

Expression MIB と Event MIB の設定例

内容

[概要](#)

[前提条件](#)

[要件](#)

[使用するコンポーネント](#)

[表記法](#)

[背景説明](#)

[設定](#)

[式 MIB](#)

[イベント MIB](#)

[確認](#)

[トラブルシュート](#)

[トラブルシューティングのためのコマンド](#)

[関連情報](#)

概要

このドキュメントでは、障害管理に使用する Expression MIB と Event MIB を結合する方法について説明します。付属の例は、現実的ではありませんが、多くの使用可能な機能を示しています。

ルータは、次の 2 つの操作を実行する必要があります。

1. ループバック インターフェイスに 100 を超える帯域幅がある場合はトラップを送信し、および管理上ダウンする
2. インターフェイスの 1 つに定義済みの値から変更された帯域幅ステートメントがある場合、ループバック インターフェイスはシャット ダウンする

例は、コマンドラインからの操作が容易であり、両方とも整数とブール値を表示するので、帯域幅と管理ステータスによって示されます。

このドキュメントのコマンドは、オブジェクト ID (OID) パラメータを使用し、オブジェクト名を使用しません。これにより、MIB をロードせずにテストができます。

前提条件

要件

このドキュメントの情報を使用する前に、次の前提条件を満たしていることを確認してください。

- ワークステーションは、Hewlett-Packard (HP) OpenView によって提供される簡易ネットワーク管理プロトコル (SNMP) のツールを備えている必要があります。その他の SNMP ツールは動作しますが、構文が異なる場合があります。
- デバイスは、Cisco IOS® ソフトウェア リリース 12.2(4)T3 以降を実行する必要があります。以前のバージョンは、イベント MIB の RFC バージョンをサポートしていません。
- プラットフォームは、イベント MIB をサポートする必要があります。Cisco IOS ソフトウェア リリース 12.1(3)T でサポートされているプラットフォームのリストについては、[イベント MIB サポート](#)の「サポート対象プラットフォーム」セクションを参照してください。

[使用するコンポーネント](#)

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- Cisco IOS ソフトウェア リリース 12.3(1a)
- Cisco 3640 モジュラ アクセス ルータ

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、初期 (デフォルト) 設定の状態から起動しています。対象のネットワークが実稼働中である場合には、どのようなコマンドについても、その潜在的な影響について確実に理解しておく必要があります。

[表記法](#)

ドキュメント表記の詳細は、「[シスコテクニカルティップスの表記法](#)」を参照してください。

[背景説明](#)

- Expression MIB では、ユーザが他のオブジェクトの組み合わせに基づき、独自の MIB オブジェクトを作成することができます。詳細については、[RFC 2982](#) を参照してください。
- イベント MIB では、ユーザが独自の MIB オブジェクトをモニタリングするデバイスを備え、定義済みイベントに基づいてアクション (通知または SNMP SET コマンド) を生成することができます。詳細については、[RFC 2981](#) を参照してください。

[設定](#)

注：出力コードの一部の行は、画面に収まるように2行にわたって表示されます。

この例では、ループバック インターフェイスの ifIndex は 16 になります。

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

e1で始まる最初のイベントに関連する変数名と、e2router/private

[式 MIB](#)

[式 1 の作成](#)

最初に、条件がグループバック インターフェイスに対して `ifSpeed 100,000 ifAdminStatus 1` 条件が満たされない場合、値 0 を返します。

1. [expExpressionDeltaInterval](#) - このオブジェクトは使用されません。それがポーリングされない場合、表現を計算する必要はありません。No 値が設定される場合、オブジェクトが照会されるときに、式が計算されます。式の名前は `e1exp` であり、ASCII テーブルの 101 49 101 120 112 に対応します。

2. [expNameStatus](#) - これは作成される最後の古い式を破棄します。

```
# 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](#) - 作成し、待機します。

