

Remote Monitoring MIB Extensions for High Capacity Alarms

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2002). All Rights Reserved.

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects for extending the alarm thresholding capabilities found in the Remote Monitoring (RMON) MIB (RFC 2819), to provide similar threshold monitoring of objects based on the Counter64 data type.

Table of Contents

1	The Internet-Standard Management Framework	2
2	Terms	2
3	Overview	2
3.1	Relationship to the Remote Monitoring MIBs	3
4	MIB Structure	4
4.1	MIB Group Overview	4
4.1.1	High Capacity Alarm Control Group	5
4.1.2	High Capacity Alarm Capabilities	6
4.1.3	High Capacity Alarm Notifications	6
5	Definitions	6
6	Intellectual Property	21
7	Acknowledgements	21
8	Normative References	21
9	Informative References	22

10 Security Considerations	22
11 Authors' Addresses	23
12 Full Copyright Statement	24

1. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

2. Terms

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119. [RFC2119]

3. Overview

There is a need for a standardized way of providing the same type of alarm thresholding capabilities for Counter64 objects, as already exists for Counter32 objects. The RMON-1 alarmTable objects and RMON-1 notification types are specific to 32-bit objects, and cannot be used to properly monitor Counter64-based objects. Extensions to these existing constructs which explicitly support Counter64-based objects are needed. These extensions are completely independent of the existing RMON-1 alarm mechanisms.

The usage of Counter64 objects is increasing. One of the causes for this increase is the increasing speeds of network interfaces; RFC 2863 [RFC2863] says:

As the speed of network media increase, the minimum time in which a 32 bit counter will wrap decreases. For example, a 10Mbps stream of back-to-back, full-size packets causes ifInOctets to wrap in just over 57 minutes; at 100Mbps, the minimum wrap time is 5.7 minutes, and at 1Gbs, the minimum is 34 seconds. Requiring that interfaces be polled frequently enough not to miss a counter wrap is increasingly problematic.

and therefore requires:

For interfaces that operate at 20,000,000 (20 million) bits per second or less, 32-bit byte and packet counters MUST be supported. For interfaces that operate faster than 20,000,000 bits/second, and slower than 650,000,000 bits/second, 32-bit packet counters MUST be supported and 64-bit octet counters MUST be supported. For interfaces that operate at 650,000,000 bits/second or faster, 64-bit packet counters AND 64-bit octet counters MUST be supported.

Of the variables on which thresholds are set using RMON-1's alarmTable, two of the most popular are: ifInOctets and ifOutOctets. Thus, the increasing usage of the 64-bit versions: ifHCInOctets and ifHCOctets means that there is an increasing requirement to use RMON-1's thresholding capability for ifHCInOctets and ifHCOctets.

The RMON-1 Alarm Group is implemented not only by all RMON probes, but also by the SNMP agents in many other types of devices for the purpose of monitoring any of their (non-RMON) integer-valued MIB objects. The fact that it has been so widely implemented indicates its obvious value. Without this extension, that obvious value is becoming incomplete because of its lack of support for 64-bit integers. This extension is the easiest, simplest, and most compatible way for an implementation to overcome that lack of support.

