

表达式MIB和事件MIB配置示例

目录

[简介](#)

[先决条件](#)

[要求](#)

[使用的组件](#)

[规则](#)

[背景信息](#)

[配置](#)

[表达式MIB](#)

[事件MIB](#)

[验证](#)

[故障排除](#)

[故障排除命令](#)

[相关信息](#)

简介

本文显示如何结合表达式MIB和事件MIB用于故障管理。包括的示例不是可实现的，而是显示许多可用的功能。

路由器必须进行两操作：

1. 发送陷阱，如果回环接口有一个的带宽高于100并且管理上下降状态
2. 如果其中一个接口有从一个定义值，更改的其带宽说明回环接口关闭了

示例以带宽和管理状态表示，因为他们是容易从line命令操作，并且两个显示整数和布尔值。

in命令本文使用Object Identifier (OID)参数而不是对象名。这允许测试，无需装载MIB。

先决条件

要求

在使用本文档中的信息前，请保证您满足以下前提条件：

- 工作站应该有惠普(HP) OpenView提供的简单网络管理协议(SNMP)工具。其他SNMP工具工作，但是可能有另外语法。
- 设备必须运行Cisco IOS软件版本12.2(4)T3或以后。更早版本不支持事件MIB的RFC版本。
- 平台必须支持事件MIB。对于支持的平台列表Cisco IOS软件版本12.1(3)T的，参考[事件MIB支持的“支持的平台”部分](#)。

使用的组件

本文档中的信息基于以下软件和硬件版本：

- Cisco IOS软件版本12.3(1a)
- Cisco 3640模块化访问路由器

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您使用的是真实网络，请确保您已经了解所有命令的潜在影响。

规则

有关文档规则的详细信息，请参阅 [Cisco 技术提示规则](#)。

背景信息

- 表达式MIB允许用户创建根据其他对象的组合的他自己的MIB对象。欲知更多信息，参考[RFC 2982](#)。
- 事件MIB允许用户有监控其自己的MIB对象的设备和生成根据一个定义事件(通知或SNMP SET命令)的操作。欲知更多信息，参考[RFC 2981](#)。

配置

注意：某些输出代码线路显示两条线路改善在您的屏幕上的适应。

在本例中，回环接口的IfIndex是相等的到16。

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

与第一个事件涉及的变量名称从e1开始，并且与第二涉及的那些从e2开始。路由器名称是“路由器”，并且读/写社区字符串“私有”。

表达式MIB

创建表达式1

首先请创建返回值为1的表达式，如果情况，ifSpeed100,000并且ifAdminStatus为回环接口。如果情况没有符合，返回值0。

1. [expExpressionDeltaInterval](#) —没有使用此对象。没有理由计算表达式，当它没有轮询。如果值没有设置，表达式计算，当对象被查询时。表达式名称是e1exp，在ASCII表里对应到101 49 101 120 112。
2. [expNameStatus](#) —这毁坏创建的一个最后的旧有表达式。

```
# 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 integer 6
```
3. [expNameStatus](#) —创建并且等待。

```
# 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 integer 5
```

4. [expExpressionIndex](#) —这创建索引使用以后获取表达式的结果。

```
# 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 gauge 1
```

5. [expExpressionComment](#) —在这里.1 (选定的expExpressionIndex)表达式的说明。

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

6. [expExpression](#) —这是表达式，变量\$1和\$2定义在下一步。唯一的允许操作员是(关于详细信息，参考[RFC 2982](#))：

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

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.2.1 octetstring '$1 < 100000 && $2 == 2'
```

7. [expObjectID](#)

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.2.1 octetstring '$1 < 100000 && $2 == 2'
```

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.2.1.1 objectidentifier 1.3.6.1.2.1.2.2.1.5.16
```

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.2.1.2 objectidentifier 1.3.6.1.2.1.2.2.1.7.16
```

