Expression MIB and Event MIB Configuration Example

Document ID: 45282

Contents

Introduction

This document shows how to combine the Expression MIB and the Event MIB for use in fault management. The included example is not realistic but shows many available features.

The router must perform two actions:

1. Send a trap if a loopback interface has a bandwidth higher than 100 and is administratively down
2. The loopback interface shuts down if one of the interfaces has its bandwidth statement changed from a defined value

The example is shown with bandwidth and admin status because they are easy to manipulate from the command line and both show integer and boolean values.

The commands in this document use the object identifier (OID) parameter and not the object names. This allows testing without loading the MIB.

Prerequisites

Requirements

Before using the information in this document, ensure that you meet the following prerequisites:

- The workstation should have Simple Network Management Protocol (SNMP) tools provided by Hewlett−Packard (HP) Openview. Other SNMP tools work but may have different syntax.
- The device must run Cisco IOS® Software Release 12.2(4)T3 or later. Earlier versions do not support the RFC version of the Event MIB.
- The platform must support the Event MIB. For a list of supported platforms for Cisco IOS Software Release 12.1(3)T, refer to the "Supported Platform" section of Event MIB Support.

Components Used

The information in this document is based on these software and hardware versions:

- Cisco IOS Software Release 12.3(1a)
- Cisco 3640 Modular Access Router

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Background Information

- The Expression MIB allows the user to create his own MIB object based on a combination of other objects. For more information, refer to RFC 2982 .
- The Event MIB allows the user to have the device monitoring its own MIB objects and to generate actions (notification or **SNMP SET** commands) based on a defined event. For more information, refer to RFC 2981 .

Configure

Note: Some of the lines of output code are displayed over two lines to better fit onto your screen.

In this example, the ifIndex of the loopback interface is equal to 16.

```
# snmpget -v 2c -c private router .1.3.6.1.2.1.2.2.1.2.16
IF-MIB::ifDescr.16 = STRING: Loopback0
```

The variable names related to the first event start with `e1` and those related to the second start with `e2`. The router name is "router" and the read/write community string is "private."

The Expression MIB

Creating the Expression 1

First create an expression that returns a value of 1 if the condition, `ifSpeed is greater than 100,000` AND `ifAdminStatus is down` for the loopback interface. If the condition is not met, it returns the value 0.

1. expExpressionDeltaInterval This object is not used.

   There is no reason to calculate an expression when it is not polled. If no value is set, the expression is calculated when the object is queried.

   The expression name is `e1exp`, which in the ASCII table corresponds to `101 49 101 120 112`.
2. expNameStatus This destroys an eventual old expression that is created.

   ```
   # snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.3.101.49.101.120.112 i
   ```
3. expNameStatus Create and wait.

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.3.101.49.101.120.112 i
```

4. expExpressionIndex This creates the index to use later to retrieve the result of the expression.

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.2.101.49.101.120.112 g
```

5. expExpressionComment Here .1 (the chosen expExpressionIndex) is the description of the expression.

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.4.1 octetstring "e1 ex
```

6. expExpression This is the expression itself, the variables $1 and $2 are defined at the next step.

   The only allowed operators are (for details, refer to RFC 2982):

```
( )
- (unary)
+ - * / %
& | ^ << >> ~
! && || == != > >= < <=
```

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.2.1 octetstring '$1 <
```

7. expObjectID

```
.1 is for the variable $1 => ifSpeed
.2 for $2 => ifAdminStatus
```

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.2.1.1 objectidentifier
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.2.1.2 objectidentifier
```

8. expObjectSampleType The two values are taken in absolute values (for Delta, take 2 as the value).

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.4.1.1 integer 1
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.4.1.2 integer 1
```

9. expObjectIDWildcard The object IDs are not wildcarded. This is the default value, so do not snmpset expObjectIDWildcard.
10. expObjectStatus Set the rows in the expObjectTable to active.

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.10.1.1 integer 1
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.10.1.2 integer 1
```

11. Activate the expression 1.

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.3.101.49.101.120.112 i
```

Testing the Expression 1

```
router(config)#interface loopback 0
router(config-if)#shutdown
router(config-if)#bandwidth 150
```

1. If the condition is met, the value of expValueCounter32Val is 1 (as the value of expExpressionValueType remains unchanged, the result is a counter32).

