



```
SnmpAlarmEntry ::= SEQUENCE {
    snmpAlarmIndex          INTEGER,
    snmpAlarmVariable       InstancePointer,
    snmpAlarmInterval       Integer32,
    snmpAlarmSampleType     INTEGER,
    snmpAlarmValue          Integer32,
    snmpAlarmStartupAlarm   INTEGER,
    snmpAlarmRisingThreshold Integer32,
    snmpAlarmFallingThreshold Integer32,
    snmpAlarmRisingEventIndex INTEGER,
    snmpAlarmFallingEventIndex INTEGER,
    snmpAlarmUnavailableEventIndex INTEGER,
    snmpAlarmStatus         RowStatus
}

snmpAlarmIndex OBJECT-TYPE
    SYNTAX      INTEGER (1..65535)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "An index that uniquely identifies an entry in the
        snmpAlarm table for a particular sampling context.
        Each such entry defines a diagnostic sample at a
        particular interval for a variable in the
        particular context's object resources."
    ::= { snmpAlarmEntry 1 }
```

`snmpAlarmVariable OBJECT-TYPE``SYNTAX InstancePointer``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The object identifier of the particular variable to be sampled. Only variables that resolve to an ASN.1 primitive type of INTEGER (Integer32, Counter32, Gauge32, TimeTicks, Counter64, or UInteger32) may be sampled.

If it is detected by an error response of `authorizationError`, `noSuchObject`, or `noSuchInstance` that the variable name of an established `snmpAlarmEntry` is no longer available in the sampling context, a single `snmpObjectUnavailableAlarm` event is generated and the status of this `snmpAlarmEntry` is set to 'destroy'. Likewise, if the syntax of the variable retrieved by the query is not Integer32, Counter32, Gauge32, TimeTicks, Counter64, or UInteger32, the same actions will be taken.

If the SNMPv2 entity acting in a dual role detects that the sampled value can not be obtained due to lack of response to management queries, it should either:

- 1) Set the status of this `snmpAlarmEntry` to 'destroy', if it is determined that further communication is not possible;

or,

- 2) Delete the associated `snmpAlarmValue` instance (but not the entire conceptual row), and continue to attempt to sample the variable and recreate the associated `snmpAlarmValue` instance should communication be reestablished.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated `snmpAlarmStatus` object would be equal to 'active' both before and after the modification attempt."

```
::= { snmpAlarmEntry 2 }
```

snmpAlarmInterval OBJECT-TYPE

SYNTAX Integer32

UNITS "seconds"

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The interval in seconds over which the data is sampled and compared with the rising and falling thresholds. When setting this object and the sampling type is 'deltaValue', care should be taken to ensure that the change during this interval of the variable being sampled will not exceed the  $(-2^{31} \dots 2^{31}-1)$  range of the snmpAlarmValue.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated snmpAlarmStatus object would be equal to 'active' both before and after the modification attempt."

```
::= { snmpAlarmEntry 3 }
```

snmpAlarmSampleType OBJECT-TYPE

```
SYNTAX      INTEGER {  
                absoluteValue(1),  
                deltaValue(2)  
            }
```

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The method of sampling the selected variable and calculating the value to be compared against the thresholds. If the value of this object is 'absoluteValue', the value of the selected variable at the end of the sampling interval will be compared directly with both the snmpAlarmRisingThreshold and the snmpAlarmFallingThreshold values. If the value of this object is 'deltaValue', the value of the selected variable at the end of the sampling interval will be subtracted from its value at the end of the previous sampling interval, and the difference compared with both the snmpAlarmRisingThreshold and the snmpAlarmFallingThreshold values.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated snmpAlarmStatus object would be equal to 'active' both before and after the modification attempt."

```
DEFVAL { deltaValue }  
 ::= { snmpAlarmEntry 4 }
```

snmpAlarmValue OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of the statistic during the last sampling period. The value during the current sampling period is not made available until the period is completed. If the value of the statistic does not fit in the signed 32 bit representation of this object, it should be truncated in an implementation specific manner.

Note that if the associated snmpAlarmSampleType is set to 'deltaValue', the value of this object is the difference in the sampled variable since the last sample.

