APPENDIX A

MIB の使用

この章では、Cisco ASR 1000 シリーズ ルータ上でタスクを実行する方法について説明します。

Cisco Unique Device Identifier のサポート

ENTITY-MIB は、現在、IDPROM に保存されている Cisco Unique Device Identifier（UDI）規格に対するシスコのコンプライアンスへの取り組みをサポートします。

Cisco UDI は各シスコ製品に一意の ID を提供します。UDI は、entPhysicalTable に保存する必要がある 3 つの個別のデータ要素で構成されます。

- 注文可能な製品 ID（PID）：製品 ID（PID）。PID はシスコ製品を注文するためにお客様が使用する英数字の識別子です。2 つの例として、NM-1FE-TX と CISCO3745 があります。PID は 18 文字に制限され、entPhysicalModelName オブジェクトに保存する必要があります。
- バージョン ID（VID）：バージョン ID（VID）。VID は PID のバージョンです。VID はお客様に報告される方法で製品がバージョン化された回数を示します。たとえば、製品 ID NM-1FE-TX の VID は V04 である可能性があります。VID は、英数字 3 文字に制限され、entPhysicalHardwareRev オブジェクトに保存する必要があります。

- シリアル番号（SN）：シリアル番号は、製品内の特定の部品を識別するために使用される 11 文字の ID で、entPhysicalSerialNum オブジェクトに保存する必要があります。シリアル番号の内容は、製造部品番号 7018060-0000 によって定義されます。SN には、次の Web サイトで部品番号 701806-0000 を検索してアクセスします。

  https://mco.cisco.com/servlet/mco.ecm.inbiz

  シリアル番号の形式は、4 つのフィールドで定義されます。

  - 場所（L）
  - 年（Y）
  - 週労働日（W）
  - 連続したシリアル ID（S）

  SN ラベルは、LLLYYWWSSS と表されます。

（注）　バージョン ID は IDPROM にバージョン ID フィールドがない古いカードまたは既存のカードについて NULL を戻します。したがって、対応する entPhysicalHardwareRev は、IDPROM にバージョン ID フィールドがないカードについては NULL を戻します。

シスコの冗長性機能

この章の内容は次のとおりです。

冗長性により、データ要素とソフトウェア機能が複製され、障害時に代替手段が提供されます。シスコの冗長性機能の目的は、各インターフェイスに関連付けられたリンクとプロトコル状態に影響を与えずにカットオーバーし、パケット転送を続けることです。インターフェイスおよびサブインターフェイスの状態は、ラインカードやさまざまなパケット処理ハードウェアの状態とともに保持されます。

冗長性のレベル

ここでは、Cisco ASR 1000 シリーズ ルータでサポートされている冗長性のレベルおよびこの機能を使用できることを確認する方法について説明します。Cisco ASR 1000 シリーズ ルータでは、アクティブなスーパーバイザ エンジンが故障した場合にシスコの冗長スーパーバイザ エンジン（SE）が処理を引き継ぐようにすることによって、障害耐性をサポートします。冗長性は、装置の障害によってサービスが停止することを防止し、中断のないメンテナンスとアップグレードのアクティビティをサポートします。インターフェイスおよびサブインターフェイスの状態は、ラインカードやさまざまなパケット処理ハードウェアの状態とともに保持されます。

冗長システムは 2 台のルート プロセッサをサポートします。1 台はアクティブなルート プロセッサとして動作し、もう一方はスタンバイ ルート プロセッサとして動作します。

Route Processor Redundancy 機能は、次のいずれかの状況が発生した場合に、スタンバイ ルート プロセッサへの切り替えによって Cisco ルータにハイ アベイラビリティを提供します。

- Cisco IOS ソフトウェア障害
- Cisco ASR 1000 シリーズ ルート プロセッサ（RP）のハードウェア障害
- ソフトウェア アップグレード
- メンテナンス手順

Cisco ASR 1000 シリーズ ルータは、次の 2 つの冗長モードのいずれかで動作できます。

- Route Processor Redundancy（RPR）モード
- Nonstop Forwarding/Stateful Switchover（NSF/SSO）モード

すべてのモードで、アクティブ RP に障害が発生するとスタンバイ RP が処理を引き継ぎます。

Route Processor Redundancy

ここでは、Cisco ASR 1000 シリーズ ルータの Route Processor Redundancy（RPR）モードについて説明します。

スイッチの電源投入時に、2 つのシスコ スーパーバイザ エンジン間で RPR が稼働します。最初に起動するスーパーバイザ エンジンは、RPR アクティブスーパーバイザ エンジンになります。

Cisco ASR 1000 シリーズ ルータでは、アクティブなスーパーバイザ エンジンが故障した場合に冗長スーパーバイザ エンジンが処理を引き継ぐようにすることによって、障害耐性をサポートします。