   Note: The type cannot be a floating point value.

```
# snmpwalk -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.4.1.1.2
cisco.ciscoExperiment.22.1.4.1.1.2.1.0.0.0 : Counter: 1
```

```
router(config-if)#bandwidth 150000
```

2. If the condition is not met, the value is 0.

```
# snmpwalk -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.4.1.1.2
cisco.ciscoExperiment.22.1.4.1.1.2.1.0.0.0 : Counter: 0

router(config-if)#bandwidth 1
router(config-if)#no shutdown
```

3. If the condition is not met, the value is 0.

```
# snmpwalk -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.4.1.1.2
cisco.ciscoExperiment.22.1.4.1.1.2.1.0.0.0 : Counter: 0
```

Creating and Testing Expression 2

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.3.101.50.101.120.112 integer
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.3.101.50.101.120.112 integer
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.2.101.50.101.120.112 gauge 2
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.4.2 octetstring "e2 expressi
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.2.2 octetstring ''($1 * 18) /
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.2.2.1 objectidentifier 1.3.6
```

1. expObjectIDWildcard This indicates that `1.3.6.1.2.1.2.2.1.5` is a table and not an object.

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.3.2.1 integer 1
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.4.2.1 integer 1
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.10.2.1 integer 1
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.2.3.1.3.101.50.101.120.112 i
```

2. Test:

```
# snmpwalk router 1.3.6.1.4.1.9.10.22.1.4.1.1
[...]
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.10 : Counter: 0
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.11 : Counter: 23250000
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.12 : Counter: 42949672
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.13 : Counter: 18450
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.14 : Counter: 150
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.15 : Counter: 1350
cisco.ciscoExperiment.22.1.4.1.1.2.2.0.0.16 : Counter: 9600
```

Event MIB

Creating Event 1

Now create an event that checks the output value of the first expression every 60 seconds and compares it with a reference. When the reference matches the expression value, a trap is triggered with the chosen VARBIND.

1. Create the trigger in the trigger table.

   - The name of the trigger is trigger1, which in ASCII code is `116 114 105 103 103 101 114 49`.
   - The owner is tom: `116 111 109`.
   - The index of the mteTriggerEntry is composed of the trigger owner and the trigger name. The first value of the index gives the number of characters for the mteOwner.

     In this case, there are three characters for tom, so the index is:
     `3.116.111.109.116.114.105.103.103.101.114.49.`

2. Destroy the old entry if it exists.
3. Set the trigger status to **create and wait**.
4. The last step activates it:

a. mteTriggerEntryStatus

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116
integer 6
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116
integer 5
```

b. mteTriggerValueID The value of the first expression is `e1exp`.

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

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.6.3.116.111.109.116.
objectidentifier
1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0
```

c. mteTriggerValueIDWildcard Without using a wildcard for the value ID.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.7.3.116.111.109.116.
integer 2
```

d. mteTriggerTest Existence (0), boolean (1), and threshold (2).

The method to select one of the above values is a complex. To select an existence, give a value in eight digits in which the first is a 1, such as `10000000` or `100xxxxx`.

For a boolean, the second digit must be a 1: `01000000` or `010xxxxx`.

For a threshold, the third digit must be a 1: `00100000` or `001xxxxx`.

It is easiest to work this way:

- ◊ For existence, the value is octetstringhex 80.
- ◊ For boolean, the value is octetstringhex¤0.
- ◊ For threshold, the value is octetstringhex¢0.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.4.3.116.111.109.116.
octetstringhex "40"
```

e. mteTriggerFrequency This determines the number of seconds to wait between trigger samples.

The minimum value is set with the object mteResourceSampleMinimum (default is 60 seconds), lowering this value increases the CPU usage, so it must be done carefully.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.11.3.116.111.109.116
gauge 60
```

f. mteTriggerSampleType These are absoluteValue (1) and deltaValue (2). In this case, the value is absolute:

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.5.3.116.111.109.116.
integer 1
```

g. mteTriggerEnabled This is a control that allows a trigger to be configured but not used. Set it to true (default is false).

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.14.3.116.111.109.116
integer 1
```

h. Now that the trigger as been created, define the event the trigger will use. The event name is event1.

mteEventEntryStatus

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.
integer 6
```

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.
integer 5
```

i. mteEventActions These are notification (0) and set (1).

The process is the same as for mteTriggerTest. Notification is `10xxxxxx` and set is `01xxxxxx`.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.3.3.116.111.109.101.
octetstringhex "80"
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.4.3.116.111.109.101.
integer 1
```

j. This next step defines the test to be done on the object selected for trigger1.

mteTriggerBooleanComparison These are unequal (1), equal (2), less (3), lessOrEqual (4), greater (5), and greaterOrEqual (6). In this case equal.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.5.1.1.3.116.111.109.116.
integer 2
```

k. mteTriggerBooleanValue This is the value to use for the test. If the value of `1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0` is equal to 1, then the condition is met.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.5.1.2.3.116.111.109.116.
integer 1
```

l. Now define the object to be sent with the event.

mteTriggerBooleanObjectsOwner

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.5.1.4.3.116.111.109.116.
octetstring "tom"
```

mteTriggerBooleanObjects

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.5.1.5.3.116.111.109.116.
octetstring "objects1"
```

mteTriggerBooleanEventOwner

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.5.1.6.3.116.111.109.116.
octetstring "tom"
```

mteTriggerBooleanEvent

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.5.1.7.3.116.111.109.116.
octetstring "event1"
```

m. Create the object table.