This object will be created by the SNMPv2 entity acting in a dual role when this entry is set to 'active', and the first sampling period has completed. It may be created and deleted at other times by the SNMPv2 entity acting in a dual role when the sampled value can not be obtained, as specified in the snmpAlarmVariable object."

::= { snmpAlarmEntry 5 }

## snmpAlarmStartupAlarm OBJECT-TYPE

```
SYNTAX      INTEGER {
                risingAlarm(1),
                fallingAlarm(2),
                risingOrFallingAlarm(3)
            }
```

```
MAX-ACCESS read-create
```

```
STATUS      current
```

## DESCRIPTION

"The alarm that may be sent when this entry is first set to 'active'. If the first sample after this entry becomes active is greater than or equal to the risingThreshold and snmpAlarmStartupAlarm is equal to 'risingAlarm' or 'risingOrFallingAlarm', then a single rising alarm will be generated. If the first sample after this entry becomes active is less than or equal to the fallingThreshold and snmpAlarmStartupAlarm is equal to 'fallingAlarm' or 'risingOrFallingAlarm', then a single falling alarm will be generated. Note that a snmpObjectUnavailableAlarm is sent upon startup whenever it is applicable, independent of the setting of snmpAlarmStartupAlarm.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated snmpAlarmStatus object would be equal to 'active' both before and after the modification attempt."

```
DEFVAL { risingOrFallingAlarm }
::= { snmpAlarmEntry 6 }
```

## snmpAlarmRisingThreshold OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

"A threshold for the sampled statistic. 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 will be generated. A single event will also be generated if the first sample after this entry becomes active is greater than or equal to this threshold and the associated snmpAlarmStartupAlarm is equal to 'risingAlarm' or 'risingOrFallingAlarm'.

After a rising event is generated, another such event will not be generated until the sampled value falls below this threshold and reaches the snmpAlarmFallingThreshold.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated snmpAlarmStatus object would be equal to 'active' both before and after the modification attempt."

::= { snmpAlarmEntry 7 }

## snmpAlarmFallingThreshold OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

"A threshold for the sampled statistic. 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 will be generated. A single event will also be generated if the first sample after this entry becomes active is less than or equal to this threshold and the associated snmpAlarmStartupAlarm is equal to 'fallingAlarm' or 'risingOrFallingAlarm'.

After a falling event is generated, another such event will not be generated until the sampled value rises above this threshold and reaches the snmpAlarmRisingThreshold.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated snmpAlarmStatus object would be equal to 'active' both before and after the modification attempt."

::= { snmpAlarmEntry 8 }



`snmpAlarmRisingEventIndex OBJECT-TYPE``SYNTAX INTEGER (0..65535)``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The index of the `snmpEventEntry` that is used when a rising threshold is crossed. The `snmpEventEntry` identified by a particular value of this index is the same as identified by the same value of the `snmpEventIndex` object. If there is no corresponding entry in the `snmpEventTable`, then no association exists. In particular, if this value is zero, no associated event will be generated, as zero is not a valid `snmpEventIndex`.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated `snmpAlarmStatus` object would be equal to 'active' both before and after the modification attempt."

`::= { snmpAlarmEntry 9 }`

`snmpAlarmFallingEventIndex OBJECT-TYPE``SYNTAX INTEGER (0..65535)``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The index of the `snmpEventEntry` that is used when a falling threshold is crossed. The `snmpEventEntry` identified by a particular value of this index is the same as identified by the same value of the `snmpEventIndex` object. If there is no corresponding entry in the `snmpEventTable`, then no association exists. In particular, if this value is zero, no associated event will be generated, as zero is not a valid `snmpEventIndex`.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated `snmpAlarmStatus` object would be equal to 'active' both before and after the modification attempt."

`::= { snmpAlarmEntry 10 }``snmpAlarmUnavailableEventIndex OBJECT-TYPE``SYNTAX INTEGER (0..65535)``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The index of the `snmpEventEntry` that is used when a variable becomes unavailable. The `snmpEventEntry` identified by a particular value of this index is the same as identified by the same value of the `snmpEventIndex` object. If there is no corresponding entry in the `snmpEventTable`, then no association exists. In particular, if this value is zero, no associated event will be generated, as zero is not a valid `snmpEventIndex`.

