

Network Working Group
Request for Comments: 2981
Category: Standards Track

R. Kavasseri
(Editor of this version)
B. Stewart
(Author of previous version)
Cisco Systems, Inc.
October 2000

Event MIB

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 (2000). 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 that can be used to manage and monitor MIB objects and take action through events.

The Event MIB provides the ability to monitor MIB objects on the local system or on a remote system and take simple action when a trigger condition is met.

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 RFC 2119.

Table of Contents

1 The SNMP Management Framework	2
2 Overview	3
3 Relationship to Other MIBs	3
4 MIB Sections	4
5 Operation	5
6 Security	7
7 Definitions	7
8 Intellectual Property	47
9 Acknowledgements	47

10	References	47
11	Security Considerations	49
12	Author's Address	49
13	Editor's Address	49
14	Full Copyright Statement	50

1. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- o An overall architecture, described in RFC 2571 [RFC2571].
- o Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIV1 and described in STD 16, RFC 1155 [RFC1155], STD 16, RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIV2, is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].
- o Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].
- o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].
- o A set of fundamental applications described in RFC 2573 [RFC2573] and the view-based access control mechanism described in RFC 2575 [RFC2575].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [RFC2570].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIV2. A MIB conforming to the SMIV1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (use of Counter64). Some machine readable information in SMIV2 will be converted into textual descriptions in SMIV1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB. It may not be possible to meaningfully monitor Counter64 objects using an SMIV1 version of the MIB.

2. Overview

With network sizes well beyond the ability of people to manage them directly, automated, distributed management is vital. An important aspect of such management is the ability of a system to monitor itself or for some other system to monitor it.

The Event MIB provides the ability to monitor MIB objects on the local system or on a remote system and take simple action when a trigger condition is met.

The MIB is intended to suit either a relatively powerful manager or mid-level manager, as well as a somewhat more limited self-managing system.

3. Relationship to Other MIBs

The Event MIB is based on extensive experience with the RMON MIB [RFC1757] and provides a superset of the capabilities of the RMON alarm and event groups. Conceptually, the key extension is the ability to allow alarms to be generated for MIB objects that are on another network element. The Event MIB calls "triggers" what the RMON MIB called "alarms," but the concepts are the same. Event MIB triggers maintain the RMON handling of thresholds and add the concept of booleans. Event MIB events maintain the RMON concept of sending an SNMP notification in response to a trigger and add the concept of setting a MIB object.

The Event MIB is the successor and update to SNMPv2's Manager-to-Manager MIB [RFC1451] which was declared Historic pending this work.

The Event MIB depends on the services of the SNMPv3 Management Target and Notification MIBs [RFC2573].

The Event MIB is nicely complemented by the Distributed Management Expression MIB [RFC2982], which is the expected source of boolean objects to monitor. Note that there is considerable overlap between

the wildcard and delta sample capabilities of the Event and Expression MIBs. A carefully-planned implementation might well use common code to provide the overlapping functions.

4. MIB Sections

The MIB has four sections: triggers, objects, events, and notifications. Triggers define the conditions that lead to events. Events may cause notifications.

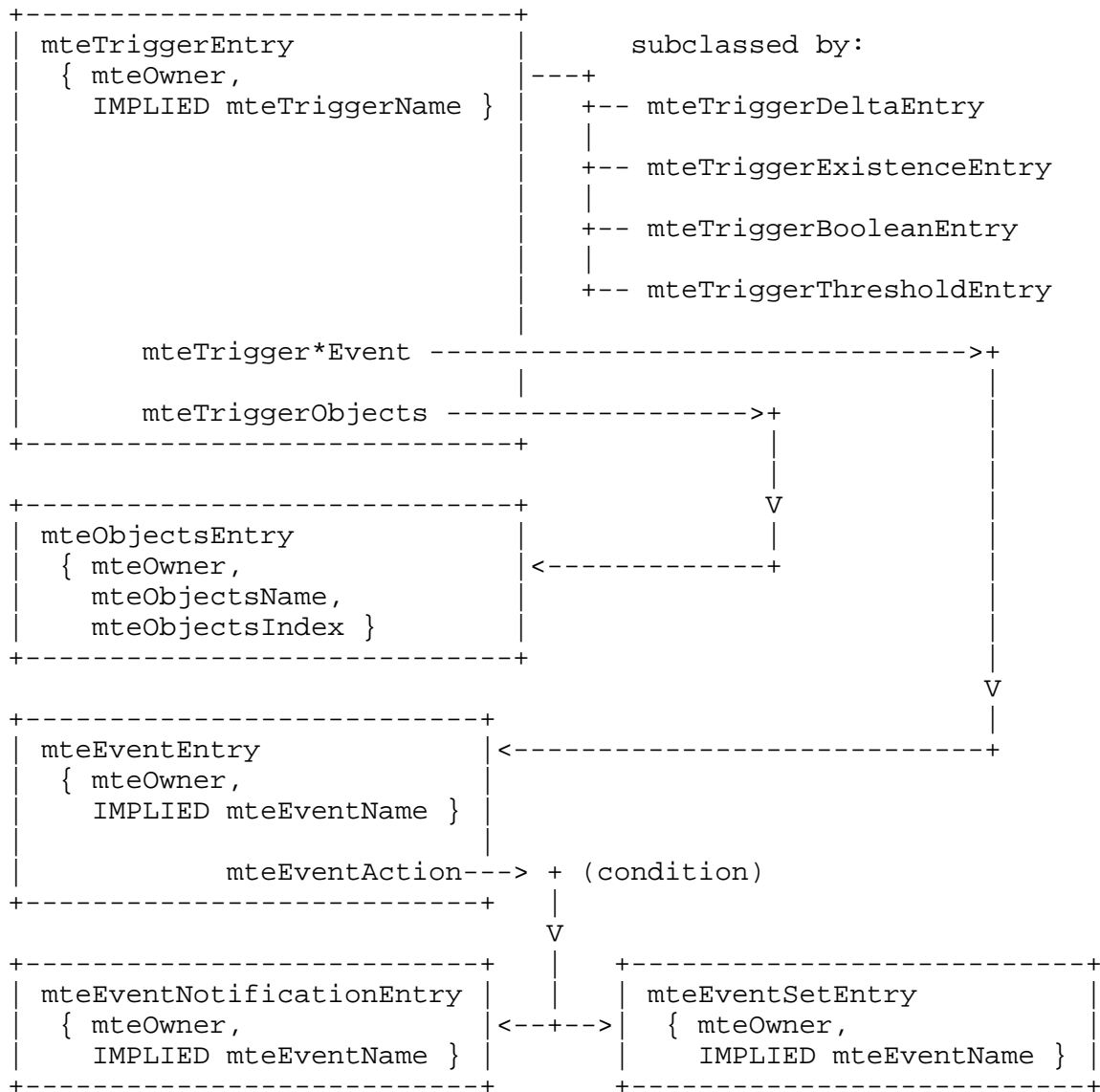
The trigger table lists what objects are to be monitored and how and relates each trigger to an event. It has supplementary, companion tables for additional objects that depend on the type of test done for the trigger.

The objects table lists objects that can be added to notifications based on the trigger, the trigger test type, or the event that resulted in the notification.

The event table defines what happens when an event is triggered: sending a notification, setting a MIB object or both. It has supplementary, companion tables for additional objects that depend on the action taken.

The notification section defines a set of generic notifications to go with the events and for Event MIB error handling, and it defines a set of objects to put in those notifications.

The following diagram describes the relationships between the tables in the Event MIB.



5. Operation

The Event MIB is instrumentation for a distributed management application that monitors MIB objects. In its simplest form this application monitors individual, local MIB objects, just as an RMON probe fulfills the functions implied by RMON's alarm and event operation. Additionally the application can monitor remote objects and wildcarded groups of objects.

Remote monitoring uses the tag service of the Management Target MIB [RFC2573] to select and access remote systems as an ordinary SNMP-based management application. Local monitoring may be via a more intimate, local interface which may, for example, bypass SNMP encoding but otherwise is functionally identical to remote SNMP operation, including the application of access control. A self-management only system MAY not implement remote monitoring.

Wildcards indicate that the application SHOULD use a GetNext-type operation to find the zero or more instances implied by a truncated object identifier, just like an ordinary SNMP-based management application. Each instance of a wildcard is treated as if it were a separate entry, that is the instances of a wildcarded object are independent of one another. For example, a wild-carded object may trigger an event, and result in the setting of another wildcarded object. The instance that satisfied the trigger function is used to perform the set function. All of this takes place independently of any additional instances that may fill the wildcard.