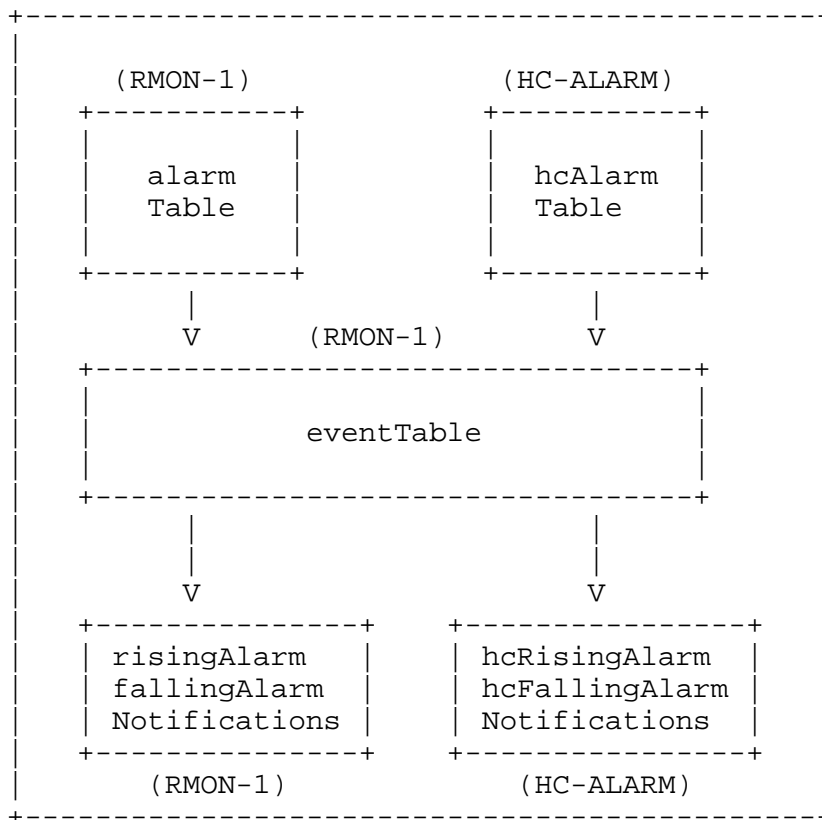
3.1. Relationship to the Remote Monitoring MIBs

This MIB is intended to be implemented in Remote Monitoring (RMON) probes, which may also support the RMON-1 MIB [RFC2819]. Such probes may be stand-alone devices, or may be co-located with other networking devices (e.g., ethernet switches and repeaters).

The functionality of the High Capacity Alarm Group is a superset of RMON-1's Alarm Group. Thus, one day in the distant future, it is a possibility that RMON-1's Alarm Group will be deprecated in favor of this MIB's High Capacity Alarm Group. However, that day will not come before this document, or one of its successors, reaches the same standardization state as RMON-1.

4. MIB Structure

Figure 1: HC-ALARM MIB Functional Structure



4.1. MIB Group Overview

The HC-ALARM MIB contains three MIB groups:

- hcAlarmControlObjects group
Controls the configuration of alarms for high capacity MIB object instances.
- hcAlarmCapabilities group
Describes the high capacity alarm capabilities provided by the agent.
- hcAlarmNotifications group
Provide new rising and falling threshold notifications for high capacity objects.

4.1.1. High Capacity Alarm Control Group

This group contains one table, which is used by a management station to configure high capacity alarm entries. To configure alarm thresholding for Counter64 or CounterBasedGauge64 objects, a management application must configure the hcAlarmTable in a manner similar to how RMON-1's alarmTable is configured.

Because the language in some of the DESCRIPTION clauses of objects in the alarmTable is specific to the alarmTable itself, their defined semantics do not allow them to be used for this MIB also. Therefore, the following objects are essentially cloned from the alarmTable to the hcAlarmTable:

alarmTable	hcAlarmTable
-----	-----
alarmIndex	hcAlarmIndex
alarmInterval	hcAlarmInterval
alarmVariable	hcAlarmVariable
alarmSampleType	hcAlarmSampleType
alarmStartupAlarm	hcAlarmStartupAlarm
alarmRisingEventIndex	hcAlarmRisingEventIndex
alarmFallingEventIndex	hcAlarmFallingEventIndex
alarmOwner	hcAlarmOwner
alarmStatus	hcAlarmStatus

In addition, the following hcAlarmTable objects are used as high capacity values instead of the corresponding 32-bit version in the alarmTable.

alarmTable	hcAlarmTable
-----	-----
alarmValue	hcAlarmAbsValue
	hcAlarmValueStatus
alarmRisingThreshold	hcAlarmRisingThreshAbsValueLo
	hcAlarmRisingThreshAbsValueHi
	hcAlarmRisingThresholdValStatus
alarmFallingThreshold	hcAlarmFallingThreshAbsValueLo
	hcAlarmFallingThreshAbsValueHi
	hcAlarmFallingThresholdValStatus

Nevertheless, the hcAlarmTable does have a few differences from the alarmTable:

- Counter64 based objects are thresholded properly
- an entry is not destroyed if the instance identified by the hcAlarmVariable is not available during a polling interval.