```
# 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](#) - これは、式の結果を取得するために、後で使用するインデックスを作成します。

```
# 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](#) : ここで、.1 (選択した `expExpressionIndex`) は式の説明です。

```
# 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](#) - これは、式そのものであり、変数 \$1 および \$2 は、次の手順で定義されます。使用できる演算子は次のとおりです(詳細については、[RFC 2982](#)を参照してください)。

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

```
# snmpset -v 2c -c private router 1.3.6.1.4.1.9.10.22.1.3.1.1.2.1 octetstring '$1 < 10000 && $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](#):2つの値は絶対値で取得されます (Deltaの場合は2を値として使用)。

```
# 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](#) - オブジェクト ID にワイルドカードは使用されません。これはデフォルト値なので、`snmpset expObjectIDWildcard` を実行しません。

10. [expObjectStatus](#) - [expObjectTable](#) の行をアクティブに設定します。

```
# 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. 式 1 をアクティブにします。

```
# 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
```

式 1 のテスト

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

1. 条件が満たされる場合、[expValueCounter32Val は値 1 を返します \(expExpressionValueType の値を変更されません。結果は counter32 です \)](#)。注記：型は浮動小数点値にできません。

```
# 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. 条件が満たされない場合、値は 0 です。

```
# 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. 条件が満たされない場合、値は 0 です。

```
# 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
```

式 2 の作成およびテスト

```
# 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](#) - これは、[1.3.6.1.2.1.2.2.1.5](#) がテーブルであり、オブジェクトではないことを示します。

```
# 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. テスト：

```
# 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
```

イベント MIB

イベント 1 の作成

ここで、60 秒ごとに最初の式の出力値を確認し、基準と比較するイベントを作成します。基準が式の値に一致する場合、VARBIND を選択するとトラップがトリガーされます。

1. トリガー テーブルにトリガーを作成します。トリガーの名前は trigger1 であり、ASCII コードでは、116 114 105 103 103 101 114 49 です。オーナーは Tom : 116 111 109 です。mteTriggerEntry のインデックスは、トリガーのオーナーとトリガーの名前で構成されます。インデックスの最初の値は mteOwner の文字数を示します。この場合、Tom の 3 文字があるため、インデックスは次のようになります。
3.116.111.109.116.114.105.103.103.101.114.49
2. 古いエントリが存在する場合、破棄されます。
3. トリガーのステータスを、**create and wait** に設定します。
4. 最後の手順では、これをアクティブにします。[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](#) – 最初の式の値は e1exp。MIB オブジェクトのオブジェクト ID は、トリガーが発生するかどうかを確認するためにサンプリングする 1 つです。

```
# 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](#) - 値 ID にワイルドカードは使用されません。

```
# 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](#) - 存在値 (0)、ブール値 (1)、おおよしきい値 (2)。上記の値いずれかを選択する方法は複雑です。存在値を選択するには、10000000 100xxxxxx など、先頭が 1 となる 8 桁の値を指定します。ブール値の場合、次のように 2 桁目が 1 となる必要があります。01000000 010xxxxxx。しきい値の場合、次のように 3 桁目が 1 となる必要があります。00100000 001xxxxxx。次の方法による操作が最も簡単です。存在値の場合、値が octetstringhex - 80。ブール値の場合、値が octetstringhex - 40。しきい値の場合、値が 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](#) - これは、トリガー サンプル間で待機する秒数を指定します。最小値は、mteResourceSampleMinimum オブジェクト (デフォルトは 60 秒) を使用して設定します。この値を下げると CPU の使用率が增加するので、慎重に設定する必要があります。

```
# 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](#) - これらは [absoluteValue \(1 \)](#) と [deltaValue \(2 \)](#) です。この場合、値は絶対値です。

```
# 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](#) - これはトリガーを可能にするために設定できるが、使用されないコントロールです。true 設定します (デフォルトは false です) 。