Error handling is by notification. These error notifications SHOULD be enabled only for the diagnosis of problems indicated by error counters. If minimizing the probability of notification loss is a concern they SHOULD be transmitted as Inform PDUs as described in the [SNMP-TARGET-MIB] or directed to a log as described in the Notification Log MIB [rfcNotificationLogMIB]. Note that this does not mean the Notification Log MIB is REQUIRED, since in fact notifications usually are not lost, but that the Notification Log MIB can be helpful with this as well as other MIBs that include notifications.

Although like most MIBs this one has no explicit controls for the persistence of the values set in configuring events, a robust, polite implementation would certainly not force its managing applications to reconfigure it whenever it resets.

Again, as with most MIBs, it is implementation-specific how a system provides and manages such persistence. To speculate, one could imagine, for example, that persistence depended on the context in which the expression was configured, or perhaps system-specific characteristics of the expression's owner. Or perhaps everything in a MIB such as this one, which is clearly aimed at persistent configuration, is automatically part of a system's other persistent configuration.

6. Security

Security of Event MIB entries depends on SNMPv3 access control for the entire MIB or for subsets based on entry owner names.

Security of monitored objects for remote access depends on the Management Target MIB [RFC2573]. Security for local access can depend on the Management Target MIB or on recording appropriate security credentials of the creator of an entry and using those to access the local objects. These security credentials are the parameters necessary as inputs to `isAccessAllowed` from the Architecture for Describing SNMP Management Frameworks. When accessing local objects without using a local target tag, the system MUST (conceptually) use `isAccessAllowed` to ensure that it does not violate security.

To facilitate the provisioning of access control by a security administrator for this MIB itself using the View-Based Access Control Model (VACM) defined in RFC 2275 [RFC2575] for tables in which multiple users may need to independently create or modify entries, the initial index is used as an "owner index". Such an initial index has a syntax of `SnmpAdminString`, and can thus be trivially mapped to a `securityName` or `groupName` as defined in VACM, in accordance with a security policy.

If a security administrator were to employ such an approach, all entries in related tables belonging to a particular user will have the same value for this initial index. For a given user's entries in a particular table, the object identifiers for the information in these entries will have the same sub-identifiers (except for the "column" sub-identifier) up to the end of the encoded owner index. To configure VACM to permit access to this portion of the table, one would create `vacmViewTreeFamilyTable` entries with the value of `vacmViewTreeFamilySubtree` including the owner index portion, and `vacmViewTreeFamilyMask` "wildcarding" the column sub-identifier. More elaborate configurations are possible.

7. Definitions

DISMAN-EVENT-MIB DEFINITIONS ::= BEGIN

IMPORTS

```
    MODULE-IDENTITY, OBJECT-TYPE,
    Integer32, Unsigned32,
    NOTIFICATION-TYPE, Counter32,
    Gauge32, mib-2, zeroDotZero          FROM SNMPv2-SMI
    TEXTUAL-CONVENTION, RowStatus,
    TruthValue                           FROM SNMPv2-TC
```

```

MODULE-COMPLIANCE, OBJECT-GROUP,
NOTIFICATION-GROUP          FROM SNMPv2-CONF
sysUpTime                   FROM SNMPv2-MIB
SnmpTagValue                FROM SNMP-TARGET-MIB
SnmpAdminString             FROM SNMP-FRAMEWORK-MIB;

```

```

dismanEventMIB MODULE-IDENTITY

```

```

    LAST-UPDATED "200010160000Z"          -- 16 October 2000

```

```

    ORGANIZATION "IETF Distributed Management Working Group"

```

```

    CONTACT-INFO "Ramanathan Kavasseri
                  Cisco Systems, Inc.
                  170 West Tasman Drive,
                  San Jose CA 95134-1706.
                  Phone: +1 408 526 4527
                  Email: ramk@cisco.com"

```

```

    DESCRIPTION

```

```

        "The MIB module for defining event triggers and actions
        for network management purposes."

```

```

-- Revision History

```

```

    REVISION      "200010160000Z"          -- 16 October 2000

```

```

    DESCRIPTION   "This is the initial version of this MIB.
                  Published as RFC 2981"

```

```

    ::= { mib-2 88 }

```

```

dismanEventMIBObjects OBJECT IDENTIFIER ::= { dismanEventMIB 1 }

```

```

-- Management Triggered Event (MTE) objects

```

```

mteResource      OBJECT IDENTIFIER ::= { dismanEventMIBObjects 1 }
mteTrigger       OBJECT IDENTIFIER ::= { dismanEventMIBObjects 2 }
mteObjects       OBJECT IDENTIFIER ::= { dismanEventMIBObjects 3 }
mteEvent         OBJECT IDENTIFIER ::= { dismanEventMIBObjects 4 }

```

```

--

```

```

-- Textual Conventions

```

```

--

```

```

FailureReason ::= TEXTUAL-CONVENTION

```

```

    STATUS      current

```

```

    DESCRIPTION

```

```

        "Reasons for failures in an attempt to perform a management
        request."

```

```

        The first group of errors, numbered less than 0, are related
        to problems in sending the request.  The existence of a
        particular error code here does not imply that all
        implementations are capable of sensing that error and

```


returning that code.

The second group, numbered greater than 0, are copied directly from SNMP protocol operations and are intended to carry exactly the meanings defined for the protocol as returned in an SNMP response.

localResourceLack	some local resource such as memory lacking or mteResourceSampleInstanceMaximum exceeded
badDestination	unrecognized domain name or otherwise invalid destination address
destinationUnreachable	can't get to destination address
noResponse	no response to SNMP request
badType	the data syntax of a retrieved object as not as expected
sampleOverrun	another sample attempt occurred before the previous one completed"

```
SYNTAX      INTEGER { localResourceLack(-1),
                        badDestination(-2),
                        destinationUnreachable(-3),
                        noResponse(-4),
                        badType(-5),
                        sampleOverrun(-6),

                        noError(0),

                        tooBig(1),
                        noSuchName(2),
                        badValue(3),
                        readOnly(4),
                        genErr(5),
                        noAccess(6),
                        wrongType(7),
                        wrongLength(8),
                        wrongEncoding(9),
                        wrongValue(10),
                        noCreation(11),
                        inconsistentValue(12),
                        resourceUnavailable(13),
                        commitFailed(14),
                        undoFailed(15),
                        authorizationError(16),
                        notWritable(17),
                        inconsistentName(18) }
```

--

-- Resource Control Section
--

mteResourceSampleMinimum OBJECT-TYPE

SYNTAX Integer32 (1..2147483647)

UNITS "seconds"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The minimum mteTriggerFrequency this system will accept. A system may use the larger values of this minimum to lessen the impact of constant sampling. For larger sampling intervals the system samples less often and suffers less overhead. This object provides a way to enforce such lower overhead for all triggers created after it is set.

Unless explicitly resource limited, a system's value for this object SHOULD be 1, allowing as small as a 1 second interval for ongoing trigger sampling.

Changing this value will not invalidate an existing setting of mteTriggerFrequency."

::= { mteResource 1 }

mteResourceSampleInstanceMaximum OBJECT-TYPE

SYNTAX Unsigned32

UNITS "instances"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The maximum number of instance entries this system will support for sampling.

These are the entries that maintain state, one for each instance of each sampled object as selected by mteTriggerValueID. Note that wildcarded objects result in multiple instances of this state.

A value of 0 indicates no preset limit, that is, the limit is dynamic based on system operation and resources.

Unless explicitly resource limited, a system's value for this object SHOULD be 0.