An attempt to modify this object will fail with an 'inconsistentValue' error if the associated `snmpAlarmStatus` object would be equal to 'active' both before and after the modification attempt."

`::= { snmpAlarmEntry 11 }`

## snmpAlarmStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

"The status of this snmpAlarm entry. This object may not be set to 'active' unless the following columnar objects exist in this row:

snmpAlarmVariable, snmpAlarmInterval,  
snmpAlarmSampleType, snmpAlarmStartupAlarm,  
snmpAlarmRisingThreshold,  
snmpAlarmFallingThreshold,  
snmpAlarmRisingEventIndex,  
snmpAlarmFallingEventIndex, and  
snmpAlarmUnavailableEventIndex."

::= { snmpAlarmEntry 12 }

-- alarm-related notifications

snmpAlarmNotifications

OBJECT IDENTIFIER ::= { snmpAlarm 3 }

snmpRisingAlarm NOTIFICATION-TYPE

OBJECTS { snmpAlarmVariable, snmpAlarmSampleType,  
snmpAlarmValue, snmpAlarmRisingThreshold }

STATUS current

DESCRIPTION

"An event that is generated when an alarm entry crosses its rising threshold. The instances of those objects contained within the varbind list are those of the alarm entry which generated this event."

::= { snmpAlarmNotifications 1 }

snmpFallingAlarm NOTIFICATION-TYPE

OBJECTS { snmpAlarmVariable, snmpAlarmSampleType,  
snmpAlarmValue, snmpAlarmFallingThreshold }

STATUS current

DESCRIPTION

"An event that is generated when an alarm entry crosses its falling threshold. The instances of those objects contained within the varbind list are those of the alarm entry which generated this event."

::= { snmpAlarmNotifications 2 }

snmpObjectUnavailableAlarm NOTIFICATION-TYPE

OBJECTS { snmpAlarmVariable }

STATUS current

DESCRIPTION

"An event that is generated when a variable monitored by an alarm entry becomes unavailable. The instance of snmpAlarmVariable contained within the varbind list is the one associated with the alarm entry which generated this event."

::= { snmpAlarmNotifications 3 }

```
-- the event group
--
-- a collection of objects allowing the description and
-- configuration of events from a SNMPv2 entity acting
-- in a dual role.

snmpEvent          OBJECT IDENTIFIER ::= { snmpM2MObjects 2 }

-- The snmpEvent table defines the set of events generated on
-- a SNMPv2 entity acting in a dual role. Each entry in the
-- snmpEventTable associates an event type with the
-- notification method and associated parameters. Some
-- snmpEvent entries are fired by an associated condition in
-- the snmpAlarmTable. Others are fired on behalf of
-- conditions defined in the NOTIFICATION-TYPE macro. The
-- snmpNotificationTable defines notifications that should
-- occur when an associated event is fired.

snmpEventNextIndex OBJECT-TYPE
    SYNTAX      INTEGER (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The index number of the next appropriate
        unassigned entry in the snmpEventTable. The value
        0 indicates that no unassigned entries are
        available.

        A management station should create new entries in
        the snmpEventTable using this algorithm: first,
        issue a management protocol retrieval operation to
        determine the value of snmpEventNextIndex; and,
        second, issue a management protocol set operation
        to create an instance of the snmpEventStatus
        object setting its value to 'createAndWait' or
        'createAndGo'."
    ::= { snmpEvent 1 }
```

```
snmpEventTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SnmpEventEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A list of events."
    ::= { snmpEvent 2 }
```

```
snmpEventEntry OBJECT-TYPE
    SYNTAX      SnmpEventEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of parameters that describe an event that
         is generated when certain conditions are met."
    INDEX       { snmpEventIndex }
    ::= { snmpEventTable 1 }
```

```
SnmpEventEntry ::= SEQUENCE {
    snmpEventIndex      INTEGER,
    snmpEventID         OBJECT IDENTIFIER,
    snmpEventDescription DisplayString,
    snmpEventEvents     Counter32,
    snmpEventLastTimeSent TimeStamp,
    snmpEventStatus     RowStatus
}
```

```
snmpEventIndex OBJECT-TYPE
    SYNTAX      INTEGER (1..65535)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "An index that uniquely identifies an entry in the
         snmpEvent table. Each such entry defines an event
         generated when the appropriate conditions occur."
    ::= { snmpEventEntry 1 }
```

```
snmpEventID OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The authoritative identification of the event
        type generated by this entry.  This variable
        occurs as the second varbind of an InformRequest-
        PDU.  If this OBJECT IDENTIFIER maps to a
        NOTIFICATION-TYPE the sender will place the
        objects listed in the NOTIFICATION-TYPE in the
        varbind list."
    ::= { snmpEventEntry 2 }

snmpEventDescription OBJECT-TYPE
    SYNTAX      DisplayString (SIZE (0..127))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "A comment describing this snmpEvent entry."
    ::= { snmpEventEntry 3 }

snmpEventEvents OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of events caused by event generators
        associated with this snmpEvent entry."
    ::= { snmpEventEntry 4 }
```