- the RowStatus textual convention is used instead of EntryStatus for the hcAlarmStatus object.
- the non-volatile storage of an HC alarm entry is explicitly controlled with a StorageType parameter.
- a counter is provided to indicate the number of times the hcAlarmVariable object value could not be retrieved by the agent.

4.1.2. High Capacity Alarm Capabilities

This group contains a single scalar object, called hcAlarmCapabilities. It describes the basic high capacity alarm features supported by the agent.

4.1.3. High Capacity Alarm Notifications

This group contains two notifications, hcRisingAlarm and hcFallingAlarm. These are generated for high capacity alarms in the same manner and used to convey essentially the same information as RMON-1's risingAlarm and fallingAlarm notifications do for alarmTable-specified alarms.

5. Definitions

HC-ALARM-MIB DEFINITIONS ::= BEGIN

IMPORTS

```
MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
Integer32, Counter32, Unsigned32
    FROM SNMPv2-SMI
MODULE-COMPLIANCE, OBJECT-GROUP,
NOTIFICATION-GROUP
    FROM SNMPv2-CONF
RowStatus, VariablePointer, StorageType,
TEXTUAL-CONVENTION
    FROM SNMPv2-TC
CounterBasedGauge64
    FROM HCNM-TC
rmon, OwnerString, rmonEventGroup
    FROM RMON-MIB;
```

hcAlarmMIB MODULE-IDENTITY

```
LAST-UPDATED      "200212160000Z"
ORGANIZATION      "IETF RMONMIB Working Group"
CONTACT-INFO
    "
        Andy Bierman
        Cisco Systems, Inc.
        Tel: +1 408 527-3711
```

E-mail: abierman@cisco.com
 Postal: 170 West Tasman Drive
 San Jose, CA USA 95134

Keith McCloghrie
 Cisco Systems, Inc.
 Tel: +1 408 526-5260
 E-mail: kzm@cisco.com
 Postal: 170 West Tasman Drive
 San Jose, CA USA 95134

Send comments to <rmonmib@ietf.org>
 Mailing list subscription info:
<http://www.ietf.org/mailman/listinfo/rmonmib> "

DESCRIPTION

"This module defines Remote Monitoring MIB extensions for High Capacity Alarms.

Copyright (C) The Internet Society (2002). This version of this MIB module is part of RFC 3434; see the RFC itself for full legal notices."

REVISION "200212160000Z"

DESCRIPTION

"Initial version of the High Capacity Alarm MIB module.
 This version published as RFC 3434."

::= { rmon 29 }

hcAlarmObjects OBJECT IDENTIFIER ::= { hcAlarmMIB 1 }
 hcAlarmNotifications OBJECT IDENTIFIER ::= { hcAlarmMIB 2 }
 hcAlarmConformance OBJECT IDENTIFIER ::= { hcAlarmMIB 3 }

hcAlarmControlObjects OBJECT IDENTIFIER ::= { hcAlarmObjects 1 }
 hcAlarmCapabilitiesObjects OBJECT IDENTIFIER
 ::= { hcAlarmObjects 2 }

--

-- Textual Conventions

--

HcValueStatus ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"This data type indicates the validity and sign of the data in associated object instances which represent the absolute value of a high capacity numeric quantity. Such an object may be represented with one or more object instances. An object of type HcValueStatus MUST be defined within the same

structure as the object(s) representing the high capacity absolute value.