Changing this value will not eliminate or inhibit existing sample state but could prevent allocation of additional state information."

```
::= { mteResource 2 }
```

```
mteResourceSampleInstances OBJECT-TYPE
```

```
SYNTAX      Gauge32
```

```
UNITS       "instances"
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
    "The number of currently active instance entries as  
    defined for mteResourceSampleInstanceMaximum."
```

```
::= { mteResource 3 }
```

```
mteResourceSampleInstancesHigh OBJECT-TYPE
```

```
SYNTAX      Gauge32
```

```
UNITS       "instances"
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
    "The highest value of mteResourceSampleInstances that has  
    occurred since initialization of the management system."
```

```
::= { mteResource 4 }
```

```
mteResourceSampleInstanceLacks OBJECT-TYPE
```

```
SYNTAX      Counter32
```

```
UNITS       "instances"
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
    "The number of times this system could not take a new sample  
    because that allocation would have exceeded the limit set by  
    mteResourceSampleInstanceMaximum."
```

```
::= { mteResource 5 }
```

```
--
```

```
-- Trigger Section
```

```
--
```

```
-- Counters
```

```
mteTriggerFailures OBJECT-TYPE
```

```
SYNTAX      Counter32
```

```
UNITS       "failures"
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
    "The number of times an attempt to check for a trigger  
    condition has failed.  This counts individually for each  
    attempt in a group of targets or each attempt for a
```

```

        wildcarded object."
 ::= { mteTrigger 1 }

--
-- Trigger Table
--

mteTriggerTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF MteTriggerEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of management event trigger information."
    ::= { mteTrigger 2 }

mteTriggerEntry OBJECT-TYPE
    SYNTAX      MteTriggerEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Information about a single trigger. Applications create and
        delete entries using mteTriggerEntryStatus."
    INDEX       { mteOwner, IMPLIED mteTriggerName }
    ::= { mteTriggerTable 1 }

MteTriggerEntry ::= SEQUENCE {
    mteOwner                SnmpAdminString,
    mteTriggerName          SnmpAdminString,
    mteTriggerComment       SnmpAdminString,
    mteTriggerTest          BITS,
    mteTriggerSampleType    INTEGER,
    mteTriggerValueID       OBJECT IDENTIFIER,
    mteTriggerValueIDWildcard TruthValue,
    mteTriggerTargetTag     SnmpTagValue,
    mteTriggerContextName   SnmpAdminString,
    mteTriggerContextNameWildcard TruthValue,
    mteTriggerFrequency     Unsigned32,
    mteTriggerObjectsOwner  SnmpAdminString,
    mteTriggerObjects       SnmpAdminString,
    mteTriggerEnabled       TruthValue,
    mteTriggerEntryStatus   RowStatus
}

mteOwner OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION

```

"The owner of this entry. The exact semantics of this string are subject to the security policy defined by the security administrator."

::= { mteTriggerEntry 1 }

mteTriggerName OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (1..32))

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A locally-unique, administratively assigned name for the trigger within the scope of mteOwner."

::= { mteTriggerEntry 2 }

mteTriggerComment OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"A description of the trigger's function and use."

DEFVAL { ''H }

::= { mteTriggerEntry 3 }

mteTriggerTest OBJECT-TYPE

SYNTAX BITS { existence(0), boolean(1), threshold(2) }

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The type of trigger test to perform. For 'boolean' and 'threshold' tests, the object at mteTriggerValueID MUST evaluate to an integer, that is, anything that ends up encoded for transmission (that is, in BER, not ASN.1) as an integer.

For 'existence', the specific test is as selected by mteTriggerExistenceTest. When an object appears, vanishes or changes value, the trigger fires. If the object's appearance caused the trigger firing, the object MUST vanish before the trigger can be fired again for it, and vice versa. If the trigger fired due to a change in the object's value, it will be fired again on every successive value change for that object.

For 'boolean', the specific test is as selected by mteTriggerBooleanTest. If the test result is true the trigger fires. The trigger will not fire again until the value has become false and come back to true.

For 'threshold' the test works as described below for

mteTriggerThresholdStartup, mteTriggerThresholdRising, and mteTriggerThresholdFalling.

Note that combining 'boolean' and 'threshold' tests on the same object may be somewhat redundant."

```
DEFVAL { { boolean } }  
::= { mteTriggerEntry 4 }
```

mteTriggerSampleType OBJECT-TYPE

```
SYNTAX      INTEGER { absoluteValue(1), deltaValue(2) }  
MAX-ACCESS  read-create  
STATUS      current  
DESCRIPTION
```

"The type of sampling to perform.

An 'absoluteValue' sample requires only a single sample to be meaningful, and is exactly the value of the object at mteTriggerValueID at the sample time.

A 'deltaValue' requires two samples to be meaningful and is thus not available for testing until the second and subsequent samples after the object at mteTriggerValueID is first found to exist. It is the difference between the two samples. For unsigned values it is always positive, based on unsigned arithmetic. For signed values it can be positive or negative.

For SNMP counters to be meaningful they should be sampled as a 'deltaValue'.

For 'deltaValue' mteTriggerDeltaTable contains further parameters.

If only 'existence' is set in mteTriggerTest this object has no meaning."

```
DEFVAL { absoluteValue }  
::= { mteTriggerEntry 5 }
```

mteTriggerValueID OBJECT-TYPE

```
SYNTAX      OBJECT IDENTIFIER  
MAX-ACCESS  read-create  
STATUS      current  
DESCRIPTION
```

"The object identifier of the MIB object to sample to see if the trigger should fire.

This may be wildcarded by truncating all or part of the instance portion, in which case the value is obtained as if with a GetNext function, checking multiple values

if they exist. If such wildcarding is applied, mteTriggerValueIDWildcard must be 'true' and if not it must be 'false'.

Bad object identifiers or a mismatch between truncating the identifier and the value of mteTriggerValueIDWildcard result in operation as one would expect when providing the wrong identifier to a Get or GetNext operation. The Get will fail or get the wrong object. The GetNext will indeed get whatever is next, proceeding until it runs past the initial part of the identifier and perhaps many unintended objects for confusing results. If the value syntax of those objects is not usable, that results in a 'badType' error that terminates the scan.

Each instance that fills the wildcard is independent of any additional instances, that is, wildcarded objects operate as if there were a separate table entry for each instance that fills the wildcard without having to actually predict all possible instances ahead of time."

```
DEFVAL { zeroDotZero }
::= { mteTriggerEntry 6 }
```

mteTriggerValueIDWildcard OBJECT-TYPE

```
SYNTAX      TruthValue
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "Control for whether mteTriggerValueID is to be treated as
    fully-specified or wildcarded, with 'true' indicating wildcard."
DEFVAL { false }
::= { mteTriggerEntry 7 }
```

mteTriggerTargetTag OBJECT-TYPE

```
SYNTAX      SnmpTagValue
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The tag for the target(s) from which to obtain the condition
    for a trigger check."
```

A length of 0 indicates the local system. In this case, access to the objects indicated by mteTriggerValueID is under the security credentials of the requester that set mteTriggerEntryStatus to 'active'. Those credentials are the input parameters for isAccessAllowed from the Architecture for Describing SNMP Management Frameworks.

Otherwise access rights are checked according to the security

parameters resulting from the tag."
 DEFVAL { ''H }
 ::= { mteTriggerEntry 8 }

mteTriggerContextName OBJECT-TYPE

SYNTAX SnmpAdminString
 MAX-ACCESS read-create
 STATUS current
 DESCRIPTION

"The management context from which to obtain mteTriggerValueID.

This may be wildcarded by leaving characters off the end. For example use 'Repeater' to wildcard to 'Repeater1', 'Repeater2', 'Repeater-999.87b', and so on. To indicate such wildcarding is intended, mteTriggerContextNameWildcard must be 'true'.

Each instance that fills the wildcard is independent of any additional instances, that is, wildcarded objects operate as if there were a separate table entry for each instance that fills the wildcard without having to actually predict all possible instances ahead of time.

Operation of this feature assumes that the local system has a list of available contexts against which to apply the wildcard. If the objects are being read from the local system, this is clearly the system's own list of contexts. For a remote system a local version of such a list is not defined by any current standard and may not be available, so this function MAY not be supported."

DEFVAL { ''H }
 ::= { mteTriggerEntry 9 }

mteTriggerContextNameWildcard OBJECT-TYPE

SYNTAX TruthValue
 MAX-ACCESS read-create
 STATUS current
 DESCRIPTION

"Control for whether mteTriggerContextName is to be treated as fully-specified or wildcarded, with 'true' indicating wildcard."

DEFVAL { false }
 ::= { mteTriggerEntry 10 }

mteTriggerFrequency OBJECT-TYPE

SYNTAX Unsigned32
 UNITS "seconds"
 MAX-ACCESS read-create
 STATUS current

DESCRIPTION

"The number of seconds to wait between trigger samples. To encourage consistency in sampling, the interval is measured from the beginning of one check to the beginning of the next and the timer is restarted immediately when it expires, not when the check completes.

If the next sample begins before the previous one completed the system may either attempt to make the check or treat this as an error condition with the error 'sampleOverrun'.

A frequency of 0 indicates instantaneous recognition of the condition. This is not possible in many cases, but may be supported in cases where it makes sense and the system is able to do so. This feature allows the MIB to be used in implementations where such interrupt-driven behavior is possible and is not likely to be supported for all MIB objects even then since such sampling generally has to be tightly integrated into low-level code.