**snmpEventLastTimeSent OBJECT-TYPE**

SYNTAX       TimeStamp

MAX-ACCESS read-only

STATUS       current

## DESCRIPTION

"The value of sysUpTime at the time this snmpEvent entry last generated an event. If this entry has not generated any events, this value will be zero."

DEFVAL { 0 }

::= { snmpEventEntry 5 }

**snmpEventStatus OBJECT-TYPE**

SYNTAX       RowStatus

MAX-ACCESS read-create

STATUS       current

## DESCRIPTION

"The status of this snmpEvent entry. This object may not be set to 'active' unless the following columnar objects exist in this row: snmpEventID, snmpEventDescription, snmpEventEvents, and snmpEventLastTimeSent.

Setting an instance of this object to the value 'destroy' has the effect of invalidating any/all entries in the snmpEventTable, and the snmpEventNotifyTable which reference the corresponding snmpEventEntry."

::= { snmpEventEntry 6 }



`snmpEventNotifyMinInterval OBJECT-TYPE``SYNTAX Integer32``UNITS "seconds"``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"The minimum interval that the SNMPv2 entity acting in a dual role will wait before retransmitting an InformRequest-PDU. This object specifies the minimal value supported by the SNMPv2 entity acting in a dual role, based on resource or implementation constraints.

For a particular entry in the `snmpEventNotifyTable`, if the associated `snmpEventNotifyIntervalRequested` variable is greater than this object, the `snmpEventNotifyIntervalRequested` value shall be used as the minimum interval for retransmissions of InformRequest-PDUs sent on behalf of that entry."

`::= { snmpEvent 3 }``snmpEventNotifyMaxRetransmissions OBJECT-TYPE``SYNTAX Integer32``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"The maximum number of time that the SNMPv2 entity acting in a dual role will retransmit an InformRequest-PDU. This object specifies the maximal value supported by the SNMPv2 entity acting in a dual role, based on resource or implementation constraints.

For a particular entry in the `snmpEventNotifyTable`, if the associated `snmpEventNotifyRetransmissionsRequested` variable is less than this object, the `snmpEventNotifyRetransmissionsRequested` value shall be used as the retransmission count for InformRequest-PDUs sent on behalf of that entry."

`::= { snmpEvent 4 }`

-- The `snmpEventNotifyTable` is used to configure the

```
-- destination and type of notifications sent by a SNMPv2
-- entity acting in a manager role when a particular event
-- is triggered.
```

```
snmpEventNotifyTable OBJECT-TYPE
```

```
    SYNTAX      SEQUENCE OF SnmpEventNotifyEntry
```

```
    MAX-ACCESS  not-accessible
```

```
    STATUS      current
```

```
    DESCRIPTION
```

```
        "A list of protocol configuration entries for
        event notifications from this entity."
```

```
    ::= { snmpEvent 5 }
```

```
snmpEventNotifyEntry OBJECT-TYPE
```

```
    SYNTAX      SnmpEventNotifyEntry
```

```
    MAX-ACCESS  not-accessible
```

```
    STATUS      current
```

```
    DESCRIPTION
```

```
        "A set of parameters that describe the type and
        destination of InformRequest-PDUs sent for a
        particular event.  The snmpEventIndex in this
        entry's INDEX clause identifies the snmpEventEntry
        which, when triggered, will generate a
        notification as configured in this entry.  The
        contextIdentity in this entry's INDEX clause
        identifies the context to which a notification
        will be sent."
```

```
    INDEX      { snmpEventIndex, contextIdentity }
```

```
    ::= { snmpEventNotifyTable 1 }
```

```
SnmpEventNotifyEntry ::= SEQUENCE {
```

```
    snmpEventNotifyIntervalRequested      Integer32,
```

```
    snmpEventNotifyRetransmissionsRequested Integer32,
```

```
    snmpEventNotifyLifetime               Integer32,
```

```
    snmpEventNotifyStatus                 RowStatus
```

```
}
```