Send the value of `1.3.6.1.2.1.2.2.1.5.16` as VARBIND with the trap.

Object Table mteObjectsName Objects1.

mteObjectsEntryStatus

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.3.1.1.5.3.116.111.109.8.11
integer 6
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.3.1.1.5.3.116.111.109.8.11
integer 5
```

mteObjectsID

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.3.1.1.3.3.116.111.109.8.11
objectidentifier 1.3.6.1.2.1.2.2.1.5.16
```

mteObjectsIDWildcard There is no wildcard used.

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.3.1.1.4.3.116.111.109.8.11
integer 1
```

n. Activate the object table.

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.3.1.1.5.3.116.111.109.8.11
integer 1
```

o. Attach the object to the event1.

Notify mteEventName Event1.

mteEventNotificationObjectsOwner

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.4.3.1.2.3.116.111.109.101.
octetstring "tom"
```

mteEventNotificationObjects

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.4.3.1.3.3.116.111.109.101.
octetstring "objects1"
```

p. Activate the trigger.

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116
integer 1
```

q. Activate the event.

```
# snmpset –v 2c –c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.
integer 1
```

Trap Received

```
Enterprise : 1.3.6.1.2.1.88.2
Trap type : ENTERPRISE SPECIFIC (6)
Specific trap type: 1
object 1 : mteHotTrigger
value : STRING: "trigger1"
object 2 : mteHotTargetName
value: ""
object 3 : mteHotContextName
value: ""
object 4: mteHotOID
value: OID: 1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0
object 5: mteHotValue
value: INTEGER: 1
object 6:  1.3.6.1.2.1.2.2.1.5.16
value: Gauge32: 1000
```

Note: Object 6 is the VARBIND that was added.

Creating Event 2

Follow these steps:

1. mteTriggerName Trigger2.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.10
integer 6
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.10
integer 5
```

2. mteTriggerValueID This is the value of the first expression and mteTriggerValueIDWildcard.

   This time, the process wildcards the value ID, the object identifier of the MIB object to sample to determine if the trigger fires.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.6.3.116.111.109.116.114.105
objectidentifier
1.3.6.1.4.1.9.10.22.1.4.1.1.2.2.0.0
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.7.3.116.111.109.116.114.105
integer 1
```

3. mteTriggerTest Threshold.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.4.3.116.111.109.116.114.105
octetstringhex "20"
```

4. mteTriggerFrequency

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.11.3.116.111.109.116.114.10
gauge 60
```

5. mteTriggerSampleType Delta value.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.5.3.116.111.109.116.114.105
integer 2
```

6. mteTriggerEnabled

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.2.1.14.3.116.111.109.116.114.10
integer 1
```

7. Create an event in the event table // mteEventName event2.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101
integer 6
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101
integer 5
```

8. mteEventActions The value 40 is for Set, meaning that when the condition is met, the router issues an **snmp set** command. In this case, it makes the Set for itself, but it could also make the operation on a remote device.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.3.3.116.111.109.101.118.101
octetstringhex "40"
```

9. Enable the event.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.4.3.116.111.109.101.118.101
integer 1
```

10. Set The Trigger Threshold in the Trigger Table // index = mteTriggerName Trigger2.

    As it is a threshold, give values for failing and rising conditions. Take only the rising condition this time.

11. mteTriggerThresholdDeltaRising This is the threshold value to check against.

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.6.1.4.3.116.111.109.116.114.105
integer 100
```

12. mteTriggerThresholdDeltaRisingEventOwner

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.2.6.1.12.3.116.111.109.116.114.10
```

```
octetstring "tom"
```

13. mteTriggerThresholdDeltaRisingEvent

```
# snmpset −v 2c −c private router 1.3.6.1.2.1.88.1.2.6.1.13.3.116.111.109.116.114.10
octetstring "event2"
```

14. mteEventSetObject This is the object identifier from the MIB object to set. Here, ifAdminStatus for the loopback interface.

```
# snmpset −v 2c −c private router 1.3.6.1.2.1.88.1.4.4.1.1.3.116.111.109.101.118.101
objectidentifier 1.3.6.1.2.1.2.2.1.7.16
```