Systems that can support this SHOULD document those cases where it can be used. In cases where it can not, setting this object to 0 should be disallowed."

```
DEFVAL { 600 }
 ::= { mteTriggerEntry 11 }
```

mteTriggerObjectsOwner OBJECT-TYPE

```
SYNTAX      SnmpAdminString (SIZE (0..32))
MAX-ACCESS  read-create
STATUS      current
```

DESCRIPTION

"To go with mteTriggerObjects, the mteOwner of a group of objects from mteObjectsTable."

```
DEFVAL { ''H }
 ::= { mteTriggerEntry 12 }
```

mteTriggerObjects OBJECT-TYPE

```
SYNTAX      SnmpAdminString (SIZE (0..32))
MAX-ACCESS  read-create
STATUS      current
```

DESCRIPTION

"The mteObjectsName of a group of objects from mteObjectsTable. These objects are to be added to any Notification resulting from the firing of this trigger.

A list of objects may also be added based on the event or on the value of mteTriggerTest.

A length of 0 indicates no additional objects."
 DEFVAL { ''H }
 ::= { mteTriggerEntry 13 }

mteTriggerEnabled OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"A control to allow a trigger to be configured but not used.

When the value is 'false' the trigger is not sampled."

DEFVAL { false }

::= { mteTriggerEntry 14 }

mteTriggerEntryStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The control that allows creation and deletion of entries.

Once made active an entry may not be modified except to delete it."

::= { mteTriggerEntry 15 }

--

-- Trigger Delta Table

--

mteTriggerDeltaTable OBJECT-TYPE

SYNTAX SEQUENCE OF MteTriggerDeltaEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A table of management event trigger information for delta sampling."

::= { mteTrigger 3 }

mteTriggerDeltaEntry OBJECT-TYPE

SYNTAX MteTriggerDeltaEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Information about a single trigger's delta sampling. Entries automatically exist in this table for each mteTriggerEntry that has mteTriggerSampleType set to 'deltaValue'."

INDEX { mteOwner, IMPLIED mteTriggerName }

::= { mteTriggerDeltaTable 1 }

```

MteTriggerDeltaEntry ::= SEQUENCE {
    mteTriggerDeltaDiscontinuityID          OBJECT IDENTIFIER,
    mteTriggerDeltaDiscontinuityIDWildcard TruthValue,
    mteTriggerDeltaDiscontinuityIDType     INTEGER
}

```

```

sysUpTimeInstance OBJECT IDENTIFIER ::= { sysUpTime 0 }

```

```

mteTriggerDeltaDiscontinuityID OBJECT-TYPE

```

```

    SYNTAX          OBJECT IDENTIFIER

```

```

    MAX-ACCESS      read-write

```

```

    STATUS          current

```

```

    DESCRIPTION

```

"The OBJECT IDENTIFIER (OID) of a TimeTicks, TimeStamp, or DateAndTime object that indicates a discontinuity in the value at mteTriggerValueID.

The OID may be for a leaf object (e.g. sysUpTime.0) or may be wildcarded to match mteTriggerValueID.

This object supports normal checking for a discontinuity in a counter. Note that if this object does not point to sysUpTime discontinuity checking MUST still check sysUpTime for an overall discontinuity.

If the object identified is not accessible the sample attempt is in error, with the error code as from an SNMP request.

Bad object identifiers or a mismatch between truncating the identifier and the value of mteDeltaDiscontinuityIDWildcard result in operation as one would expect when providing the wrong identifier to a Get operation. The Get will fail or get the wrong object. If the value syntax of those objects is not usable, that results in an error that terminates the sample with a 'badType' error code."

```

    DEFVAL { sysUpTimeInstance }

```

```

    ::= { mteTriggerDeltaEntry 1 }

```

```

mteTriggerDeltaDiscontinuityIDWildcard OBJECT-TYPE

```

```

    SYNTAX          TruthValue

```

```

    MAX-ACCESS      read-write

```

```

    STATUS          current

```

```

    DESCRIPTION

```

"Control for whether mteTriggerDeltaDiscontinuityID is to be treated as fully-specified or wildcarded, with 'true' indicating wildcard. Note that the value of this object will be the same as that of the corresponding instance of mteTriggerValueIDWildcard when the corresponding

```

        mteTriggerSampleType is 'deltaValue'."
DEFVAL { false }
 ::= { mteTriggerDeltaEntry 2 }

mteTriggerDeltaDiscontinuityIDType OBJECT-TYPE
    SYNTAX      INTEGER { timeTicks(1), timeStamp(2), dateAndTime(3) }
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "The value 'timeTicks' indicates the
         mteTriggerDeltaDiscontinuityID of this row is of syntax
         TimeTicks. The value 'timeStamp' indicates syntax TimeStamp.
         The value 'dateAndTime' indicates syntax DateAndTime."
    DEFVAL { timeTicks }
    ::= { mteTriggerDeltaEntry 3 }

--
-- Trigger Existence Table
--

mteTriggerExistenceTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF MteTriggerExistenceEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "A table of management event trigger information for existence
         triggers."
    ::= { mteTrigger 4 }

mteTriggerExistenceEntry OBJECT-TYPE
    SYNTAX      MteTriggerExistenceEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "Information about a single existence trigger. Entries
         automatically exist in this table for each mteTriggerEntry
         that has 'existence' set in mteTriggerTest."
    INDEX       { mteOwner, IMPLIED mteTriggerName }
    ::= { mteTriggerExistenceTable 1 }

MteTriggerExistenceEntry ::= SEQUENCE {
    mteTriggerExistenceTest          BITS,
    mteTriggerExistenceStartup       BITS,
    mteTriggerExistenceObjectsOwner  SnmpAdminString,
    mteTriggerExistenceObjects       SnmpAdminString,
    mteTriggerExistenceEventOwner    SnmpAdminString,
    mteTriggerExistenceEvent         SnmpAdminString
}

```

mteTriggerExistenceTest OBJECT-TYPE

SYNTAX BITS { present(0), absent(1), changed(2) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The type of existence test to perform. The trigger fires when the object at mteTriggerValueID is seen to go from present to absent, from absent to present, or to have it's value changed, depending on which tests are selected:

present(0) - when this test is selected, the trigger fires when the mteTriggerValueID object goes from absent to present.

absent(1) - when this test is selected, the trigger fires when the mteTriggerValueID object goes from present to absent.

changed(2) - when this test is selected, the trigger fires the mteTriggerValueID object value changes.

Once the trigger has fired for either presence or absence it will not fire again for that state until the object has been to the other state. "

DEFVAL { { present, absent } }

::= { mteTriggerExistenceEntry 1 }

mteTriggerExistenceStartup OBJECT-TYPE

SYNTAX BITS { present(0), absent(1) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Control for whether an event may be triggered when this entry is first set to 'active' and the test specified by mteTriggerExistenceTest is true. Setting an option causes that trigger to fire when its test is true."

DEFVAL { { present, absent } }

::= { mteTriggerExistenceEntry 2 }

mteTriggerExistenceObjectsOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"To go with mteTriggerExistenceObjects, the mteOwner of a group of objects from mteObjectsTable."

DEFVAL { ''H }

::= { mteTriggerExistenceEntry 3 }

mteTriggerExistenceObjects OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write
 STATUS current
 DESCRIPTION

"The mteObjectsName of a group of objects from mteObjectsTable. These objects are to be added to any Notification resulting from the firing of this trigger for this test.

A list of objects may also be added based on the overall trigger, the event or other settings in mteTriggerTest.

A length of 0 indicates no additional objects."

DEFVAL { ''H }
 ::= { mteTriggerExistenceEntry 4 }

mteTriggerExistenceEventOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))
 MAX-ACCESS read-write
 STATUS current
 DESCRIPTION

"To go with mteTriggerExistenceEvent, the mteOwner of an event entry from the mteEventTable."

DEFVAL { ''H }
 ::= { mteTriggerExistenceEntry 5 }

mteTriggerExistenceEvent OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))
 MAX-ACCESS read-write
 STATUS current
 DESCRIPTION

"The mteEventName of the event to invoke when mteTriggerType is 'existence' and this trigger fires. A length of 0 indicates no event."

DEFVAL { ''H }
 ::= { mteTriggerExistenceEntry 6 }

--

-- Trigger Boolean Table

--

mteTriggerBooleanTable OBJECT-TYPE

SYNTAX SEQUENCE OF MteTriggerBooleanEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION

"A table of management event trigger information for boolean triggers."