If the associated object instance(s) representing the high capacity absolute value could not be accessed during the sampling interval, and is therefore invalid, then the associated HcValueStatus object will contain the value 'valueNotAvailable(1)'.

If the associated object instance(s) representing the high capacity absolute value are valid and actual value of the sample is greater than or equal to zero, then the associated HcValueStatus object will contain the value 'valuePositive(2)'.

If the associated object instance(s) representing the high capacity absolute value are valid and the actual value of the sample is less than zero, then the associated HcValueStatus object will contain the value 'valueNegative(3)'. The associated absolute value should be multiplied by -1 to obtain the true sample value."

```
SYNTAX INTEGER {
    valueNotAvailable(1),
    valuePositive(2),
    valueNegative(3)
}
```

```
--
-- High Capacity Alarm Table
--
```

```
hcAlarmTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF HcAlarmEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A list of entries for the configuration of high capacity
        alarms."
    ::= { hcAlarmControlObjects 1 }
```

```
hcAlarmEntry OBJECT-TYPE
    SYNTAX      HcAlarmEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A conceptual row in the hcAlarmTable. Entries are usually
        created in this table by management application action, but
        may also be created by agent action as well."
```



```

INDEX { hcAlarmIndex }
 ::= { hcAlarmTable 1 }

```

```

HcAlarmEntry ::= SEQUENCE {
    hcAlarmIndex                Integer32,
    hcAlarmInterval             Integer32,
    hcAlarmVariable             VariablePointer,
    hcAlarmSampleType           INTEGER,
    hcAlarmAbsValue             CounterBasedGauge64,
    hcAlarmValueStatus          HcValueStatus,
    hcAlarmStartupAlarm         INTEGER,
    hcAlarmRisingThreshAbsValueLo Unsigned32,
    hcAlarmRisingThreshAbsValueHi Unsigned32,
    hcAlarmRisingThresholdValStatus HcValueStatus,
    hcAlarmFallingThreshAbsValueLo Unsigned32,
    hcAlarmFallingThreshAbsValueHi Unsigned32,
    hcAlarmFallingThresholdValStatus HcValueStatus,
    hcAlarmRisingEventIndex      Integer32,
    hcAlarmFallingEventIndex     Integer32,
    hcAlarmValueFailedAttempts   Counter32,
    hcAlarmOwner                 OwnerString,
    hcAlarmStorageType           StorageType,
    hcAlarmStatus                RowStatus }

```

hcAlarmIndex OBJECT-TYPE

SYNTAX Integer32 (1..65535)

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An arbitrary integer index value used to uniquely identify this high capacity alarm entry."

```
 ::= { hcAlarmEntry 1 }
```

hcAlarmInterval OBJECT-TYPE

SYNTAX Integer32 (1..2147483647)

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 variable, care should be taken in the case of deltaValue sampling - the interval should be set short enough that the sampled variable is very unlikely to increase or decrease by more than $2^{63} - 1$ during a single sampling interval."

This object may not be modified if the associated
hcAlarmStatus object is equal to active(1)."
 ::= { hcAlarmEntry 2 }

hcAlarmVariable OBJECT-TYPE
SYNTAX VariablePointer
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 (INTEGER, Integer32, Counter32, Counter64,
Gauge, or TimeTicks) may be sampled.

Because SNMP access control is articulated entirely in terms
of the contents of MIB views, no access control mechanism
exists that can restrict the value of this object to
identify only those objects that exist in a particular MIB
view. Because there is thus no acceptable means of
restricting the read access that could be obtained through
the alarm mechanism, the probe must only grant write access
to this object in those views that have read access to all
objects on the probe.

This object may not be modified if the associated
hcAlarmStatus object is equal to active(1)."
 ::= { hcAlarmEntry 3 }

hcAlarmSampleType 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(1), the value
of the selected variable will be compared directly with the
thresholds at the end of the sampling interval. If the
value of this object is deltaValue(2), the value of the
selected variable at the last sample will be subtracted from
the current value, and the difference compared with the
thresholds.