8. [expObjectSampleType](#) —两个值被占用按绝对值(对于达美航空，请采取2作为值)。

```
# 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](#) —对象ID不是使用通配符的。这是默认值，如此不snmpset expObjectIDWildcard。

10. [expObjectStatus](#) —设置在expObjectTable的行为激活。

```
# 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. 激活表达式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 integer 1
```

[测试表达式1](#)

```
router(config)#interface loopback 0
```

```
router(config-if)#shutdown
```

```
router(config-if)#bandwidth 150
```

1. 如果情况符合，值[expValueCounter32Val](#)是1 (因为值[expExpressionValueType](#)依然是不可更改，结果是counter32)。注意：类型不可以是浮点值。

```
# 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. 如果情况没有符合，值是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. 如果情况没有符合，值是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
```

[创建和测试表达式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 6  
# 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 5  
# 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 expression"  
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.2.2 octetstring '($1 * 18) / 23'  
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.2.1.2.2.1 objectidentifier  
1.3.6.1.2.1.2.2.1.5
```

1. [expObjectIDWildcard](#) —这表明1.3.6.1.2.1.2.2.1.5是表而不是对象。

```
# 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 integer  
1
```

2. 测验：

```
# 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
```

[事件MIB](#)

[创建Event1](#)

现在请创建检查第一个表达式输出值每60秒并且它与参考比较的事件。当参考匹配表达式值时，陷阱触发与选定的VARBIND。

1. 创建触发在触发表里。触发的名称是trigger1，用ASCII代码是116 114 105 103 103 101 114

49. 所有者是汤姆：116 111 109。mteTriggerEntry的索引撰写触发所有者和触发名称。索引的第一个值给字符数量mteOwner的。在这种情况下，有汤姆的三个字符，因此索引是

```
: 3.116.111.109.116.114.105.103.103.101.114.49.
```

2. 如果存在，请毁坏旧有条目。
3. 设置触发状态**创建和等待**。
4. 最后一步激活它：[mteTriggerEntryStatus](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.49
integer 6
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.49
integer 5
```

[mteTriggerValueID](#) —第一个表达式的值是 e_{1exp} 。MIB对象的对象标识符是采样的那个发现触发应该是否射击。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.6.3.116.111.109.116.114.105.103.103.101.114.49
objectidentifier
1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0
```

[mteTriggerValueIDWildcard](#) —没有使用通配符值ID。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.7.3.116.111.109.116.114.105.103.103.101.114.49
integer 2
```

[mteTriggerTest](#) —存在(0)，布尔型(1)和阈值(2)。选择其中一个的方法上述值是复杂。要选择存在，请给在第一是-1的八个位，例如10000000或100xxxxxx一个值。对于布尔型，第二个数字必须是1：01000000或010xxxxxx。对于阈值，第三个数字必须是1：00100000或001xxxxxx。运转这样是最容易的：对于存在，值是octetstringhex — 80。对于布尔型，值是octetstringhex — 40。对于阈值，值是octetstringhex — 20。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.4.3.116.111.109.116.114.105.103.103.101.114.49
octetstringhex "40"
```

[mteTriggerFrequency](#) —这确定秒钟数量等待在触发示例之间。最小值设置对象mteResourceSampleMinimum (默认是60秒)，降低此值增加CPU使用情况，因此必须仔细完成。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.11.3.116.111.109.116.114.105.103.103.101.114.49
gauge 60
```

[mteTriggerSampleType](#) —这些是absoluteValue (1)和deltaValue (2)。在这种情况下，值绝对

```
:
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.5.3.116.111.109.116.114.105.103.103.101.114.49
integer 1
```

[mteTriggerEnabled](#) —这是允许没被使用的触发将配置，但是的控制。设置它对真(默认是错误)。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.14.3.116.111.109.116.114.105.103.103.101.114.49
integer 1
```