15. mteEventSetValue This is the value to set (2 for down).

```
# snmpset −v 2c −c private router 1.3.6.1.2.1.88.1.4.4.1.3.3.116.111.109.101.118.101
integer 2
```

16. Activate the trigger.

```
# snmpset −v 2c −c private router 1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.10
integer 1
```

17. Activate the event.

```
# snmpset −v 2c −c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101
integer 1
```

Result

```
router(config)#int lo1
router(config−if)#bandwidth 5000000
16:24:11: %SYS−5−CONFIG_I: Configured from 10.48.71.71 by snmp
16:24:13: %LINK−5−CHANGED: Interface Loopback1, changed state to administratively down
16:24:14: %LINEPROTO−5−UPDOWN: Line protocol on Interface Loopback1, changed state to down
```

Note: Here, `10.48.71.71` is the address of the router itself.

Verify

This section provides information to use to confirm the configuration is working properly.

Certain **show** commands are supported by the Output Interpreter Tool (registered customers only) , which allows you to view an analysis of **show** command output.

```
router #show management event
 Mgmt Triggers:
 (1): Owner: tom
   (1): trigger1, Comment: , Sample: Abs, Freq: 15
         Test:  Boolean
         ObjectOwner: , Object:
         OID: ciscoExperiment.22.1.4.1.1.2.1.0.0.0, Enabled 1, Row Status 1
      Boolean Entry:
         Value: 1, Cmp: 2, Start: 1
         ObjOwn: tom, Obj: objects1, EveOwn: tom, Eve: event1

      Delta Value Table:
   (0): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.1.0.0.0 , val: 0
   (2): trigger2, Comment: , Sample: Del, Freq: 60
         Test:  Threshold
         ObjectOwner: , Object:
         OID: ciscoExperiment.22.1.4.1.1.2.2.0.0, Enabled 1, Row Status 1
      Threshold Entry:
         Rising: 0, Falling: 0, DeltaRising: 100, DeltaFalling: 0
         ObjOwn: , Obj:
```

```
          RisEveOwn: , RisEve: , FallEveOwn: , FallEve:
          DelRisEveOwn: tom, DelRisEve: event2, DelFallEveOwn: , DelFallEve:

       Delta Value Table:
    (0): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.1 , val: 62000
    (1): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.2 , val: 40000
    (2): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.3 , val: 61760
    (3): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.4 , val: 61760
    (4): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.5 , val: 61760
    (5): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.6 , val: 61760
    (6): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.7 , val: 85899
    (7): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.8 , val: 0
    (8): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.9 , val: 62000
    (9): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.10 , val: 0
    (10): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.11 , val: 620
    (11): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.12 , val: 858
    (12): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.13 , val: 858
    (13): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.14 , val: 400
    (14): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.15 , val: 360
    (15): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.16 , val: 256

 Mgmt Events:
 (1): Owner: tom
   (1)Name: event1, Comment: , Action: Notify, Enabled: 1 Status: 1
      Notification Entry:
         ObjOwn: tom, Obj: objects1, OID: ccitt.0
   (2)Name: event2, Comment: , Action: Set, Enabled: 1 Status: 1
      Set:
         OID: ifEntry.7.13, SetValue: 2, Wildcard: 2
         TAG: , ContextName:

 Object Table:
 (1): Owner: tom
   (1)Name: objects1, Index: 1, OID: ifEntry.5.13, Wild: 2, Status: 1

 Failures: Event = 44716, Trigger = 0


router #show management expression
Expression: e1exp is active
  Expression to be evaluated is $1 < 100000 && $2 == 2 where:
  $1 = ifEntry.5.13
  Object Condition is not set
  Sample Type is absolute
  Both ObjectID and ObjectConditional are not wildcarded
  $2 = ifEntry.7.13
  Object Condition is not set
  Sample Type is absolute
  Both ObjectID and ObjectConditional are not wildcarded

Expression: e2exp is active
  Expression to be evaluated is ($1 * 18) / 23 where:
  $1 = ifEntry.5
  Object Condition is not set
  Sample Type is absolute
  ObjectID is wildcarded
```

Troubleshoot

This section provides information to use to troubleshoot the configuration.

Troubleshooting Commands

These are the commands to enable debugging:

```
router#debug management expression mib
router#debug management event mib
```

Note: Before you issue **debug** commands, refer to Important Information on Debug Commands.

Related Information

- **Expression MIB : RFC 2982**
- **Event MIB : RFC 2981**
- **EXPRESSION−MIB.my / EVENT−MIB.my**
- **IOS feature guide : Event MIB support**
- **Technical Support − Cisco Systems**

Updated: Oct 26, 2005 Document ID: 45282