Cisco Nonstop Forwarding and Stateful Switchover

ここでは、Cisco Nonstop Forwarding and Stateful Switchover モードについて説明します。NSF/SSO を使用すると、Cisco ASR 1000 シリーズ ルータは、ほとんどすぐにアクティブからスタンバイ ルート プロセッサにフェールオーバーし、パケットを転送し続けます。このプラットフォームの Cisco IOS ソフトウェア NSF/SSO のサポートは、即時のフェールオーバーをイネーブルにします。

NSF/SSO が動作するネットワーキング デバイスでは、アクティブ RP に障害が発生した後、スタンバイ RP がいつでも制御を引き継げるように、両方の RP で同じコンフィギュレーションを実行する必要があります。起動時およびアクティブ RP のコンフィギュレーションに変更が生じるたびに、アクティブ RP からスタンバイ RP にコンフィギュレーション情報が同期されます。

2 つのプロセッサ間の初期同期後に、NFS/SSO は転送情報などの両者間の RP ステート情報を維持します。

Cisco Nonstop Forwarding（NSF）とステートフル スイッチオーバー（SSO）を組み合わせることにより、デュアル RP を搭載したルータにおけるルータ プロセッサ（RP）のフェールオーバー後に、ユーザがネットワークを使用できない時間が最小限に抑えられます。NSF/SSO 機能を使用すると、ルータは、スイッチオーバーを検出し、ネットワーク トラフィックを転送し続けてピア デバイスからルート情報を回復するために必要なアクションを実行できます。

Cisco IOS ソフトウェアで Cisco NSF とステートフル スイッチオーバー（SSO）機能を組み合わせることにより、スイッチオーバー後に、ユーザがネットワークを使用できない時間が最小限に抑えられます。Cisco NSF/SSO の主な目的は、ルーティング プロトコル情報がルートのスイッチオーバー後に復元される間、既知のルートでデータ パケットの転送を継続することです。

（注）　Nonstop Forwarding 機能に関する詳細については、次を参照してください。
http://www.cisco.com/en/US/docs/ios/12_2s/feature/guide/fsnsf20s.html

（注） ステートフル スイッチオーバー機能に関する詳細については、次を参照してください。
http://www.cisco.com/en/US/docs/ios/12_2s/feature/guide/fssso20s.html

ソフトウェア冗長性

RP スロットが 1 つだけ搭載された Cisco ASR 1004 ルータは、ハードウェア冗長性をサポートしていません。代わりに、これらのルータには、2 つの IOSD プロセスを実行することによるソフトウェア冗長性というオプションがあります。IOSD は、必要に応じて冗長構成で実行できます。1 つの IOSD インスタンスがアクティブで、他のインスタンスはホット スタンバイ モードで維持されます。ステート情報は IPC 上の標準の SSO サポートを使用してインスタンス間で交換されます。ソフトウェア冗長性のオプションは、2 番目の RP がシャーシに追加されている場合は使用できず、非アクティブになります。アクティブ RP がアクティブとスタンバイの両方の FP およびすべての I/O（キャリア）カードを制御します。アクティブ IOSD インスタンスが失敗した場合、バックアップが処理を引き継ぎ、FP および I/O カードと状態を再同期します。

Cisco ASR 1000 シリーズ ルータの冗長性の確認

Cisco ASR 1000 シリーズ ルータにインストールされたアクティブおよびスタンバイ スーパーバイザ エンジンに関する情報を表示するには、**show** redundancy コマンドおよび **show redundancy states** コマンドを使用します。R0 スロットのルータ プロセッサの場合、ユニット ID の値は、ASCII の「0」と同じ 48（16 進で 30）です。R1 スロットのルータ プロセッサの場合、ユニット ID の値は 49、ASCII の「1」（16 進で 31）です。

例 A-1　アクティブ プロセッサからの冗長状態の表示

```
R5-mcp-6ru-2#sh redundancy states
       my state = 13 -ACTIVE
     peer state = 8  -STANDBY HOT
           Mode = Duplex
        Unit ID = 48

Redundancy Mode (Operational) = sso
Redundancy Mode (Configured)  = sso
Redundancy State              = sso
     Maintenance Mode = Disabled
    Manual Swact = enabled
 Communications = Up

   client count = 66
 client_notification_TMR = 30000 milliseconds
           RF debug mask = 0x0

R5-mcp-6ru-2#exit
```