即然触发如创建，定义了事件触发将使用。事件名称是event1。[mteEventEntryStatus](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.49
integer 6
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.49
integer 5
```

[mteEventActions](#) —这些是通知(0)和集(1)。进程是相同的象为mteTriggerTest。通知是10xxxxxxx和集是01xxxxxxx。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.3.3.116.111.109.101.118.101.110.116.49
octetstringhex "80"
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.4.3.116.111.109.101.118.101.110.116.49
integer 1
```

此下一步定义了在对对象将执行的测验选择为trigger1。 [mteTriggerBooleanComparison](#) —这些是不同等(1)，等于(2)，(3)， lessOrEqual (4)，更加极大(5)和greaterOrEqual (6)。在这种情况下—等于。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.5.1.1.3.116.111.109.116.114.105.103.103.101.114.49
integer 2
```

[mteTriggerBooleanValue](#) —这是使用的值测验。如果值为

```
1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0是相等的到1，则情况符合。
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.5.1.2.3.116.111.109.116.114.105.103.103.101.114.49
integer 1
```

现在请定义用事件将传送的对象。 [mteTriggerBooleanObjectsOwner](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.5.1.4.3.116.111.109.116.114.105.103.103.101.114.49
octetstring "tom"
```

[mteTriggerBooleanObjects](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.5.1.5.3.116.111.109.116.114.105.103.103.101.114.49
octetstring "objects1"
```

[mteTriggerBooleanEventOwner](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.5.1.6.3.116.111.109.116.114.105.103.103.101.114.49
octetstring "tom"
```

[mteTriggerBooleanEvent](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.5.1.7.3.116.111.109.116.114.105.103.103.101.114.49
octetstring "event1"
```

创建对象表。发送值为1.3.6.1.2.1.2.2.1.5.16作为VARBIND用陷阱。反对表[mteObjectsName](#) — Objects1。 [mteObjectsEntryStatus](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.3.1.1.5.3.116.111.109.8.111.98.106.101.99.116.115.49.1
integer 6
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.3.1.1.5.3.116.111.109.8.111.98.106.101.99.116.115.49.1
integer 5
```

[mteObjectsID](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.3.1.1.3.3.116.111.109.8.111.98.106.101.99.116.115.49.1
objectidentifier 1.3.6.1.2.1.2.2.1.5.16
```

[mteObjectsIDWildcard](#) —没有使用的通配符。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.3.1.1.4.3.116.111.109.8.111.98.106.101.99.116.115.49.1
integer 1
```

激活对象表。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.3.1.1.5.3.116.111.109.8.111.98.106.101.99.116.115.49.1
integer 1
```

附加对event1的对象。 [通知mteEventName](#) — Event1。 [mteEventNotificationObjectsOwner](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.3.1.2.3.116.111.109.101.118.101.110.116.49
octetstring "tom"
```

[mteEventNotificationObjects](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.3.1.3.3.116.111.109.101.118.101.110.116.49
octetstring "objects1"
```

激活触发。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.49
integer 1
```

激活事件。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.49
integer 1
```

[接收的陷阱](#)

```
# snmpset -v 2c -c private router 1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.49
integer 1
```

注意：对象6是被添加的VARBIND。

[创建Event2](#)

执行下列步骤：

1. [mteTriggerName](#) — Trigger2。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.50
integer 6
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.50
```

```
integer 5
```

2. [mteTriggerValueID](#) —这是第一个表达式和[mteTriggerValueIDWildcard](#)的值。这时，进程通配符ID，MIB对象的对象标识符采样确定触发是否射击。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.6.3.116.111.109.116.114.105.103.103.101.114.50
objectidentifier
1.3.6.1.4.1.9.10.22.1.4.1.1.2.2.0.0
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.7.3.116.111.109.116.114.105.103.103.101.114.50
integer 1
```

3. [mteTriggerTest](#) —阈值。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.4.3.116.111.109.116.114.105.103.103.101.114.50
octetstringhex "20"
```

4. [mteTriggerFrequency](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.11.3.116.111.109.116.114.105.103.103.101.114.50
gauge 60
```

5. [mteTriggerSampleType](#) —达美航空值。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.5.3.116.111.109.116.114.105.103.103.101.114.50
integer 2
```