::= { mteTrigger 5 }

```

mteTriggerBooleanEntry OBJECT-TYPE
    SYNTAX      MteTriggerBooleanEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Information about a single boolean trigger.  Entries
        automatically exist in this table for each mteTriggerEntry
        that has 'boolean' set in mteTriggerTest."
    INDEX       { mteOwner, IMPLIED mteTriggerName }
    ::= { mteTriggerBooleanTable 1 }

```

```

MteTriggerBooleanEntry ::= SEQUENCE {
    mteTriggerBooleanComparison      INTEGER,
    mteTriggerBooleanValue           Integer32,
    mteTriggerBooleanStartup         TruthValue,
    mteTriggerBooleanObjectsOwner    SnmpAdminString,
    mteTriggerBooleanObjects         SnmpAdminString,
    mteTriggerBooleanEventOwner      SnmpAdminString,
    mteTriggerBooleanEvent           SnmpAdminString
}

```

```

mteTriggerBooleanComparison OBJECT-TYPE
    SYNTAX      INTEGER { unequal(1), equal(2),
                        less(3), lessOrEqual(4),
                        greater(5), greaterOrEqual(6) }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The type of boolean comparison to perform.

        The value at mteTriggerValueID is compared to
        mteTriggerBooleanValue, so for example if
        mteTriggerBooleanComparison is 'less' the result would be true
        if the value at mteTriggerValueID is less than the value of
        mteTriggerBooleanValue."
    DEFVAL { unequal }
    ::= { mteTriggerBooleanEntry 1 }

```

```

mteTriggerBooleanValue OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value to use for the test specified by
        mteTriggerBooleanTest."
    DEFVAL { 0 }
    ::= { mteTriggerBooleanEntry 2 }

```

mteTriggerBooleanStartup OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Control for whether an event may be triggered when this entry is first set to 'active' or a new instance of the object at mteTriggerValueID is found and the test specified by mteTriggerBooleanComparison is true. In that case an event is triggered if mteTriggerBooleanStartup is 'true'."

DEFVAL { true }

::= { mteTriggerBooleanEntry 3 }

mteTriggerBooleanObjectsOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"To go with mteTriggerBooleanObjects, the mteOwner of a group of objects from mteObjectsTable."

DEFVAL { ''H }

::= { mteTriggerBooleanEntry 4 }

mteTriggerBooleanObjects OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The mteObjectsName of a group of objects from mteObjectsTable. These objects are to be added to any Notification resulting from the firing of this trigger for this test."

A list of objects may also be added based on the overall trigger, the event or other settings in mteTriggerTest.

A length of 0 indicates no additional objects."

DEFVAL { ''H }

::= { mteTriggerBooleanEntry 5 }

mteTriggerBooleanEventOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"To go with mteTriggerBooleanEvent, the mteOwner of an event entry from mteEventTable."

DEFVAL { ''H }


```
::= { mteTriggerBooleanEntry 6 }
```

```
mteTriggerBooleanEvent OBJECT-TYPE
```

```
SYNTAX      SnmpAdminString (SIZE (0..32))
```

```
MAX-ACCESS  read-write
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The mteEventName of the event to invoke when mteTriggerType is
'boolean' and this trigger fires. A length of 0 indicates no
event."
```

```
DEFVAL { ''H }
```

```
::= { mteTriggerBooleanEntry 7 }
```

```
--
```

```
-- Trigger Threshold Table
```

```
--
```

```
mteTriggerThresholdTable OBJECT-TYPE
```

```
SYNTAX      SEQUENCE OF MteTriggerThresholdEntry
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
"A table of management event trigger information for threshold
triggers."
```

```
::= { mteTrigger 6 }
```

```
mteTriggerThresholdEntry OBJECT-TYPE
```

```
SYNTAX      MteTriggerThresholdEntry
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
"Information about a single threshold trigger. Entries
automatically exist in this table for each mteTriggerEntry
that has 'threshold' set in mteTriggerTest."
```

```
INDEX      { mteOwner, IMPLIED mteTriggerName }
```

```
::= { mteTriggerThresholdTable 1 }
```

```
MteTriggerThresholdEntry ::= SEQUENCE {
```

```
    mteTriggerThresholdStartup
```

```
    INTEGER,
```

```
    mteTriggerThresholdRising
```

```
    Integer32,
```

```
    mteTriggerThresholdFalling
```

```
    Integer32,
```

```
    mteTriggerThresholdDeltaRising
```

```
    Integer32,
```

```
    mteTriggerThresholdDeltaFalling
```

```
    Integer32,
```

```
    mteTriggerThresholdObjectsOwner
```

```
    SnmpAdminString,
```

```
    mteTriggerThresholdObjects
```

```
    SnmpAdminString,
```

```
    mteTriggerThresholdRisingEventOwner
```

```
    SnmpAdminString,
```

```
    mteTriggerThresholdRisingEvent
```

```
    SnmpAdminString,
```

```
    mteTriggerThresholdFallingEventOwner
```

```
    SnmpAdminString,
```

```

mteTriggerThresholdFallingEvent          SnmpAdminString,
mteTriggerThresholdDeltaRisingEventOwner  SnmpAdminString,
mteTriggerThresholdDeltaRisingEvent       SnmpAdminString,
mteTriggerThresholdDeltaFallingEventOwner SnmpAdminString,
mteTriggerThresholdDeltaFallingEvent      SnmpAdminString
}

mteTriggerThresholdStartup OBJECT-TYPE
    SYNTAX      INTEGER { rising(1), falling(2), risingOrFalling(3) }
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "The event that may be triggered when this entry is first
        set to 'active' and a new instance of the object at
        mteTriggerValueID is found.  If the first sample after this
        instance becomes active is greater than or equal to
        mteTriggerThresholdRising and mteTriggerThresholdStartup is
        equal to 'rising' or 'risingOrFalling', then one
        mteTriggerThresholdRisingEvent is triggered for that instance.
        If the first sample after this entry becomes active is less
        than or equal to mteTriggerThresholdFalling and
        mteTriggerThresholdStartup is equal to 'falling' or
        'risingOrFalling', then one mteTriggerThresholdRisingEvent is
        triggered for that instance."
    DEFVAL { risingOrFalling }
    ::= { mteTriggerThresholdEntry 1 }

mteTriggerThresholdRising OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "A threshold value to check against if mteTriggerType is
        'threshold'.

        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, one
        mteTriggerThresholdRisingEvent is triggered.  That event is
        also triggered if the first sample after this entry becomes
        active is greater than or equal to this threshold and
        mteTriggerThresholdStartup is equal to 'rising' or
        'risingOrFalling'.

        After a rising event is generated, another such event is not
        triggered until the sampled value falls below this threshold
        and reaches mteTriggerThresholdFalling."
    DEFVAL { 0 }

```

```
::= { mteTriggerThresholdEntry 2 }
```

mteTriggerThresholdFalling OBJECT-TYPE

```
SYNTAX      Integer32
MAX-ACCESS  read-write
STATUS      current
```

DESCRIPTION

"A threshold value to check against if mteTriggerType is 'threshold'.

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, one mteTriggerThresholdFallingEvent is triggered. That event is also triggered if the first sample after this entry becomes active is less than or equal to this threshold and mteTriggerThresholdStartup is equal to 'falling' or 'risingOrFalling'.

After a falling event is generated, another such event is not triggered until the sampled value rises above this threshold and reaches mteTriggerThresholdRising."

```
DEFVAL { 0 }
```

```
::= { mteTriggerThresholdEntry 3 }
```

mteTriggerThresholdDeltaRising OBJECT-TYPE

```
SYNTAX      Integer32
MAX-ACCESS  read-write
STATUS      current
```

DESCRIPTION

"A threshold value to check against if mteTriggerType is 'threshold'.

When the delta value (difference) between the current sampled value (value(n)) and the previous sampled value (value(n-1)) is greater than or equal to this threshold, and the delta value calculated at the last sampling interval (i.e. value(n-1) - value(n-2)) was less than this threshold, one mteTriggerThresholdDeltaRisingEvent is triggered. That event is also triggered if the first delta value calculated after this entry becomes active, i.e. value(2) - value(1), where value(1) is the first sample taken of that instance, is greater than or equal to this threshold.

After a rising event is generated, another such event is not triggered until the delta value falls below this threshold and reaches mteTriggerThresholdDeltaFalling."

```
DEFVAL { 0 }
```

```
::= { mteTriggerThresholdEntry 4 }
```

mteTriggerThresholdDeltaFalling OBJECT-TYPE

```
SYNTAX      Integer32
MAX-ACCESS  read-write
STATUS      current
```

DESCRIPTION

"A threshold value to check against if mteTriggerType is 'threshold'.