`snmpEventNotifyIntervalRequested OBJECT-TYPE``SYNTAX Integer32``UNITS "seconds"``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The requested interval for retransmission of Inform PDUs generated on the behalf of this entry.

This variable will be the actual interval used unless the `snmpEventNotifyMinInterval` is greater than this object, in which case the interval shall be equal to `snmpEventNotifyMinInterval`."

`DEFVAL { 30 }``::= { snmpEventNotifyEntry 1 }``snmpEventNotifyRetransmissionsRequested OBJECT-TYPE``SYNTAX Integer32``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The requested number of retransmissions of an InformRequest-PDU generated on behalf of this entry.

This variable will be the actual number of retransmissions used unless the `snmpEventNotifyMaxRetransmissions` is less than this object, in which case the retransmission count shall be equal to `snmpEventNotifyMaxRetransmissions`."

`DEFVAL { 5 }``::= { snmpEventNotifyEntry 2 }`

`snmpEventNotifyLifetime OBJECT-TYPE``SYNTAX Integer32``UNITS "seconds"``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The number of seconds this entry shall live until the corresponding instance of `snmpEventNotifyStatus` is set to 'destroy'. This value shall count down to zero, at which time the corresponding instance of `snmpEventNotifyStatus` will be set to 'destroy'. Any management station that is using this entry must periodically refresh this value to ensure the continued delivery of events."

`DEFVAL { 86400 }``::= { snmpEventNotifyEntry 3 }``snmpEventNotifyStatus OBJECT-TYPE``SYNTAX RowStatus``MAX-ACCESS read-create``STATUS current``DESCRIPTION`

"The state of this `snmpEventNotifyEntry`. This object may not be set to 'active' unless the following columnar objects exist in this row: `snmpEventNotifyIntervalRequested`, `snmpEventNotifyRetransmissionsRequested`, and `snmpEventNotifyLifetime`."

`::= { snmpEventNotifyEntry 4 }`

```
-- conformance information

snmpM2MConformance
    OBJECT IDENTIFIER ::= { snmpM2M 2 }

snmpM2MCompliances
    OBJECT IDENTIFIER ::= { snmpM2MConformance 1 }
snmpM2MGroups OBJECT IDENTIFIER ::= { snmpM2MConformance 2 }

-- compliance statements

snmpM2MCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "The compliance statement for SNMPv2 entities
        which implement the Manager-to-Manager MIB."
    MODULE -- this module
        MANDATORY-GROUPS { snmpAlarmGroup, snmpEventGroup }
    ::= { snmpM2MCompliances 1 }

-- units of conformance

snmpAlarmGroup OBJECT-GROUP
    OBJECTS { snmpAlarmNextIndex,
              snmpAlarmVariable, snmpAlarmInterval,
              snmpAlarmSampleType, snmpAlarmValue,
              snmpAlarmStartupAlarm, snmpAlarmRisingThreshold,
              snmpAlarmFallingThreshold,
              snmpAlarmRisingEventIndex,
              snmpAlarmFallingEventIndex,
              snmpAlarmUnavailableEventIndex,
              snmpAlarmStatus }
    STATUS current
    DESCRIPTION
        "A collection of objects allowing the description
        and configuration of threshold alarms from a
        SNMPv2 entity acting in a dual role."
    ::= { snmpM2MGroups 1 }
```

```
snmpEventGroup OBJECT-GROUP
  OBJECTS { snmpEventNextIndex,
             snmpEventID, snmpEventDescription,
             snmpEventEvents, snmpEventLastTimeSent,
             snmpEventStatus, snmpEventNotifyMinInterval,
             snmpEventNotifyMaxRetransmissions,
             snmpEventNotifyIntervalRequested,
             snmpEventNotifyRetransmissionsRequested,
             snmpEventNotifyLifetime, snmpEventNotifyStatus }
  STATUS current
  DESCRIPTION
    "A collection of objects allowing the description
    and configuration of events from a SNMPv2 entity
    acting in a dual role."
  ::= { snmpM2MGroups 2 }
```