例 A-2　スタンバイ プロセッサからの冗長状態の表示

```
R5-mcp-6ru-2-stby#sh redundancy state
       my state = 8  -STANDBY HOT
     peer state = 13 -ACTIVE
           Mode = Duplex
        Unit ID = 49

Redundancy Mode (Operational) = sso
```

```
Redundancy Mode (Configured)  = sso
Redundancy State              = sso
     Maintenance Mode = Disabled
    Manual Swact = cannot be initiated from this the standby unit  Communications = Up

   client count = 67
 client_notification_TMR = 30000 milliseconds
           RF debug mask = 0x0

R5-mcp-6ru-2-stby#
```

例 A-3　　ソフトウェア冗長性の冗長状態の表示：ASR 1004

```
R5-mcp-4ru-1#sh redundancy states
       my state = 13 -ACTIVE
     peer state = 8  -STANDBY HOT
           Mode = Duplex
        Unit ID = 48

Redundancy Mode (Operational) = sso
Redundancy Mode (Configured)  = sso
Redundancy State              = sso
     Maintenance Mode = Disabled
    Manual Swact = enabled
 Communications = Up

   client count = 66
 client_notification_TMR = 30000 milliseconds
           RF debug mask = 0x0

R5-mcp-4ru-1#
```

関連情報および有益なリンク

次の URL から、シスコの冗長性機能に関する有用な情報にアクセスできます。

- Cisco Nonstop Forwarding に関する詳細情報：
  http://www.cisco.com/en/US/docs/ios/12_2s/feature/guide/fsnsf20s.html
- ステートフル スイッチオーバー機能に関する詳細情報：
  http://www.cisco.com/en/US/docs/ios/12_2s/feature/guide/fssso20s.html
- Route Processor Redundancy 機能に関する詳細情報：
  http://www.cisco.com/en/US/docs/ios/12_1/12_1ex/feature/guide/12e_rpr.html

物理エンティティの管理

この項では、SNMP を使用して次の方法でルータの物理エンティティ（コンポーネント）を管理する方法について説明します。

目的と利点

Cisco ASR 1000 シリーズ ルータの SNMP 実装の物理エンティティ管理機能では、次の作業を行います。

- 現場交換可能ユニット（FRU）のステータスのモニタリングと設定
- インターフェイス マッピングへの物理ポートについての情報の提供
- アセット タギングのアセット情報の提供
- シャーシ コンポーネントのファームウェアおよびソフトウェア情報の提供

物理エンティティ管理に使用する MIB

- CISCO-ENTITY-FRU-CONTROL-MIB：ENTITY-MIB の entPhysicalTable に挙げた電源やラインカードなどの現場交換可能ユニット（FRU）の管理ステータスおよび動作ステータスのモニタと設定に使用されるオブジェクトが含まれます。
- CISCO-ENTITY-EXT-MIB：ENTITY-MIB の entPhysicalTable に対するシスコ定義の拡張機能が含まれ、entPhysicalClass の値が「module」で CPU、RAM/NVRAM、コンフィギュレーション レジスタを搭載したエンティティの情報を提供します。
- CISCO-ENTITY-SENSOR-MIB および ENTITY-SENSOR-MIB：entPhysicalClass の値が「sensor」である entPhysicalTable のエンティティに関する情報が含まれます。
- CISCO-ENTITY-VENDORTYPE-OID-MIB：ルータのすべての物理エンティティのオブジェクト ID（OID）が含まれます。
- ENTITY-MIB：ルータの物理エンティティを管理するための情報が含まれます。また、階層と相互の関係を示す包含ツリーにエンティティを編成します。MIB には、次のテーブルがあります。
  - entPhysicalTable は、ルータの各物理コンポーネント（エンティティ）を記述します。テーブルには、シャーシのトップレベルのエンティティ（シャーシ）と各エンティティのエントリが含まれます。各エントリは、エンティティの情報（その名前、タイプ、ベンダーと説明）を提供し、エンティティがシャーシ エンティティの階層にどのように収まっているかを記述します。

    各エンティティは、この MIB や他の MIB 内のエンティティに関する情報へのアクセスに使用する一意のインデックス（*entPhysicalIndex*）によって識別されます。
  - entAliasMappingTable は、IF-MIB の ifTable の対応する ifIndex 値に各物理ポートの entPhysicalIndex 値をマッピングします。
  - entPhysicalContainsTable は、シャーシ内の物理エンティティ間の関係を示します。物理エンティティごとに、テーブルは、エンティティの各子オブジェクトの entPhysicalIndex を示します。
  - entPhysicalIsFRU は、物理エンティティが現場交換可能ユニット（FRU）と見なされるかどうかを示します。エンティティが FRU として識別される場合、物理エンティティに次のデバイス固有の情報が含まれます。
    - entPhysicalModelName：注文可能な製品番号と同じ製品 ID（PID）。
    - entPhysicalHardwareRev：バージョン ID（VID）
    - entPhysicalSerialNum：シリアル番号（SN）
    - Cisco Unique Device Identifier（UDI）：PID、VID、および SN で構成され、イネーブル化されているすべてのシスコ ハードウェア製品の一意の ID を提供します。