If the associated hcAlarmVariable instance could not be
obtained at the previous sample interval, then a delta

sample is not possible, and the value of the associated hcAlarmValueStatus object for this interval will be valueNotAvailable(1).

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 4 }

hcAlarmAbsValue OBJECT-TYPE

SYNTAX CounterBasedGauge64

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The absolute value (i.e., unsigned value) of the hcAlarmVariable statistic during the last sampling period. The value during the current sampling period is not made available until the period is completed.

To obtain the true value for this sampling interval, the associated instance of hcAlarmValueStatus must be checked, and the value of this object adjusted as necessary.

If the MIB instance could not be accessed during the sampling interval, then this object will have a value of zero and the associated instance of hcAlarmValueStatus will be set to 'valueNotAvailable(1)'."

::= { hcAlarmEntry 5 }

hcAlarmValueStatus OBJECT-TYPE

SYNTAX HcValueStatus

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object indicates the validity and sign of the data for the hcAlarmAbsValue object, as described in the HcValueStatus textual convention."

::= { hcAlarmEntry 6 }

hcAlarmStartupAlarm 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 rising threshold and this object is equal to risingAlarm(1) or risingOrFallingAlarm(3), then a single rising alarm will be generated. If the first sample after this entry becomes valid is less than or equal to the falling threshold and this object is equal to fallingAlarm(2) or risingOrFallingAlarm(3), then a single falling alarm will be generated.

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 7 }

hcAlarmRisingThreshAbsValueLo OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The lower 32 bits of the absolute value for threshold for the sampled statistic. The actual threshold value is determined by the associated instances of the hcAlarmRisingThreshAbsValueHi and hcAlarmRisingThresholdValStatus objects, as follows:

$$\text{ABS}(\text{threshold}) = \text{hcAlarmRisingThreshAbsValueLo} + (\text{hcAlarmRisingThreshAbsValueHi} * 2^{32})$$

The absolute value of the threshold is adjusted as required, as described in the HcValueStatus textual convention. These three object instances are conceptually combined to represent the rising threshold for this entry.

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 valid is greater than or equal to this threshold and the associated hcAlarmStartupAlarm is equal to risingAlarm(1) or risingOrFallingAlarm(3).

After a rising event is generated, another such event will not be generated until the sampled value falls below this threshold and reaches the threshold identified by the hcAlarmFallingThreshAbsValueLo, hcAlarmFallingThreshAbsValueHi, and hcAlarmFallingThresholdValStatus objects.

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 8 }

hcAlarmRisingThreshAbsValueHi OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The upper 32 bits of the absolute value for threshold for the sampled statistic. The actual threshold value is determined by the associated instances of the hcAlarmRisingThreshAbsValueLo and hcAlarmRisingThresholdValStatus objects, as follows:

$$\text{ABS}(\text{threshold}) = \text{hcAlarmRisingThreshAbsValueLo} + (\text{hcAlarmRisingThreshAbsValueHi} * 2^{32})$$

The absolute value of the threshold is adjusted as required, as described in the HcValueStatus textual convention. These three object instances are conceptually combined to represent the rising threshold for this entry.

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 valid is greater than or equal to this threshold and the associated hcAlarmStartupAlarm is equal to risingAlarm(1) or risingOrFallingAlarm(3).

After a rising event is generated, another such event will not be generated until the sampled value falls below this threshold and reaches the threshold identified by the hcAlarmFallingThreshAbsValueLo, hcAlarmFallingThreshAbsValueHi, and hcAlarmFallingThresholdValStatus objects.

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 9 }

hcAlarmRisingThresholdValStatus OBJECT-TYPE

SYNTAX HcValueStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object indicates the sign of the data for the rising threshold, as defined by the hcAlarmRisingThreshAbsValueLo and hcAlarmRisingThreshAbsValueHi objects, as described in the HcValueStatus textual convention.