6. [mteTriggerEnabled](#)

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.14.3.116.111.109.116.114.105.103.103.101.114.50
integer 1
```

7. 创建在事件表//[mteEventName](#)的一个事件— event2。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.50
integer 6
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.50
integer 5
```

8. [mteEventActions](#) —值40是为集，含义那，当情况符合时，路由器问题snmp set命令。在这种情况下，它做本身的集，但是在远程设备可能也做操作。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.3.3.116.111.109.101.118.101.110.116.50
octetstringhex "40"
```

9. 启用事件。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.4.2.1.4.3.116.111.109.101.118.101.110.116.50
integer 1
```

10. 设置在触发表//索引= [mteTriggerName](#)的触发阈值— Trigger2。因为它是阈值，请给失败的和上升的情况的值。花费仅上升的情况时间。

11. [mteTriggerThresholdDeltaRising](#) —这是检查的阈值。

```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.6.1.4.3.116.111.109.116.114.105.103.103.101.114.50
```



```
integer 100
```

12. [mteTriggerThresholdDeltaRisingEventOwner](#)

```
# snmpset -v 2c -c private router  
1.3.6.1.2.1.88.1.2.6.1.12.3.116.111.109.116.114.105.103.103.101.114.50  
octetstring "tom"
```

13. [mteTriggerThresholdDeltaRisingEvent](#)

```
# snmpset -v 2c -c private router  
1.3.6.1.2.1.88.1.2.6.1.13.3.116.111.109.116.114.105.103.103.101.114.50  
octetstring "event2"
```

14. [mteEventSetObject](#) —这是从设置的MIB对象的对象标识符。这里，回环接口的ifAdminStatus。

```
# snmpset -v 2c -c private router  
1.3.6.1.2.1.88.1.4.4.1.1.3.116.111.109.101.118.101.110.116.50  
objectidentifier 1.3.6.1.2.1.2.2.1.7.16
```

15. [mteEventSetValue](#) —这是设置的值(2下来的)。

```
# snmpset -v 2c -c private router  
1.3.6.1.2.1.88.1.4.4.1.3.3.116.111.109.101.118.101.110.116.50  
integer 2
```

16. 激活触发。

```
# snmpset -v 2c -c private router  
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.50  
integer 1
```

17. 激活事件。

```
# snmpset -v 2c -c private router  
1.3.6.1.2.1.88.1.4.2.1.5.3.116.111.109.101.118.101.110.116.50  
integer 1
```

结果

```
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
```

注意：这里，10.48.71.71是路由器的地址。

验证

此部分提供信息使用确认配置是工作正常。

[命令输出解释程序工具](#) ([仅限注册用户](#)) 支持某些 **show** 命令，使用此工具可以查看对 **show** 命令输出的分析。

```
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.22.1.4.1.1.2.2.0.0.0, 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: 62000000
(1): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.2 , val: 4000000
(2): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.3 , val: 617600
(3): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.4 , val: 617600
(4): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.5 , val: 617600
(5): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.6 , val: 617600
(6): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.7 , val: 858993458
(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: 62000000
(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: 62000000
(11): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.12 , val: 858993458
(12): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.13 , val: 858993458
(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: 3600
(15): Thresh: , Exis: 1, Read: 0, OID: ciscoExperiment.22.1.4.1.1.2.2.0.0.16 , val: 25600

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: elexp 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

[故障排除](#)

此部分提供信息使用排除故障配置。

[故障排除命令](#)

这些是命令启用调试：

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

注意： 在发出 `debug` 命令之前，请参阅[有关 debug 命令的重要信息](#)。

[相关信息](#)

- [表达式MIB : RFC 2982](#)
- [事件MIB : RFC 2981](#)
- [EXPRESSION-MIB.my/EVENT-MIB.my](#)
- [IOS功能指南：事件MIB支持](#)
- [技术支持 - Cisco Systems](#)