インベントリ管理の実行

ルータのエンティティに関する情報を取得するには、ENTITY-MIB entPhysicalTable に対する MIB ウォークを実行します。

ENTITY-MIB entPhysicalTable のサンプル エントリを検証する場合は、次の点を考慮します。

- entPhysicalIndex：シャーシの各エンティティを一意に識別します。このインデックスは、他の MIB のエンティティに関する情報へのアクセスにも使用されます。
- entPhysicalContainedIn：コンポーネントの親エンティティの entPhysicalIndex を示します。
- entPhysicalParentRelPos：同じ entPhysicalContainedIn 値を持つ同じタイプのエンティティ（たとえば、シャーシ スロット、およびラインカード ポート）の相対的な位置を示します。

（注） コンテナは、物理エンティティ クラスに 1 つ以上の着脱可能な物理エンティティを含めることができる場合に適用できます。たとえば、シャーシの各（空または完全な）スロットは、コンテナとしてモデル化されます。すべての着脱可能な物理エンティティは、現地交換可能なモジュール、ファン、電源などのコンテナ エンティティ内でモデル化する必要があります。

ENTITY-MIB entPhysicalTable のサンプル エントリ

ここに挙げるサンプルは、情報が entPhysicalTable にどのように保存されているかを示します。entPhysicalTable エントリを調べて、アセットのインベントリを実行できます。

（注） この章全体で示すサンプルの出力と値は、MIB を使用する場合に表示できるデータの例です。

次の表示は、カードに挿入されたルータ シャーシおよび 4 つの SPA に取り付けられている ASR1000 SIP-10 カードの ENTITY-MIB entPhysicalTable のサンプル エントリを示しています。

ENTITY-MIB entPhysicalTable のエントリ

```
entPhysicalDescr.1000 = Cisco ASR1000 SPA Interface Processor 10
entPhysicalDescr.1001 = V1: VMA
entPhysicalDescr.1002 = V1: VMB
entPhysicalDescr.1003 = V1: VMC
entPhysicalDescr.1004 = V1: VMD
entPhysicalDescr.1005 = V1: VME
entPhysicalDescr.1006 = V1: VMF
entPhysicalDescr.1007 = V1: 12v
entPhysicalDescr.1008 = V1: VDD
entPhysicalDescr.1009 = V1: GP1
entPhysicalDescr.1010 = V1: GP2
entPhysicalDescr.1011 = V2: VMB
entPhysicalDescr.1012 = V2: 12v
entPhysicalDescr.1013 = V2: VDD
entPhysicalDescr.1014 = V2: GP2
entPhysicalDescr.1015 = Temp: Left
entPhysicalDescr.1016 = Temp: Center
entPhysicalDescr.1017 = Temp: Asic1
entPhysicalDescr.1018 = Temp: Right
entPhysicalDescr.1026 = CPU 0 of module 0
entPhysicalDescr.1027 = SPA Bay
entPhysicalDescr.1028 = SPA Bay
entPhysicalDescr.1029 = SPA Bay
entPhysicalDescr.1030 = SPA Bay
.....
entPhysicalVendorType.1000 = cevModuleASR1000SIP10
```

```
entPhysicalVendorType.1001 = cevSensor
entPhysicalVendorType.1002 = cevSensor
entPhysicalVendorType.1003 = cevSensor
entPhysicalVendorType.1004 = cevSensor
entPhysicalVendorType.1005 = cevSensor
entPhysicalVendorType.1006 = cevSensor
entPhysicalVendorType.1007 = cevSensor
entPhysicalVendorType.1008 = cevSensor
entPhysicalVendorType.1009 = cevSensor
entPhysicalVendorType.1010 = cevSensor
entPhysicalVendorType.1011 = cevSensor
entPhysicalVendorType.1012 = cevSensor
entPhysicalVendorType.1013 = cevSensor
entPhysicalVendorType.1014 = cevSensor
entPhysicalVendorType.1015 = cevSensorModuleDeviceTemp
entPhysicalVendorType.1016 = cevSensorModuleDeviceTemp
entPhysicalVendorType.1017 = cevSensorModuleDeviceTemp
entPhysicalVendorType.1018 = cevSensorModuleDeviceTemp
entPhysicalVendorType.1026 = cevModuleCpuType
entPhysicalVendorType.1027 = cevContainerSPABay
entPhysicalVendorType.1028 = cevContainerSPABay
entPhysicalVendorType.1029 = cevContainerSPABay
entPhysicalVendorType.1030 = cevContainerSPABay
....
```