END

#### 4. Acknowledgements

The comments of the SNMP version 2 working group are gratefully acknowledged:

Beth Adams, Network Management Forum  
Steve Alexander, INTERACTIVE Systems Corporation  
David Arneson, Cabletron Systems  
Toshiya Asaba  
Fred Baker, ACC  
Jim Barnes, Xylogics, Inc.  
Brian Bataille  
Andy Bierman, SynOptics Communications, Inc.  
Uri Blumenthal, IBM Corporation  
Fred Bohle, Interlink  
Jack Brown  
Theodore Brunner, Bellcore  
Stephen F. Bush, GE Information Services  
Jeffrey D. Case, University of Tennessee, Knoxville  
John Chang, IBM Corporation  
Szusin Chen, Sun Microsystems  
Robert Ching  
Chris Chiotasso, Ungermann-Bass  
Bobby A. Clay, NASA/Boeing  
John Cooke, Chipcom  
Tracy Cox, Bellcore  
Juan Cruz, Datability, Inc.  
David Cullerot, Cabletron Systems  
Cathy Cunningham, Microcom  
James R. (Chuck) Davin, Bellcore  
Michael Davis, Clearpoint  
Mike Davison, FiberCom  
Cynthia DellaTorre, MITRE  
Taso N. Devetzis, Bellcore  
Manual Diaz, DAVID Systems, Inc.  
Jon Dreyer, Sun Microsystems  
David Engel, Optical Data Systems  
Mike Erlinger, Lexcel  
Roger Fajman, NIH  
Daniel Fauvarque, Sun Microsystems  
Karen Frisa, CMU  
Shari Galitzer, MITRE  
Shawn Gallagher, Digital Equipment Corporation  
Richard Graveman, Bellcore  
Maria Greene, Xyplex, Inc.

Michel Guittet, Apple  
Robert Gutierrez, NASA  
Bill Hagerty, Cabletron Systems  
Gary W. Haney, Martin Marietta Energy Systems  
Patrick Hanil, Nokia Telecommunications  
Matt Hecht, SNMP Research, Inc.  
Edward A. Heiner, Jr., Synernetics Inc.  
Susan E. Hicks, Martin Marietta Energy Systems  
Gerald Holzhauser, Apple  
John Hopprich, DAVID Systems, Inc.  
Jeff Hughes, Hewlett-Packard  
Robin Iddon, Axon Networks, Inc.  
David Itusak  
Kevin M. Jackson, Concord Communications, Inc.  
Ole J. Jacobsen, Interop Company  
Ronald Jacoby, Silicon Graphics, Inc.  
Satish Joshi, SynOptics Communications, Inc.  
Frank Kastenholz, FTP Software  
Mark Kepke, Hewlett-Packard  
Ken Key, SNMP Research, Inc.  
Zbiginew Kielczewski, Eicon  
Jongyeoi Kim  
Andrew Knutsen, The Santa Cruz Operation  
Michael L. Kornegay, VisiSoft  
Deirdre C. Kostik, Bellcore  
Cheryl Krupczak, Georgia Tech  
Mark S. Lewis, Telebit  
David Lin  
David Lindemulder, AT&T/NCR  
Ben Lisowski, Sprint  
David Liu, Bell-Northern Research  
John Lunny, The Wollongong Group  
Robert C. Lushbaugh Martin, Marietta Energy Systems  
Michael Luufer, BBN  
Carl Madison, Star-Tek, Inc.  
Keith McCloghrie, Hughes LAN Systems  
Evan McGinnis, 3Com Corporation  
Bill McKenzie, IBM Corporation  
Donna McMaster, SynOptics Communications, Inc.  
John Medicke, IBM Corporation  
Doug Miller, Telebit  
Dave Minnich, FiberCom  
Mohammad Mirhakkak, MITRE  
Rohit Mital, Protools  
George Mouradian, AT&T Bell Labs