```
# 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
```

ここで、トリガーが作成されたので、トリガーが使用するイベントを定義します。イベント名は event1 です。 [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](#) - これらは [通知 \(0 \)](#) と [設定 \(1 \)](#) です。プロセスは mteTriggerTest の場合と同じです。通知は 10xxxxxxx 01xxxxxxx です。

```
# 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
```

次の手順では、trigger1 で選択されたオブジェクトで実行するテストを定義します。

[mteTriggerBooleanComparison](#) - これらは、[unequal \(1 \)](#)、[equal \(2 \)](#)、[less \(3 \)](#)、[lessOrEqual \(4 \)](#)、[greater \(5 \)](#)、および [greaterOrEqual \(6 \)](#) です。この場合、equal です。

```
# 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](#) - これはテストに使用する値です。

```
1.3.6.1.4.1.9.10.22.1.4.1.1.2.1.0.0.0 1
# 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
```

ここで、イベントによって送信するオブジェクトを定義します。

[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"
```

オブジェクトテーブルを作成します。1.3.6.1.2.1.2.2.1.5.16 VARBIND [オブジェクトテーブル](#)
[/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 - ワイルドカードは使用されません。](#)

```
# 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
```

オブジェクトテーブルをアクティブにします。

```
# 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
```

オブジェクトを event1 にアタッチします。 [Notify mteEventName:Event1](#)。

[mteEventNotificationObjectsOwner](#)

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

トリガーをアクティブにします。

```
# 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
```

イベントをアクティブにします。

```
# 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
```

受信されたトラップ

```
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
```

注：オブジェクト6は、追加されたVARBINDです。

イベント2の作成

手順は以下のとおりです。

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 - これは、最初の式と mteTriggerValueIDWildcard の値です。](#)今回は、プロセスがワイルドカードを使用して、値 ID、MIB オブジェクトのオブジェクト ID をサンプリングし、トリガーが起動するかどうかを決定します。

```
# 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 - しきい値。](#)

```
# 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](#) - デルタ値。

```
# 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. イベントテーブル// [mteEventName](#) - event2にイベントを作成します。

```
# 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](#) - 値 40 は設定用です。つまり、条件が満たされる場合、ルータは [snmp set](#) コマンドを発行します。この場合は、それ自体に設定しますが、リモート デバイスで動作する可能性もあります。

```
# 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. イベントを有効にします。

```
# 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. トリガーテーブルのトリガーしきい値// index = [mteTriggerName](#) - Trigger2を設定します。これはしきい値なので、機能停止および機能回復条件の値を指定します。今回は、機能回復条件のみ取り上げます。

11. [mteTriggerThresholdDeltaRising](#) - これは、確認されるしきい値です。

```
# 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](#) - これは、設定する MIB オブジェクトのオブジェクト ID です。ここでは、ループバック インターフェイスの 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 - これは設定する値です \(下方の場合 2 \)](#)。

```
# 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. トリガーをアクティブにします。

```
# 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. イベントをアクティブにします。

```
# 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
```

結果

```
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: ここで、10.48.71.71は、ルータ自体のアドレスです。

確認

このセクションでは、設定が正しく動作していることを確認するために使用する情報を提供します。

一部の show コマンドは[アウトプット インタープリタ ツールによってサポートされています \(登録ユーザ専用 \)](#)。このツールを使用することによって、show コマンド出力の分析結果を表示できます。

```
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

トラブルシュート

このセクションでは、設定のトラブルシューティングに使用する情報を提供します。

トラブルシューティングのためのコマンド

次は、デバッグを有効にするコマンドです。

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

注：debugコマンドを発行する前に、『[debugコマンドの重要な情報](#)』を参照してください。

関連情報

- [式 MIB : RFC 2982](#)
- [イベント MIB RFC 2981](#)
- [EXPRESSION-MIB.my / EVENT-MIB.my](#)
- [IOS 機能ガイド：イベント MIB のサポート](#)
- [テクニカルサポート - Cisco Systems](#)