entPhysicalVendorType は、物理エンティティの一意のベンダー固有のハードウェア タイプを識別します。

```
entPhysicalContainedIn.1000 = 2
entPhysicalContainedIn.1001 = 1000
entPhysicalContainedIn.1002 = 1000
entPhysicalContainedIn.1003 = 1000
entPhysicalContainedIn.1004 = 1000
entPhysicalContainedIn.1005 = 1000
entPhysicalContainedIn.1006 = 1000
entPhysicalContainedIn.1007 = 1000
entPhysicalContainedIn.1008 = 1000
entPhysicalContainedIn.1009 = 1000
entPhysicalContainedIn.1010 = 1000
entPhysicalContainedIn.1011 = 1000
entPhysicalContainedIn.1012 = 1000
entPhysicalContainedIn.1013 = 1000
entPhysicalContainedIn.1014 = 1000
entPhysicalContainedIn.1015 = 1000
entPhysicalContainedIn.1016 = 1000
entPhysicalContainedIn.1017 = 1000
entPhysicalContainedIn.1018 = 1000
entPhysicalContainedIn.1026 = 1000
entPhysicalContainedIn.1027 = 1000
entPhysicalContainedIn.1028 = 1000
entPhysicalContainedIn.1029 = 1000
entPhysicalContainedIn.1030 = 1000
```

entPhysicalContainedIn は、コンポーネントの親エンティティの entPhysicalIndex を示します。

```
entPhysicalClass.1000 = module(9)
entPhysicalClass.1001 = sensor(8)
entPhysicalClass.1002 = sensor(8)
entPhysicalClass.1003 = sensor(8)
entPhysicalClass.1004 = sensor(8)
entPhysicalClass.1005 = sensor(8)
entPhysicalClass.1006 = sensor(8)
entPhysicalClass.1007 = sensor(8)
```

```
entPhysicalClass.1008 = sensor(8)
entPhysicalClass.1009 = sensor(8)
entPhysicalClass.1010 = sensor(8)
entPhysicalClass.1011 = sensor(8)
entPhysicalClass.1012 = sensor(8)
entPhysicalClass.1013 = sensor(8)
entPhysicalClass.1014 = sensor(8)
entPhysicalClass.1015 = sensor(8)
entPhysicalClass.1016 = sensor(8)
entPhysicalClass.1017 = sensor(8)
entPhysicalClass.1018 = sensor(8)
entPhysicalClass.1026 = other(1)
entPhysicalClass.1027 = container(5)
entPhysicalClass.1028 = container(5)
entPhysicalClass.1029 = container(5)
entPhysicalClass.1030 = container(5)
```

entPhysicalClass は、ハードウェア デバイスの一般的なタイプを示します。

```
entPhysicalParentRelPos.1000 = 0
entPhysicalParentRelPos.1001 = 0
entPhysicalParentRelPos.1002 = 1
entPhysicalParentRelPos.1003 = 2
entPhysicalParentRelPos.1004 = 3
entPhysicalParentRelPos.1005 = 4
entPhysicalParentRelPos.1006 = 5
entPhysicalParentRelPos.1007 = 6
entPhysicalParentRelPos.1008 = 7
entPhysicalParentRelPos.1009 = 8
entPhysicalParentRelPos.1010 = 9
entPhysicalParentRelPos.1011 = 10
entPhysicalParentRelPos.1012 = 11
entPhysicalParentRelPos.1013 = 12
entPhysicalParentRelPos.1014 = 13
entPhysicalParentRelPos.1015 = 14
entPhysicalParentRelPos.1016 = 15
entPhysicalParentRelPos.1017 = 16
entPhysicalParentRelPos.1018 = 17
entPhysicalParentRelPos.1026 = 0
entPhysicalParentRelPos.1027 = 0
entPhysicalParentRelPos.1028 = 1
entPhysicalParentRelPos.1029 = 2
entPhysicalParentRelPos.1030 = 3
```

entPhysicalParentRelPos は、他のエンティティ間におけるこの子の相対的な位置を示します。

```
entPhysicalName.1000 = module 0
entPhysicalName.1001 = V1: VMA 0/0
entPhysicalName.1002 = V1: VMB 0/1
entPhysicalName.1003 = V1: VMC 0/2
entPhysicalName.1004 = V1: VMD 0/3
entPhysicalName.1005 = V1: VME 0/4
entPhysicalName.1006 = V1: VMF 0/5
entPhysicalName.1007 = V1: 12v 0/6
entPhysicalName.1008 = V1: VDD 0/7
entPhysicalName.1009 = V1: GP1 0/8
entPhysicalName.1010 = V1: GP2 0/9
entPhysicalName.1011 = V2: VMB 0/10
entPhysicalName.1012 = V2: 12v 0/11
entPhysicalName.1013 = V2: VDD 0/12
entPhysicalName.1014 = V2: GP2 0/13
entPhysicalName.1015 = Temp: Left 0/14
```

```
entPhysicalName.1016 = Temp: Center 0/15
entPhysicalName.1017 = Temp: Asic1 0/16
entPhysicalName.1018 = Temp: Right 0/17
entPhysicalName.1026 = cpu 0/0
entPhysicalName.1027 = subslot 0/0
entPhysicalName.1028 = subslot 0/1
entPhysicalName.1029 = subslot 0/2
entPhysicalName.1030 = subslot 0/3
```

