Konfigurationsbeispiel für Expression MIB und Ereignis-MIB

Inhalt

Einführung

Dieses Dokument zeigt, wie Expression MIB und Event MIB für die Verwendung in der Fehlerverwaltung kombiniert werden. Das enthaltene Beispiel ist nicht realistisch, zeigt aber viele verfügbare Funktionen.

Der Router muss zwei Aktionen ausführen:

1. Senden eines Traps, wenn eine Loopback-Schnittstelle eine Bandbreite von mehr als 100 hat und administrativ deaktiviert ist
2. Die Loopback-Schnittstelle wird deaktiviert, wenn die Bandbreitenanweisung einer Schnittstelle von einem definierten Wert geändert wurde

Das Beispiel zeigt Bandbreite und Admin-Status, da sie über die Befehlszeile leicht zu bearbeiten sind und sowohl Integer- als auch boolesche Werte anzeigen.

Die Befehle in diesem Dokument verwenden den OID-Parameter (Object Identifier) und nicht die Objektnamen. Dies ermöglicht Tests, ohne die MIB zu laden.

Voraussetzungen

Anforderungen

Bevor Sie die Informationen in diesem Dokument verwenden, stellen Sie sicher, dass Sie die

folgenden Voraussetzungen erfüllen:

- Die Workstation sollte über SNMP-Tools (Simple Network Management Protocol) verfügen, die von HP (Hewlett-Packard) OpenView bereitgestellt werden. Andere SNMP-Tools funktionieren, können aber eine andere Syntax aufweisen.
- Auf dem Gerät muss die Cisco IOS® Softwareversion 12.2(4)T3 oder höher ausgeführt werden. Ältere Versionen unterstützen die RFC-Version der Event MIB nicht.
- Die Plattform muss die Event MIB unterstützen. Eine Liste der unterstützten Plattformen für Cisco IOS Software, Version 12.1(3)T, finden Sie im Abschnitt "Unterstützte Plattform" unter Event MIB Support.

Verwendete Komponenten

Die Informationen in diesem Dokument basieren auf den folgenden Software- und Hardwareversionen:

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

Die Informationen in diesem Dokument wurden von den Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen.

Konventionen

Weitere Informationen zu Dokumentkonventionen finden Sie in den Cisco Technical Tips Conventions.

Hintergrundinformationen

- Mithilfe der Expression MIB kann der Benutzer ein eigenes MIB-Objekt erstellen, das auf einer Kombination anderer Objekte basiert. Weitere Informationen finden Sie unter RFC 2982.
- Die Ereignis-MIB ermöglicht dem Benutzer, dass das Gerät seine eigenen MIB-Objekte überwacht und Aktionen (Benachrichtigungen oder **SNMP-SET-**Befehle) auf Basis eines definierten Ereignisses generiert. Weitere Informationen finden Sie unter RFC 2981.

Konfigurieren

Hinweis: Einige Zeilen Ausgabecode werden über zwei Zeilen angezeigt, damit sie besser auf Ihren Bildschirm passen.

In diesem Beispiel ist ifIndex der Loopback-Schnittstelle gleich 16.

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

Die Variablennamen für das erste Ereignis beginnen mit `e1` und die Variablennamen für den

zweiten Start mit `e2`. Der Router-Name lautet "Router", und der Community-String für Lese-/Schreibzugriff lautet "privat".

Expression MIB

Erstellen des Ausdrucks 1

Erstellen Sie zuerst einen Ausdruck, der den Wert 1 zurückgibt, wenn die Bedingung `bei Geschwindigkeiten über 100.000` UND `wennAdminStatus` für die Loopback-Schnittstelle `ausgefallen ist`. Wenn die Bedingung nicht erfüllt wird, wird der Wert 0 zurückgegeben.

1. expExpressionDeltaInterval: Dieses Objekt wird nicht verwendet.Es gibt keinen Grund, einen Ausdruck zu berechnen, wenn er nicht abgefragt wird. Wenn kein Wert festgelegt ist, wird der Ausdruck bei der Abfrage des Objekts berechnet.Der Ausdrucksname lautet `e1exp`, der in der ASCII-Tabelle `101 49 101 120 112` entspricht.
2. expNameStatus: Dadurch wird ein eventuell erstellter alter Ausdruck zerstört.
   ```
   # 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: Erstellen und Warten.
   ```
   # 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: Hiermit wird der Index erstellt, der später verwendet werden soll, um das Ergebnis des Ausdrucks abzurufen.
   ```
   # 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: Hier .1 (der ausgewählte expExpressionIndex) ist die Beschreibung des Ausdrucks.
   ```
   # 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 - Dies ist der Ausdruck selbst, die Variablen $1 und $2 werden im nächsten Schritt definiert.Die einzigen zulässigen Operatoren sind (weitere Informationen finden Sie in 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 < 100000 && $2 == 2'
   ```
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 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: Die beiden Werte werden in absoluten Werten angenommen (für Delta wird als Wert 2 verwendet).