When the delta value (difference) between the current sampled value (value(n)) and the previous sampled value (value(n-1)) is less than or equal to this threshold, and the delta value calculated at the last sampling interval (i.e. value(n-1) - value(n-2)) was greater than this threshold, one mteTriggerThresholdDeltaFallingEvent is triggered. That event is also triggered if the first delta value calculated after this entry becomes active, i.e. value(2) - value(1), where value(1) is the first sample taken of that instance, is less than or equal to this threshold.

After a falling event is generated, another such event is not triggered until the delta value falls below this threshold and reaches mteTriggerThresholdDeltaRising."

```
DEFVAL { 0 }
```

```
::= { mteTriggerThresholdEntry 5 }
```

mteTriggerThresholdObjectsOwner OBJECT-TYPE

```
SYNTAX      SnmpAdminString (SIZE (0..32))
MAX-ACCESS  read-write
STATUS      current
```

DESCRIPTION

"To go with mteTriggerThresholdObjects, the mteOwner of a group of objects from mteObjectsTable."

```
DEFVAL { ''H }
```

```
::= { mteTriggerThresholdEntry 6 }
```

mteTriggerThresholdObjects OBJECT-TYPE

```
SYNTAX      SnmpAdminString (SIZE (0..32))
MAX-ACCESS  read-write
STATUS      current
```

DESCRIPTION

"The mteObjectsName of a group of objects from mteObjectsTable. These objects are to be added to any Notification resulting from the firing of this trigger for this test.

A list of objects may also be added based on the overall

trigger, the event or other settings in mteTriggerTest.

A length of 0 indicates no additional objects."

```
DEFVAL { ''H }  
::= { mteTriggerThresholdEntry 7 }
```

mteTriggerThresholdRisingEventOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"To go with mteTriggerThresholdRisingEvent, the mteOwner of an event entry from mteEventTable."

```
DEFVAL { ''H }  
::= { mteTriggerThresholdEntry 8 }
```

mteTriggerThresholdRisingEvent OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The mteEventName of the event to invoke when mteTriggerType is 'threshold' and this trigger fires based on mteTriggerThresholdRising. A length of 0 indicates no event."

```
DEFVAL { ''H }  
::= { mteTriggerThresholdEntry 9 }
```

mteTriggerThresholdFallingEventOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"To go with mteTriggerThresholdFallingEvent, the mteOwner of an event entry from mteEventTable."

```
DEFVAL { ''H }  
::= { mteTriggerThresholdEntry 10 }
```

mteTriggerThresholdFallingEvent OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The mteEventName of the event to invoke when mteTriggerType is 'threshold' and this trigger fires based on mteTriggerThresholdFalling. A length of 0 indicates no event."

```
DEFVAL { ''H }  
::= { mteTriggerThresholdEntry 11 }
```

```
mteTriggerThresholdDeltaRisingEventOwner OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "To go with mteTriggerThresholdDeltaRisingEvent, the mteOwner
         of an event entry from mteEventTable."
    DEFVAL { ''H }
    ::= { mteTriggerThresholdEntry 12 }

mteTriggerThresholdDeltaRisingEvent OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "The mteEventName of the event to invoke when mteTriggerType is
         'threshold' and this trigger fires based on
         mteTriggerThresholdDeltaRising. A length of 0 indicates
         no event."
    DEFVAL { ''H }
    ::= { mteTriggerThresholdEntry 13 }

mteTriggerThresholdDeltaFallingEventOwner OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "To go with mteTriggerThresholdDeltaFallingEvent, the mteOwner
         of an event entry from mteEventTable."
    DEFVAL { ''H }
    ::= { mteTriggerThresholdEntry 14 }

mteTriggerThresholdDeltaFallingEvent OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "The mteEventName of the event to invoke when mteTriggerType is
         'threshold' and this trigger fires based on
         mteTriggerThresholdDeltaFalling. A length of 0 indicates
         no event."
    DEFVAL { ''H }
    ::= { mteTriggerThresholdEntry 15 }

--
-- Objects Table
--
```

```

mteObjectsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF MteObjectsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that can be added to notifications based
        on the trigger, trigger test, or event, as pointed to by
        entries in those tables."
    ::= { mteObjects 1 }

mteObjectsEntry OBJECT-TYPE
    SYNTAX      MteObjectsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A group of objects. Applications create and delete entries
        using mteObjectsEntryStatus.

        When adding objects to a notification they are added in the
        lexical order of their index in this table. Those associated
        with a trigger come first, then trigger test, then event."
    INDEX       { mteOwner, mteObjectsName, mteObjectsIndex }
    ::= { mteObjectsTable 1 }

MteObjectsEntry ::= SEQUENCE {
    mteObjectsName          SnmpAdminString,
    mteObjectsIndex         Unsigned32,
    mteObjectsID            OBJECT IDENTIFIER,
    mteObjectsIDWildcard    TruthValue,
    mteObjectsEntryStatus   RowStatus
}

mteObjectsName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (1..32))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A locally-unique, administratively assigned name for a group
        of objects."
    ::= { mteObjectsEntry 1 }

mteObjectsIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..4294967295)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "An arbitrary integer for the purpose of identifying
        individual objects within a mteObjectsName group."

```

Objects within a group are placed in the notification in the numerical order of this index.

Groups are placed in the notification in the order of the selections for overall trigger, trigger test, and event. Within trigger test they are in the same order as the numerical values of the bits defined for mteTriggerTest.

Bad object identifiers or a mismatch between truncating the identifier and the value of mteDeltaDiscontinuityIDWildcard result in operation as one would expect when providing the wrong identifier to a Get operation. The Get will fail or get the wrong object. If the object is not available it is omitted from the notification."

```
::= { mteObjectsEntry 2 }
```

mteObjectsID OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The object identifier of a MIB object to add to a Notification that results from the firing of a trigger.

This may be wildcarded by truncating all or part of the instance portion, in which case the instance portion of the OID for obtaining this object will be the same as that used in obtaining the mteTriggerValueID that fired. If such wildcarding is applied, mteObjectsIDWildcard must be 'true' and if not it must be 'false'.

Each instance that fills the wildcard is independent of any additional instances, that is, wildcarded objects operate as if there were a separate table entry for each instance that fills the wildcard without having to actually predict all possible instances ahead of time."

DEFVAL { zeroDotZero }

```
::= { mteObjectsEntry 3 }
```

mteObjectsIDWildcard OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"Control for whether mteObjectsID is to be treated as fully-specified or wildcarded, with 'true' indicating wildcard."

DEFVAL { false }

```
::= { mteObjectsEntry 4 }
```


mteObjectsEntryStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The control that allows creation and deletion of entries.
Once made active an entry MAY not be modified except to
delete it."

::= { mteObjectsEntry 5 }

--

-- Event Section

--

-- Counters

mteEventFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an attempt to invoke an event
has failed. This counts individually for each
attempt in a group of targets or each attempt for a
wildcarded trigger object."

::= { mteEvent 1 }

--

-- Event Table

--

mteEventTable OBJECT-TYPE

SYNTAX SEQUENCE OF MteEventEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A table of management event action information."

::= { mteEvent 2 }

mteEventEntry OBJECT-TYPE

SYNTAX MteEventEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Information about a single event. Applications create and
delete entries using mteEventEntryStatus."

INDEX { mteOwner, IMPLIED mteEventName }

::= { mteEventTable 1 }

```

MteEventEntry ::= SEQUENCE {
    mteEventName          SnmpAdminString,
    mteEventComment       SnmpAdminString,
    mteEventActions       BITS,
    mteEventEnabled        TruthValue,
    mteEventEntryStatus    RowStatus
}

mteEventName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (1..32))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A locally-unique, administratively assigned name for the
        event."
    ::= { mteEventEntry 1 }

mteEventComment OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "A description of the event's function and use."
    DEFVAL { ''H }
    ::= { mteEventEntry 2 }

mteEventActions OBJECT-TYPE
    SYNTAX      BITS { notification(0), set(1) }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The actions to perform when this event occurs.

        For 'notification', Traps and/or Informs are sent according
        to the configuration in the SNMP Notification MIB.

        For 'set', an SNMP Set operation is performed according to
        control values in this entry."
    DEFVAL { {} } -- No bits set.
    ::= { mteEventEntry 3 }

mteEventEnabled OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "A control to allow an event to be configured but not used.
        When the value is 'false' the event does not execute even if

```

```

        triggered."
    DEFVAL { false }
    ::= { mteEventEntry 4 }

mteEventEntryStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "The control that allows creation and deletion of entries.
        Once made active an entry MAY not be modified except to
        delete it."
    ::= { mteEventEntry 5 }

--
-- Event Notification Table
--

mteEventNotificationTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF MteEventNotificationEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "A table of information about notifications to be sent as a
        consequence of management events."
    ::= { mteEvent 3 }

mteEventNotificationEntry OBJECT-TYPE
    SYNTAX      MteEventNotificationEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "Information about a single event's notification.  Entries
        automatically exist in this table for each mteEventEntry
        that has 'notification' set in mteEventActions."
    INDEX       { mteOwner, IMPLIED mteEventName }
    ::= { mteEventNotificationTable 1 }

MteEventNotificationEntry ::= SEQUENCE {
    mteEventNotification          OBJECT IDENTIFIER,
    mteEventNotificationObjectsOwner  SnmpAdminString,
    mteEventNotificationObjects      SnmpAdminString
}

mteEventNotification OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS   read-write
    STATUS       current

```

DESCRIPTION

"The object identifier from the NOTIFICATION-TYPE for the notification to use if mteEventActions has 'notification' set."