The enumeration 'valueNotAvailable(1)' is not allowed, and the associated hcAlarmStatus object cannot be equal to 'active(1)' if this object is set to this value.

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 10 }

hcAlarmFallingThreshAbsValueLo OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The lower 32 bits of the absolute value for threshold for the sampled statistic. The actual threshold value is determined by the associated instances of the hcAlarmFallingThreshAbsValueHi and hcAlarmFallingThresholdValStatus objects, as follows:

$$\text{ABS}(\text{threshold}) = \text{hcAlarmFallingThreshAbsValueLo} + (\text{hcAlarmFallingThreshAbsValueHi} * 2^{32})$$

The absolute value of the threshold is adjusted as required, as described in the HcValueStatus textual convention. These three object instances are conceptually combined to represent the falling threshold for this entry.

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 valid is less than or equal to this threshold and the associated hcAlarmStartupAlarm is equal to fallingAlarm(2) or risingOrFallingAlarm(3).

After a falling event is generated, another such event will not be generated until the sampled value rises above this threshold and reaches the threshold identified by the hcAlarmRisingThreshAbsValueLo, hcAlarmRisingThreshAbsValueHi, and hcAlarmRisingThresholdValStatus objects.

This object may not be modified if the associated
 hcAlarmStatus object is equal to active(1)."
 ::= { hcAlarmEntry 11 }

hcAlarmFallingThreshAbsValueHi OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The upper 32 bits of the absolute value for threshold for the sampled statistic. The actual threshold value is determined by the associated instances of the hcAlarmFallingThreshAbsValueLo and hcAlarmFallingThresholdValStatus objects, as follows:

$$\text{ABS}(\text{threshold}) = \text{hcAlarmFallingThreshAbsValueLo} + (\text{hcAlarmFallingThreshAbsValueHi} * 2^{32})$$

The absolute value of the threshold is adjusted as required, as described in the HcValueStatus textual convention. These three object instances are conceptually combined to represent the falling threshold for this entry.

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 valid is less than or equal to this threshold and the associated hcAlarmStartupAlarm is equal to fallingAlarm(2) or risingOrFallingAlarm(3).

After a falling event is generated, another such event will not be generated until the sampled value rises above this threshold and reaches the threshold identified by the hcAlarmRisingThreshAbsValueLo, hcAlarmRisingThreshAbsValueHi, and hcAlarmRisingThresholdValStatus objects.

This object may not be modified if the associated
 hcAlarmStatus object is equal to active(1)."
 ::= { hcAlarmEntry 12 }

hcAlarmFallingThresholdValStatus OBJECT-TYPE

SYNTAX HcValueStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object indicates the sign of the data for the falling threshold, as defined by the hcAlarmFallingThreshAbsValueLo and hcAlarmFallingThreshAbsValueHi objects, as described in the HcValueStatus textual convention.

The enumeration 'valueNotAvailable(1)' is not allowed, and the associated hcAlarmStatus object cannot be equal to 'active(1)' if this object is set to this value.

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 13 }

hcAlarmRisingEventIndex OBJECT-TYPE

SYNTAX Integer32 (0..65535)

MAX-ACCESS read-create

STATUS current

DESCRIPTION

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

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 14 }

hcAlarmFallingEventIndex OBJECT-TYPE

SYNTAX Integer32 (0..65535)

MAX-ACCESS read-create

STATUS current

DESCRIPTION

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

This object may not be modified if the associated hcAlarmStatus object is equal to active(1)."

::= { hcAlarmEntry 15 }

hcAlarmValueFailedAttempts OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The number of times the associated hcAlarmVariable instance was polled on behalf of this hcAlarmEntry, (while in the active state) and the value was not available. This counter may experience a discontinuity if the agent restarts, indicated by the value of sysUpTime."