entPhysicalName は、物理エンティティのテキスト名を指定します。

```
entPhysicalHardwareRev.1000 = V00
entPhysicalHardwareRev.1001 =
entPhysicalHardwareRev.1002 =
entPhysicalHardwareRev.1003 =
entPhysicalHardwareRev.1004 =
entPhysicalHardwareRev.1005 =
entPhysicalHardwareRev.1006 =
entPhysicalHardwareRev.1007 =
entPhysicalHardwareRev.1008 =
entPhysicalHardwareRev.1009 =
entPhysicalHardwareRev.1010 =
entPhysicalHardwareRev.1011 =
entPhysicalHardwareRev.1012 =
entPhysicalHardwareRev.1013 =
entPhysicalHardwareRev.1014 =
entPhysicalHardwareRev.1015 =
entPhysicalHardwareRev.1016 =
entPhysicalHardwareRev.1017 =
entPhysicalHardwareRev.1018 =
entPhysicalHardwareRev.1026 =
entPhysicalHardwareRev.1027 =
entPhysicalHardwareRev.1028 =
entPhysicalHardwareRev.1029 =
entPhysicalHardwareRev.1030 =
```

entPhysicalHardware は、物理エンティティのベンダー固有のハードウェア リビジョン番号（string）を提供します。

```
entPhysicalSerialNum.1000 = JAB11090506
entPhysicalSerialNum.1001 =
entPhysicalSerialNum.1002 =
entPhysicalSerialNum.1003 =
entPhysicalSerialNum.1004 =
entPhysicalSerialNum.1005 =
entPhysicalSerialNum.1006 =
entPhysicalSerialNum.1007 =
entPhysicalSerialNum.1008 =
entPhysicalSerialNum.1009 =
entPhysicalSerialNum.1010 =
entPhysicalSerialNum.1011 =
entPhysicalSerialNum.1012 =
entPhysicalSerialNum.1013 =
entPhysicalSerialNum.1014 =
entPhysicalSerialNum.1015 =
entPhysicalSerialNum.1016 =
entPhysicalSerialNum.1017 =
entPhysicalSerialNum.1018 =
entPhysicalSerialNum.1026 =
entPhysicalSerialNum.1027 =
entPhysicalSerialNum.1028 =
entPhysicalSerialNum.1029 =
entPhysicalSerialNum.1030 =
```

entPhysicalSerialNumber は、物理エンティティのベンダー固有のシリアル番号（string）を提供します。

```
entPhysicalMfgName.1000 = Cisco Systems Inc
entPhysicalMfgName.1001 =
entPhysicalMfgName.1002 =
entPhysicalMfgName.1003 =
entPhysicalMfgName.1004 =
entPhysicalMfgName.1005 =
entPhysicalMfgName.1006 =
entPhysicalMfgName.1007 =
entPhysicalMfgName.1008 =
entPhysicalMfgName.1009 =
entPhysicalMfgName.1010 =
entPhysicalMfgName.1011 =
entPhysicalMfgName.1012 =
entPhysicalMfgName.1013 =
entPhysicalMfgName.1014 =
entPhysicalMfgName.1015 =
entPhysicalMfgName.1016 =
entPhysicalMfgName.1017 =
entPhysicalMfgName.1018 =
entPhysicalMfgName.1026 =
entPhysicalMfgName.1027 =
entPhysicalMfgName.1028 =
entPhysicalMfgName.1029 =
entPhysicalMfgName.1030 =
```

entPhysicalMfgName は、物理コンポーネントの製造元の名前を提供します。

```
entPhysicalModelName.1000 = ASR1000-SIP10
entPhysicalModelName.1001 =
entPhysicalModelName.1002 =
entPhysicalModelName.1003 =
entPhysicalModelName.1004 =
entPhysicalModelName.1005 =
entPhysicalModelName.1006 =
entPhysicalModelName.1007 =
entPhysicalModelName.1008 =
entPhysicalModelName.1009 =
entPhysicalModelName.1010 =
entPhysicalModelName.1011 =
entPhysicalModelName.1012 =
entPhysicalModelName.1013 =
entPhysicalModelName.1014 =
entPhysicalModelName.1015 =
entPhysicalModelName.1016 =
entPhysicalModelName.1017 =
entPhysicalModelName.1018 =
entPhysicalModelName.1026 =
entPhysicalModelName.1027 =
entPhysicalModelName.1028 =
entPhysicalModelName.1029 =
entPhysicalModelName.1030 =
```