DEFVAL { zeroDotZero }

::= { mteEventNotificationEntry 1 }

mteEventNotificationObjectsOwner OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"To go with mteEventNotificationObjects, the mteOwner of a group of objects from mteObjectsTable."

DEFVAL { ''H }

::= { mteEventNotificationEntry 2 }

mteEventNotificationObjects OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The mteObjectsName of a group of objects from mteObjectsTable if mteEventActions has 'notification' set. These objects are to be added to any Notification generated by this event.

Objects may also be added based on the trigger that stimulated the event.

A length of 0 indicates no additional objects."

DEFVAL { ''H }

::= { mteEventNotificationEntry 3 }

--

-- Event Set Table

--

mteEventSetTable OBJECT-TYPE

SYNTAX SEQUENCE OF MteEventSetEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A table of management event action information."

::= { mteEvent 4 }

mteEventSetEntry OBJECT-TYPE

SYNTAX MteEventSetEntry

MAX-ACCESS not-accessible

```

STATUS      current
DESCRIPTION
    "Information about a single event's set option.  Entries
    automatically exist in this table for each mteEventEntry
    that has 'set' set in mteEventActions."
INDEX       { mteOwner, IMPLIED mteEventName }
 ::= { mteEventSetTable 1 }

```

```

MteEventSetEntry ::= SEQUENCE {
    mteEventSetObject          OBJECT IDENTIFIER,
    mteEventSetObjectWildcard TruthValue,
    mteEventSetValue          Integer32,
    mteEventSetTargetTag      SnmpTagValue,
    mteEventSetContextName    SnmpAdminString,
    mteEventSetContextNameWildcard TruthValue
}

```

```

mteEventSetObject OBJECT-TYPE
SYNTAX      OBJECT IDENTIFIER
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION

```

"The object identifier from the MIB object to set if mteEventActions has 'set' set.

This object identifier may be wildcarded by leaving sub-identifiers off the end, in which case mteEventSetObjectWildcard must be 'true'.

If mteEventSetObject is wildcarded the instance used to set the object to which it points is the same as the instance from the value of mteTriggerValueID that triggered the event.

Each instance that fills the wildcard is independent of any additional instances, that is, wildcarded objects operate as if there were a separate table entry for each instance that fills the wildcard without having to actually predict all possible instances ahead of time.

Bad object identifiers or a mismatch between truncating the identifier and the value of mteSetObjectWildcard result in operation as one would expect when providing the wrong identifier to a Set operation. The Set will fail or set the wrong object. If the value syntax of the destination object is not correct, the Set fails with the normal SNMP error code."

```

DEFVAL { zeroDotZero }
 ::= { mteEventSetEntry 1 }

```

mteEventSetObjectWildcard OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Control over whether mteEventSetObject is to be treated as fully-specified or wildcarded, with 'true' indicating wildcard if mteEventActions has 'set' set."

DEFVAL { false }

::= { mteEventSetEntry 2 }

mteEventSetValue OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The value to which to set the object at mteEventSetObject if mteEventActions has 'set' set."

DEFVAL { 0 }

::= { mteEventSetEntry 3 }

mteEventSetTargetTag OBJECT-TYPE

SYNTAX SnmpTagValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The tag for the target(s) at which to set the object at mteEventSetObject to mteEventSetValue if mteEventActions has 'set' set."

Systems limited to self management MAY reject a non-zero length for the value of this object.

A length of 0 indicates the local system. In this case, access to the objects indicated by mteEventSetObject is under the security credentials of the requester that set mteTriggerEntryStatus to 'active'. Those credentials are the input parameters for isAccessAllowed from the Architecture for Describing SNMP Management Frameworks.

Otherwise access rights are checked according to the security parameters resulting from the tag."

DEFVAL { ''H }

::= { mteEventSetEntry 4 }

mteEventSetContextName OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The management context in which to set mteEventObjectID.
if mteEventActions has 'set' set.

This may be wildcarded by leaving characters off the end. To
indicate such wildcarding mteEventSetContextNameWildcard must
be 'true'.

If this context name is wildcarded the value used to complete
the wildcarding of mteTriggerContextName will be appended."

DEFVAL { ''H }
::= { mteEventSetEntry 5 }

mteEventSetContextNameWildcard OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Control for whether mteEventSetContextName is to be treated as
fully-specified or wildcarded, with 'true' indicating wildcard
if mteEventActions has 'set' set."

DEFVAL { false }
::= { mteEventSetEntry 6 }

--

-- Notifications

--

dismanEventMIBNotificationPrefix OBJECT IDENTIFIER ::=

{ dismanEventMIB 2 }

dismanEventMIBNotifications OBJECT IDENTIFIER ::=

{ dismanEventMIBNotificationPrefix 0 }

dismanEventMIBNotificationObjects OBJECT IDENTIFIER

::= { dismanEventMIBNotificationPrefix 1 }

--

-- Notification Objects

--

mteHotTrigger OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS accessible-for-notify

STATUS current

DESCRIPTION

"The name of the trigger causing the notification."

::= { dismanEventMIBNotificationObjects 1 }

mteHotTargetName OBJECT-TYPE
SYNTAX SnmpAdminString
MAX-ACCESS accessible-for-notify
STATUS current
DESCRIPTION
 "The SNMP Target MIB's snmpTargetAddrName related to the notification."
 ::= { dismanEventMIBNotificationObjects 2 }

mteHotContextName OBJECT-TYPE
SYNTAX SnmpAdminString
MAX-ACCESS accessible-for-notify
STATUS current
DESCRIPTION
 "The context name related to the notification. This MUST be as fully-qualified as possible, including filling in wildcard information determined in processing."
 ::= { dismanEventMIBNotificationObjects 3 }

mteHotOID OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
MAX-ACCESS accessible-for-notify
STATUS current
DESCRIPTION
 "The object identifier of the destination object related to the notification. This MUST be as fully-qualified as possible, including filling in wildcard information determined in processing.

 For a trigger-related notification this is from mteTriggerValueID.

 For a set failure this is from mteEventSetObject."
 ::= { dismanEventMIBNotificationObjects 4 }

mteHotValue OBJECT-TYPE
SYNTAX Integer32
MAX-ACCESS accessible-for-notify
STATUS current
DESCRIPTION
 "The value of the object at mteTriggerValueID when a trigger fired."
 ::= { dismanEventMIBNotificationObjects 5 }

mteFailedReason OBJECT-TYPE
SYNTAX FailureReason
MAX-ACCESS accessible-for-notify
STATUS current

DESCRIPTION

"The reason for the failure of an attempt to check for a trigger condition or set an object in response to an event."

::= { dismanEventMIBNotificationObjects 6 }

--

-- Notifications

--

mteTriggerFired NOTIFICATION-TYPE

OBJECTS { mteHotTrigger,
 mteHotTargetName,
 mteHotContextName,
 mteHotOID,
 mteHotValue }

STATUS current

DESCRIPTION

"Notification that the trigger indicated by the object instances has fired, for triggers with mteTriggerType 'boolean' or 'existence'."

::= { dismanEventMIBNotifications 1 }

mteTriggerRising NOTIFICATION-TYPE

OBJECTS { mteHotTrigger,
 mteHotTargetName,
 mteHotContextName,
 mteHotOID,
 mteHotValue }

STATUS current

DESCRIPTION

"Notification that the rising threshold was met for triggers with mteTriggerType 'threshold'."

::= { dismanEventMIBNotifications 2 }

mteTriggerFalling NOTIFICATION-TYPE

OBJECTS { mteHotTrigger,
 mteHotTargetName,
 mteHotContextName,
 mteHotOID,
 mteHotValue }

STATUS current

DESCRIPTION

"Notification that the falling threshold was met for triggers with mteTriggerType 'threshold'."