::= { hcAlarmEntry 16 }

hcAlarmOwner OBJECT-TYPE
SYNTAX OwnerString
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"The entity that configured this entry and is therefore using the resources assigned to it."

::= { hcAlarmEntry 17 }

hcAlarmStorageType OBJECT-TYPE
SYNTAX StorageType
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"The type of non-volatile storage configured for this entry. If this object is equal to 'permanent(4)', then the associated hcAlarmRisingEventIndex and hcAlarmFallingEventIndex objects must be writable."

::= { hcAlarmEntry 18 }

hcAlarmStatus OBJECT-TYPE
SYNTAX RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"The status of this row.

An entry MUST NOT exist in the active state unless all objects in the entry have an appropriate value, as described in the description clause for each writable object.

The hcAlarmStatus object may be modified if the associated instance of this object is equal to active(1), notInService(2), or notReady(3). All other writable objects may be modified if the associated instance of this object is equal to notInService(2) or notReady(3)."

::= { hcAlarmEntry 19 }

```
--
-- Capabilities
--
```

```
hcAlarmCapabilities OBJECT-TYPE
```

```
    SYNTAX      BITS {
        hcAlarmCreation(0),
        hcAlarmNvStorage(1)
    }
```

```
    MAX-ACCESS  read-only
```

```
    STATUS      current
```

```
    DESCRIPTION
```

"An indication of the high capacity alarm capabilities supported by this agent.

If the 'hcAlarmCreation' BIT is set, then this agent allows NMS applications to create entries in the hcAlarmTable.

If the 'hcAlarmNvStorage' BIT is set, then this agent allows entries in the hcAlarmTable which will be recreated after a system restart, as controlled by the hcAlarmStorageType object."

```
 ::= { hcAlarmCapabilitiesObjects 1 }
```

```
--
-- Notifications
--
```

```
hcAlarmNotifPrefix OBJECT IDENTIFIER
```

```
 ::= { hcAlarmNotifications 0 }
```

```
hcRisingAlarm NOTIFICATION-TYPE
```

```
    OBJECTS { hcAlarmVariable,
        hcAlarmSampleType,
        hcAlarmAbsValue,
        hcAlarmValueStatus,
        hcAlarmRisingThreshAbsValueLo,
        hcAlarmRisingThreshAbsValueHi,
        hcAlarmRisingThresholdValStatus,
        hcAlarmRisingEventIndex }
```

```
    STATUS      current
```

```
    DESCRIPTION
```

"The SNMP notification that is generated when a high capacity alarm entry crosses its rising threshold and generates an event that is configured for sending SNMP traps.

The hcAlarmEntry object instances identified in the OBJECTS

```

        clause are from the entry that causes this notification to
        be generated."
 ::= { hcAlarmNotifPrefix 1 }

hcFallingAlarm NOTIFICATION-TYPE
  OBJECTS { hcAlarmVariable,
            hcAlarmSampleType,
            hcAlarmAbsValue,
            hcAlarmValueStatus,
            hcAlarmFallingThreshAbsValueLo,
            hcAlarmFallingThreshAbsValueHi,
            hcAlarmFallingThresholdValStatus,
            hcAlarmFallingEventIndex }
  STATUS current
  DESCRIPTION
    "The SNMP notification that is generated when a high
    capacity alarm entry crosses its falling threshold and
    generates an event that is configured for sending SNMP
    traps.

    The hcAlarmEntry object instances identified in the OBJECTS
    clause are from the entry that causes this notification to
    be generated."
 ::= { hcAlarmNotifPrefix 2 }

--
-- Conformance Section
--

hcAlarmCompliances OBJECT IDENTIFIER ::= { hcAlarmConformance 1 }
hcAlarmGroups      OBJECT IDENTIFIER ::= { hcAlarmConformance 2 }

hcAlarmCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "Describes the requirements for conformance to the High
    Capacity Alarm MIB."
  MODULE -- this module
    MANDATORY-GROUPS {
      hcAlarmControlGroup,
      hcAlarmCapabilitiesGroup,
      hcAlarmNotificationsGroup
    }

  MODULE RMON-MIB
    MANDATORY-GROUPS { rmonEventGroup }

  ::= { hcAlarmCompliances 1 }