```
# 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: Die Objekt-IDs sind nicht verworfen. Dies ist der Standardwert, also nicht snmpset expObjectIDWildcard.
10. expObjectStatus: Legen Sie die Zeilen in der expObjectTable auf active fest.
    ```
    # 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. Aktivieren Sie den Ausdruck 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
    ```

Testen des Ausdrucks 1

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

1. Wenn die Bedingung erfüllt ist, ist der Wert von expValueCounter32Val 1 (da der Wert von expExpressionValueType unverändert bleibt, ist das Ergebnis ein counter32).**Hinweis:** Beim Typ darf es sich nicht um einen Gleitkommawert handeln.
   ```
   # 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. Wenn die Bedingung nicht erfüllt ist, ist der Wert 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. Wenn die Bedingung nicht erfüllt ist, ist der Wert 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
   ```

Erstellen und Testen von Ausdruck 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: Dies zeigt an, dass `1.3.6.1.2.1.2.2.1.5` eine Tabelle und kein Objekt

ist.

```
# 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. 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
```

Ereignis-MIB

Erstellen von Ereignissen 1

Erstellen Sie jetzt ein Ereignis, das den Ausgabewert des ersten Ausdrucks alle 60 Sekunden überprüft und ihn mit einem Verweis vergleicht. Wenn der Verweis mit dem Ausdruckswert übereinstimmt, wird ein Trap mit dem ausgewählten VARBIND ausgelöst.

1. Erstellen Sie den Trigger in der Triggertabelle.Der Name des Triggers lautet trigger1, der im ASCII-Code `116 114 105 103 103 101 114 49` lautet.Der Besitzer ist Tom: `116 111 109`.Der Index des metTriggerEntry besteht aus dem Triggereigner und dem Triggernamen. Der erste Wert des Index gibt die Anzahl der Zeichen für den datoOwner an.In diesem Fall gibt es drei Zeichen für "tom". Der Index ist also: `3.116.111.109.116.114.105.103.103.101.114.49`
2. Falls vorhanden, den alten Eintrag zerstören.
3. Legen Sie den Triggerstatus zum **Erstellen und Warten fest**.
4. Im letzten Schritt wird sie aktiviert: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
```

metTriggerValueID: Der Wert des ersten Ausdrucks ist `e1exp`.Der Objekt-Identifikator des MIB-Objekts ist der Objekt, der überprüft werden soll, ob der Trigger ausgelöst werden soll.

```
# 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
```

metTriggerValueIDWildcard - Ohne einen Platzhalter für die Wert-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
```

metTriggerTest - Existence (0), boolean (1) und threshold (2).Die Methode zur Auswahl eines der oben genannten Werte ist komplex. Geben Sie zur Auswahl einer Existenz einen Wert in acht Ziffern ein, wobei die erste Ziffer eine 1 ist, z. B. `1000000` oder `100xxxxx`.Bei einem booleschen Wert muss die zweite Ziffer eine 1 sein: `0100000` oder `010xxxxx`.Bei einem Schwellenwert muss die dritte Ziffer eine 1 sein: `00100000` oder `001xxxxx`.Dies ist ganz einfach:Für Existenz ist der Wert octetstringhex - 80.Für boolean lautet der Wert Octetstringhex—40.Für den Schwellenwert ist der Wert 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"
```

metTriggerFrequency (Frequenz): Diese Eigenschaft bestimmt die Wartezeit in Sekunden zwischen den Triggerbeispielen.Der minimale Wert wird mit dem Objekt mteResourceSampleMinimum (Standardwert ist 60 Sekunden) festgelegt. Durch diesen Wert wird die CPU-Auslastung erhöht, daher muss er sorgfältig durchgeführt werden.

```
# 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
```

metTriggerSampleType: Dies sind absoluteValue (1) und deltaValue (2). In diesem Fall ist der Wert absolut:

```
# 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: Dies ist ein Steuerelement, mit dem ein Trigger konfiguriert, aber nicht verwendet werden kann. Legen Sie true fest (Standardwert ist false).

```
# 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
```

Nachdem der Trigger erstellt wurde, definieren Sie das Ereignis, das der Trigger verwendet. Der Ereignisname lautet event1.metEventEntryStatus

```
# 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
```

metEventActions: Diese sind Benachrichtigung (0) und set (1).Der Prozess ist der gleiche wie für metTriggerTest. Die Benachrichtigung lautet `10xxxxxx` und lautet `01xxxxxx`.

```
# 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
```