::= { dismanEventMIBNotifications 3 }

mteTriggerFailure NOTIFICATION-TYPE

OBJECTS { mteHotTrigger,

```

        mteHotTargetName,
        mteHotContextName,
        mteHotOID,
        mteFailedReason }
STATUS    current
DESCRIPTION
    "Notification that an attempt to check a trigger has failed.

    The network manager must enable this notification only with
    a certain fear and trembling, as it can easily crowd out more
    important information. It should be used only to help diagnose
    a problem that has appeared in the error counters and can not
    be found otherwise."
 ::= { dismanEventMIBNotifications 4 }

mteEventSetFailure NOTIFICATION-TYPE
OBJECTS { mteHotTrigger,
          mteHotTargetName,
          mteHotContextName,
          mteHotOID,
          mteFailedReason }
STATUS    current
DESCRIPTION
    "Notification that an attempt to do a set in response to an
    event has failed.

    The network manager must enable this notification only with
    a certain fear and trembling, as it can easily crowd out more
    important information. It should be used only to help diagnose
    a problem that has appeared in the error counters and can not
    be found otherwise."
 ::= { dismanEventMIBNotifications 5 }

--
-- Conformance
--

dismanEventMIBConformance OBJECT IDENTIFIER ::= { dismanEventMIB 3 }
dismanEventMIBCompliances OBJECT IDENTIFIER ::=
    { dismanEventMIBConformance 1 }
dismanEventMIBGroups      OBJECT IDENTIFIER ::=
    { dismanEventMIBConformance 2 }

-- Compliance

dismanEventMIBCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION

```

```
"The compliance statement for entities which implement
the Event MIB."
MODULE  -- this module
MANDATORY-GROUPS {
    dismanEventResourceGroup,
    dismanEventTriggerGroup,
    dismanEventObjectsGroup,
    dismanEventEventGroup,
    dismanEventNotificationObjectGroup,
    dismanEventNotificationGroup
}

OBJECT mteTriggerTargetTag
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required, thus limiting
    monitoring to the local system or pre-configured
    remote systems."

OBJECT mteEventSetTargetTag
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required, thus limiting
    setting to the local system or pre-configured
    remote systems."

OBJECT mteTriggerValueIDWildcard
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required, thus allowing
    the system not to implement wildcarding."

OBJECT mteTriggerContextNameWildcard
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required, thus allowing
    the system not to implement wildcarding."

OBJECT mteObjectsIDWildcard
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required, thus allowing
    the system not to implement wildcarding."

OBJECT mteEventSetContextNameWildcard
MIN-ACCESS read-only
DESCRIPTION
```

"Write access is not required, thus allowing the system not to implement wildcarding."

::= { dismanEventMIBCompliances 1 }

-- Units of Conformance

dismanEventResourceGroup OBJECT-GROUP

OBJECTS {
 mteResourceSampleMinimum,
 mteResourceSampleInstanceMaximum,
 mteResourceSampleInstances,
 mteResourceSampleInstancesHigh,
 mteResourceSampleInstanceLacks
}

STATUS current

DESCRIPTION

"Event resource status and control objects."

::= { dismanEventMIBGroups 1 }

dismanEventTriggerGroup OBJECT-GROUP

OBJECTS {
 mteTriggerFailures,

 mteTriggerComment,
 mteTriggerTest,
 mteTriggerSampleType,
 mteTriggerValueID,
 mteTriggerValueIDWildcard,
 mteTriggerTargetTag,
 mteTriggerContextName,
 mteTriggerContextNameWildcard,
 mteTriggerFrequency,
 mteTriggerObjectsOwner,
 mteTriggerObjects,
 mteTriggerEnabled,
 mteTriggerEntryStatus,

 mteTriggerDeltaDiscontinuityID,
 mteTriggerDeltaDiscontinuityIDWildcard,
 mteTriggerDeltaDiscontinuityIDType,
 mteTriggerExistenceTest,
 mteTriggerExistenceStartup,
 mteTriggerExistenceObjectsOwner,
 mteTriggerExistenceObjects,
 mteTriggerExistenceEventOwner,
 mteTriggerExistenceEvent,

```

        mteTriggerBooleanComparison,
        mteTriggerBooleanValue,
        mteTriggerBooleanStartup,
        mteTriggerBooleanObjectsOwner,
        mteTriggerBooleanObjects,
        mteTriggerBooleanEventOwner,
        mteTriggerBooleanEvent,

        mteTriggerThresholdStartup,
        mteTriggerThresholdObjectsOwner,
        mteTriggerThresholdObjects,
        mteTriggerThresholdRising,
        mteTriggerThresholdFalling,
        mteTriggerThresholdDeltaRising,
        mteTriggerThresholdDeltaFalling,
        mteTriggerThresholdRisingEventOwner,
        mteTriggerThresholdRisingEvent,
        mteTriggerThresholdFallingEventOwner,
        mteTriggerThresholdFallingEvent,
        mteTriggerThresholdDeltaRisingEventOwner,
        mteTriggerThresholdDeltaRisingEvent,
        mteTriggerThresholdDeltaFallingEventOwner,
        mteTriggerThresholdDeltaFallingEvent
    }
    STATUS current
    DESCRIPTION
        "Event triggers."
    ::= { dismanEventMIBGroups 2 }

dismanEventObjectsGroup OBJECT-GROUP
    OBJECTS {
        mteObjectsID,
        mteObjectsIDWildcard,
        mteObjectsEntryStatus
    }
    STATUS current
    DESCRIPTION
        "Supplemental objects."
    ::= { dismanEventMIBGroups 3 }

dismanEventEventGroup OBJECT-GROUP
    OBJECTS {
        mteEventFailures,

        mteEventComment,
        mteEventActions,
        mteEventEnabled,
        mteEventEntryStatus,
    }

```

```
        mteEventNotification,
        mteEventNotificationObjectsOwner,
        mteEventNotificationObjects,

        mteEventSetObject,
        mteEventSetObjectWildcard,
        mteEventSetValue,
        mteEventSetTargetTag,
        mteEventSetContextName,
        mteEventSetContextNameWildcard
    }
    STATUS current
    DESCRIPTION
        "Events."
    ::= { dismanEventMIBGroups 4 }

dismanEventNotificationObjectGroup OBJECT-GROUP
    OBJECTS {
        mteHotTrigger,
        mteHotTargetName,
        mteHotContextName,
        mteHotOID,
        mteHotValue,
        mteFailedReason
    }
    STATUS current
    DESCRIPTION
        "Notification objects."
    ::= { dismanEventMIBGroups 5 }

dismanEventNotificationGroup NOTIFICATION-GROUP
    NOTIFICATIONS {
        mteTriggerFired,
        mteTriggerRising,
        mteTriggerFalling,
        mteTriggerFailure,
        mteEventSetFailure
    }
    STATUS current
    DESCRIPTION
        "Notifications."
    ::= { dismanEventMIBGroups 6 }

END
```

8. 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.

9. Acknowledgements

This MIB contains considerable contributions from the RMON MIB, the Distributed Management Design Team (Andy Bierman, Maria Greene, Bob Stewart, and Steve Waldbusser), the Distributed Management Working Group, and colleagues at Cisco.

10. References

- [RFC2571] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.
- [RFC1155] Rose, M. and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.
- [RFC1212] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.
- [RFC1215] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

- [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", STD 58, RFC 2580, April 1999.
- [RFC1157] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.
- [RFC1901] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.
- [RFC1906] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.
- [RFC2572] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.
- [RFC2574] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.
- [RFC1905] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.
- [RFC2573] Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.
- [RFC2575] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.
- [RFC2570] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.

- [RFC1903] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Coexistence between Version 1 and version 2 of the Internet-standard Network Management Framework", RFC 1903, January 1996.
- [RFC2981] Stewart, B., "Event MIB", RFC 2981, October 2000.
- [RFC1757] Waldbusser, S., "Remote Network Monitoring Management Information Base", RFC 1757, February 1995.
- [RFC1451] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Manager-to- Manager Management Information Base", RFC 1451, April 1993.
- [RFC2982] Stewart, B., "Distributed Management Expression MIB", RFC 2982, October 2000.
- [LOG-MIB] Stewart, B., "Notification Log MIB", Work in Progress.

11. Security Considerations

Security issues are discussed in the Security section and in the DESCRIPTION clauses of relevant objects.

12. Author's Address

Bob Stewart
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
U.S.A.

13. Editor's Address

Ramanathan Kavasseri
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
U.S.A.

Phone: +1 408 527 2446
EMail: ramk@cisco.com

14. Full Copyright Statement

Copyright (C) The Internet Society (2000). 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.