```

-- Object Groups

hcAlarmControlGroup OBJECT-GROUP

```
OBJECTS {
    hcAlarmInterval,
    hcAlarmVariable,
    hcAlarmSampleType,
    hcAlarmAbsValue,
    hcAlarmValueStatus,
    hcAlarmStartupAlarm,
    hcAlarmRisingThreshAbsValueLo,
    hcAlarmRisingThreshAbsValueHi,
    hcAlarmRisingThresholdValStatus,
    hcAlarmFallingThreshAbsValueLo,
    hcAlarmFallingThreshAbsValueHi,
    hcAlarmFallingThresholdValStatus,
    hcAlarmRisingEventIndex,
    hcAlarmFallingEventIndex,
    hcAlarmValueFailedAttempts,
    hcAlarmOwner,
    hcAlarmStorageType,
    hcAlarmStatus
}
STATUS current
DESCRIPTION
    "A collection of objects used to configure entries for high
    capacity alarm threshold monitoring purposes."
 ::= { hcAlarmGroups 1 }
```

hcAlarmCapabilitiesGroup OBJECT-GROUP

```
OBJECTS {
    hcAlarmCapabilities
}
STATUS current
DESCRIPTION
    "A collection of objects used to indicate an agent's high
    capacity alarm threshold monitoring capabilities."
 ::= { hcAlarmGroups 2 }
```

hcAlarmNotificationsGroup NOTIFICATION-GROUP

```
NOTIFICATIONS {
    hcRisingAlarm,
    hcFallingAlarm
}
STATUS current
DESCRIPTION
    "A collection of notifications to deliver information
    related to a high capacity rising or falling threshold event"
```

```
        to a management application."  
 ::= { hcAlarmGroups 3 }
```

END

6. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

7. Acknowledgements

This memo is a product of the RMONMIB working group, and is based on existing alarmTable objects in the RMON-1 MIB module [RFC2819]. In order to maintain the RMON 'look-and-feel' and semantic consistency, some of Steve Waldbusser's text from [RFC2819] has been adapted for use in this MIB.

8. Normative References

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

- [RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Structure of Management Information Version 2 (SMIV2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Textual Conventions for SMIV2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIV2", RFC 2580, STD 58, April 1999.
- [RFC2819] Waldbusser, S., "Remote Network Monitoring Management Information Base", STD 59, RFC 2819, May 2000.
- [RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.
- [RFC3415] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.

9. Informative References

- [RFC3410] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June, 2000.

10. Security Considerations

There are a number of management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

There are a number of managed objects in this MIB that may contain sensitive information. These are:

- hcAlarmAbsValue
- hcAlarmValueStatus

These objects are used together, and may expose the values of particular MIB instances, as identified by associated instances of the hcAlarmVariable object.

hcAlarmVariable

This object identifies the object instance that the associated hcAlarmEntry will periodically sample. Because SNMP access control is articulated entirely in terms of the contents of MIB views, no access control mechanism exists that can restrict the value of this object to identify only those objects that exist in a particular MIB view. Thus, because there is no acceptable means of restricting the read access that could be obtained through the alarm mechanism, the probe must only grant write access to this object in those views that have read access to all objects on the probe.

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB.

It is recommended that the implementors consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model STD 62, RFC 3414 [RFC3414] and the View-based Access Control Model STD 62, RFC 3415 [RFC3415] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to only the objects, and to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

11. Authors' Addresses

Andy Bierman
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-3711
EMail: abierman@cisco.com

Keith McCloghrie
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-526-5260
EMail: kzm@cisco.com

12. Full Copyright Statement

Copyright (C) The Internet Society (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