Im nächsten Schritt wird der Test für das für trigger1 ausgewählte Objekt definiert.mteTriggerBooleanComparison: Diese sind ungleich (1), gleich (2), kleiner (3), lessOrEqual (4), größer (5) und größerOrEqual (6). In diesem Fall ist gleich.

```
# 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
```

metTriggerBooleanValue: Dieser Wert wird für den Test verwendet. Wenn der Wert von `1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0` 1 ist, ist die Bedingung erfüllt.

```
# 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
```

Definieren Sie nun das Objekt, das mit dem Ereignis gesendet werden soll.metTriggerBooleanObjectsOwner

```
# 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"
```

metTriggerBooleanObjects

```
# 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"
```

metTriggerBooleanEventOwner

```
# 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"
```

metTriggerBooleanEvent

```
# 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"
```

Erstellen Sie die Objekttabelle.Senden Sie den Wert von `1.3.6.1.2.1.2.2.1.5.16` als VARBIND mit dem Trap.Object Table metObjectsName - Objects1.metObjectsEntryStatus

```
# 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
```

metObjectsID

```
# 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
```

metObjectsIDWildcard: Es wird keine Platzhalterkarte verwendet.

```
# 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
```

Aktivieren Sie die Objekttabelle.

```
# 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
```

Hängen Sie das Objekt an event1 an.Notify metEventName - Event1.metEventNotificationObjectsOwner

```
# 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"
```

metEventNotificationObjects

```
# 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"
```

Aktivieren Sie den Trigger.

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

Aktivieren Sie das Ereignis.

```
# 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
```

Trap erhalten

```
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
```

Hinweis: Objekt 6 ist die hinzugefügte VARBIND.

Erstellen von Event 2

Gehen Sie folgendermaßen vor:

1. metTriggerName - 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. metTriggerValueID: Dies ist der Wert des ersten Ausdrucks und der meTriggerValueIDWildcard.Diesmal erstellt der Prozess einen Platzhalter für die Wert-ID, die Objekt-ID des MIB-Objekts, um zu bestimmen, ob der Trigger ausgelöst wird.

   ```
   # 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. metTriggerTest - Grenzwert.
```
# 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. metTriggerFrequenz
```
# 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. metTriggerSampleType - Delta-Wert.
```
# 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. metTriggerEnabled
```
# 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. Erstellen Sie ein Ereignis in der Ereignistabelle // metEventName - 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. metEventActions: Der Wert 40 ist für Set, d. h., wenn die Bedingung erfüllt ist, gibt der Router einen Befehl **snmp set aus**. In diesem Fall stellt es das Set selbst her, kann aber auch den Betrieb auf einem Remote-Gerät durchführen.
```
# 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. Aktivieren Sie das Ereignis.
```
# 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. Legen Sie den Trigger-Schwellenwert in der Triggertabelle // index = mteTriggerName - Trigger2 fest.Da es sich um einen Schwellenwert handelt, geben Sie Werte für ausgefallene und steigende Bedingungen an. Nehmen Sie dieses Mal nur den steigenden Zustand.
11. metTriggerThresholdDeltaRising: Dieser Schwellenwert muss überprüft werden.
```
# 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. metTriggerThresholdDeltaRisingEventOwner
```
# 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. metTriggerThresholdDeltaRisingEvent
```
# 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. metEventSetObject: Dies ist der Objektidentifikator aus dem einzustellenden MIB-Objekt. Hier, ifAdminStatus für die Loopback-Schnittstelle.
```
# 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. metEventSetValue: Dieser Wert wird festgelegt (2 für nicht verfügbar).
```
# 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. Aktivieren Sie den Trigger.
```
# snmpset -v 2c -c private router
1.3.6.1.2.1.88.1.2.2.1.15.3.116.111.109.116.114.105.103.103.101.114.50
integer 1
```

17. Aktivieren Sie das Ereignis.
```
# 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
```

Ergebnis

```
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
```

Hinweis: Hier ist die `10.48.71.71` die Adresse des Routers selbst.

Überprüfen

Dieser Abschnitt enthält Informationen zur Bestätigung, dass die Konfiguration ordnungsgemäß funktioniert.

Bestimmte **show**-Befehle werden vom Output Interpreter Tool unterstützt (nur registrierte Kunden), mit dem Sie eine Analyse der **show**-Befehlsausgabe anzeigen können.

```
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: 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: 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
```

Fehlerbehebung

Dieser Abschnitt enthält Informationen zur Fehlerbehebung bei der Konfiguration.

Befehle zur Fehlerbehebung

Dies sind die Befehle zum Aktivieren des Debuggens:

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

Hinweis: Bevor Sie **Debugbefehle** ausgeben, lesen Sie die Informationen Wichtige Informationen über Debug-Befehle.

Zugehörige Informationen

- Ausdruck MIB: RFC 2982
- Ereignis-MIB: RFC 2981
- EXPRESSION-MIB.my/EVENT-MIB.my
- IOS-Funktionsleitfaden: Ereignis-MIB-Unterstützung
- Technischer Support - Cisco Systems