Patrick Mullaney, Cabletron Systems  
Dan Myers, 3Com Corporation  
Rina Nathaniel, Rad Network Devices Ltd.  
Hien V. Nguyen, Sprint  
Mo Nikain  
Tom Nisbet  
William B. Norton, MERIT  
Steve Onishi, Wellfleet Communications, Inc.  
David T. Perkins, SynOptics Communications, Inc.  
Carl Powell, BBN  
Ilan Raab, SynOptics Communications, Inc.  
Richard Ramons, AT&T  
Venkat D. Rangan, Metric Network Systems, Inc.  
Louise Reingold, Sprint  
Sam Roberts, Farallon Computing, Inc.  
Kary Robertson, Concord Communications, Inc.  
Dan Romascanu, Lannet Data Communications Ltd.  
Marshall T. Rose, Dover Beach Consulting, Inc.  
Shawn A. Routhier, Epilogue Technology Corporation  
Chris Rozman  
Asaf Rubissa, Fibronics  
Jon Saperia, Digital Equipment Corporation  
Michael Sapich  
Mike Scanlon, Interlan  
Sam Schaen, MITRE  
John Seligson, Ultra Network Technologies  
Paul A. Serice, Corporation for Open Systems  
Chris Shaw, Banyan Systems  
Timon Sloane  
Robert Snyder, Cisco Systems  
Joo Young Song  
Roy Spitier, Sprint  
Einar Stefferud, Network Management Associates  
John Stephens, Cayman Systems, Inc.  
Robert L. Stewart, Xyplex, Inc. (chair)  
Kaj Tesink, Bellcore  
Dean Throop, Data General  
Ahmet Tuncay, France Telecom-CNET  
Maurice Turcotte, Racal Datacom  
Warren Vik, INTERACTIVE Systems Corporation  
Yannis Viniotis  
Steven L. Waldbusser, Carnegie Mellon University  
Timothy M. Walden, ACC  
Alice Wang, Sun Microsystems  
James Watt, Newbridge

Luanne Waul, Timeplex  
Donald E. Westlake III, Digital Equipment Corporation  
Gerry White  
Bert Wijnen, IBM Corporation  
Peter Wilson, 3Com Corporation  
Steven Wong, Digital Equipment Corporation  
Randy Worzella, IBM Corporation  
Daniel Woycke, MITRE  
Honda Wu  
Jeff Yarnell, Protools  
Chris Young, Cabletron  
Kiho Yum, 3Com Corporation

## 5. References

- [1] Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), International Organization for Standardization. International Standard 8824, (December, 1987).
- [2] Case, J., McCloghrie, K., Rose, M., and Waldbusser, S., "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1442, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.
- [3] Case, J., McCloghrie, K., Rose, M., and Waldbusser, S., "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1448, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.
- [4] Galvin, J., and McCloghrie, K., "Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1445, Trusted Information Systems, Hughes LAN Systems, April 1993.
- [5] McCloghrie, K., and Galvin, J., "Party MIB for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1447, Hughes LAN Systems, Trusted Information Systems, April 1993.

## 6. Security Considerations

Security issues are not discussed in this memo.

## 7. Authors' Addresses

Jeffrey D. Case  
SNMP Research, Inc.  
3001 Kimberlin Heights Rd.  
Knoxville, TN 37920-9716  
US

Phone: +1 615 573 1434  
Email: case@snmp.com

Keith McCloghrie  
Hughes LAN Systems  
1225 Charleston Road  
Mountain View, CA 94043  
US

Phone: +1 415 966 7934  
Email: kzm@hls.com

Marshall T. Rose  
Dover Beach Consulting, Inc.  
420 Whisman Court  
Mountain View, CA 94043-2186  
US

Phone: +1 415 968 1052  
Email: mrose@dbc.mtview.ca.us

Steven Waldbusser  
Carnegie Mellon University  
4910 Forbes Ave  
Pittsburgh, PA 15213  
US

Phone: +1 412 268 6628  
Email: waldbusser@cmu.edu