entPhysicalModelName は、物理コンポーネントのベンダー固有のモデル名文字列を提供します。

```
entPhysicalIsFRU.1000 = true(1)
entPhysicalIsFRU.1001 = false(2)
entPhysicalIsFRU.1002 = false(2)
```

```
entPhysicalIsFRU.1003 = false(2)
entPhysicalIsFRU.1004 = false(2)
entPhysicalIsFRU.1005 = false(2)
entPhysicalIsFRU.1006 = false(2)
entPhysicalIsFRU.1007 = false(2)
entPhysicalIsFRU.1008 = false(2)
entPhysicalIsFRU.1009 = false(2)
entPhysicalIsFRU.1010 = false(2)
entPhysicalIsFRU.1011 = false(2)
entPhysicalIsFRU.1012 = false(2)
entPhysicalIsFRU.1013 = false(2)
entPhysicalIsFRU.1014 = false(2)
entPhysicalIsFRU.1015 = false(2)
entPhysicalIsFRU.1016 = false(2)
entPhysicalIsFRU.1017 = false(2)
entPhysicalIsFRU.1018 = false(2)
entPhysicalIsFRU.1026 = false(2)
entPhysicalIsFRU.1027 = false(2)
entPhysicalIsFRU.1028 = false(2)
entPhysicalIsFRU.1029 = false(2)
entPhysicalIsFRU.1030 = false(2)
```

entPhysicalIsFRU は、物理エンティティが現場交換可能ユニット（FRU）と見なされるかどうかを示します。

サンプル設定に関して、次の点に注意してください。

- すべてのシャーシ スロットにおよびラインカードのポートは entPhysicalContainedIn 値が同じです。
  - シャーシ スロットの場合、entPhysicalContainedIn = 1（シャーシの entPhysicalIndex）。
  - SPA ポートの場合、entPhysicalContainedIn = 1280（SPA カードの entPhysicalIndex）。
- 各シャーシ スロットにおよびラインカードのポートは、親オブジェクト内の相対位置を示す entPhysicalParentRelPos が異なります。

物理ポートの ifIndex 値の確認

ENTITY-MIB **entAliasMappingIdentifier** は、ポートの entPhysicalIndex を IF-MIB ifTable の対応する ifIndex 値にマッピングして、物理ポートをインターフェイスにマッピングします。次のサンプルは、entPhysicalIndex が 35 である物理ポートが ifIndex 値が 4 であるインターフェイスに関連付けられていることを示します。（考えられる MIB 値の詳細については、MIB を参照してください）。

```
entAliasMappingIdentifer.1813.0 = ifIndex.4
```

FRU のステータスのモニタリングと設定

電源やラインカードなどの FRU の管理ステータスおよび動作ステータスを確認するには、CISCO-ENTITY-FRU-CONTROL-MIB cefcModuleTable のオブジェクトを表示します。

- cefcModuleAdminStatus：FRU の管理状態。cefcModuleAdminStatus を使用して、FRU をイネーブルまたはディセーブルにします。
- cefcModuleOperStatus：FRU の現在の動作状態。

図 A-1 に、entPhysicalIndex が 1000 である SIP カードの cefcModuleTable エントリを示します。

図 A-1　サンプル cefcModuleTable エントリ

```
cefcModuleAdminStatus.1000 = enabled(1)
cefcModuleOperStatus.1000 = ok(2)
cefcModuleResetReason.1000 = unknown(1)
cefcModuleStatusLastChangeTime.1000 =
15865
```

FRU 状態の変更を示す通知をルータがどのように生成するかについては、「FRU ステータスの変更」（P.A-25）を参照してください。

ENTITY-ALARM-MIB を使用したエンティティ アラームのモニタ

ENTITY-MIB

エンティティの物理的なテーブルには、ルータの物理エンティティを管理するための情報が含まれます。また、階層と相互の関係を示す包含ツリーにエンティティを編成します。エンティティ階層については、付録 A「エンティティの包含ツリー」の項を参照してください。次のサンプル出力は、電源ベイ 0 の ASR1002 AC 電源に関する情報が含まれています。

```
ptolemy-265->getmany -v2c 9.0.0.56 public entityMIB | grep "\.4 "
entPhysicalDescr.4 = Cisco ASR1002 AC Power Supply
entPhysicalVendorType.4 = cevPowerSupplyASR1002AC
entPhysicalContainedIn.4 = 3
entPhysicalClass.4 = powerSupply(6)
entPhysicalParentRelPos.4 = 0
entPhysicalName.4 = Power Supply Module 0
entPhysicalHardwareRev.4 = V01
entPhysicalFirmwareRev.4 =
entPhysicalSoftwareRev.4 =
entPhysicalSerialNum.4 = ART1132U00C
entPhysicalMfgName.4 =
entPhysicalModelName.4 = ASR1002-PWR-AC
entPhysicalAlias.4 =
entPhysicalAssetID.4 =
entPhysicalIsFRU.4 = true(1)
entPhysicalMfgDate.4 = 00 00  00 00   00 00  00 00
entPhysicalUris.4 = URN:CLEI:COUPACJBAA
entPhysicalChildIndex.3.4 = 4
```

この MIB の詳細については、「ENTITY-MIB（RFC 4133）」（P.3-99）を参照してください。

CISCO-ENTITY-ALARM-MIB

CISCO-ENTITY-ALARM-MIB は、シャーシ、スロット、モジュール、ポート、電源などのシステムに含まれる物理エンティティによって生成されたアラームのモニタリングをサポートします。物理エンティティによって生成されたアラームをモニタするには、エンティティが entPhysicalTable の行で表されている必要があります。

この MIB の詳細については、「CISCO-ENTITY-ALARM-MIB」（P.3-32）を参照してください。

アラームの説明のマッピング テーブル

（entPhysicalVendorType OID によって表される）エンティティのタイプごとに、このテーブルには、一意の ceAlarmDescrIndex と entPhysicalvendorType OID 間のマッピングが含まれます。

ceAlarmDescrMapEntry は、CeAlarmDescrMapEntry によってインデックスが作成されます。

（注）　ceAlarmDescrIndex と entPhysicalvendorType OID 間のマッピングは、エンティティのタイプがアラームのモニタリングをサポートする場合にのみ存在し、デバイス起動時からデバイスにあります。

次に、サンプル出力例を示します。

```
ptolemy-218->getmany -v2c 9.0.0.56 public ceAlarmDescrMapTable
ceAlarmDescrVendorType.1 = cevPortCT3
ceAlarmDescrVendorType.2 = cevPortT1E1
ceAlarmDescrVendorType.3 = cevPortT3E3
ceAlarmDescrVendorType.4 = cevContainerSFP
ceAlarmDescrVendorType.5 = cevContainerASR1000RPSlot
ceAlarmDescrVendorType.6 = cevContainerASR1000FPSlot
ceAlarmDescrVendorType.7 = cevContainerASR1000CCSlot
ceAlarmDescrVendorType.8 = cevContainerASR1000PowerSupplyBay
ceAlarmDescrVendorType.9 = cevSensorModuleDeviceTemp
ceAlarmDescrVendorType.10 = cevSensorModuleDeviceVoltage
ceAlarmDescrVendorType.11 = cevSensorModuleDeviceCurrent
ceAlarmDescrVendorType.12 = cevSensor
ceAlarmDescrVendorType.13 = cevModuleASR1002RP1
ceAlarmDescrVendorType.14 = cevPortUSB
ceAlarmDescrVendorType.15 = cevPortGe
ceAlarmDescrVendorType.16 = cevModuleASR1000ESP10
ceAlarmDescrVendorType.17 = cevModuleASR1002SIP10
ceAlarmDescrVendorType.18 = cevContainerSPABay
ceAlarmDescrVendorType.19 = cevPowerSupplyASR1002AC
ceAlarmDescrVendorType.20 = cevModuleASR1002Spa4pGe
```

ASR1000 モジュール（RP、FP、CC、および PEM）の温度センサーには、entPhysicalvendorType OID として cevSensorModuleDeviceTemp が含まれます。上記のサンプル出力で、インデックス（ceAlarmDescrIndex）9 は、cevSensorModuleDeviceTemp にマッピングされ、インデックス 19 は、エンティティの物理ベンダー タイプ OID が cevPowerSupplyASR1002AC である AER10002 電源にマッピングされます。

（注）　SPA はすべての ASR1000 モジュールに含まれません。その独自のベンダー タイプ OID がセンサーに定義されています。

（注）　ASR1000 snmp エージェントがセンサー タイプを特定できない場合は、汎用ベンダー OID の cevSenor が使用されます。

アラーム説明テーブル

アラーム説明テーブルには、システムで採用されている各ベンダー タイプによって定義された各アラーム タイプの説明が含まれます。各アラーム説明エントリ（ceAlarmDescrEntry）は、ceAlarmDescrIndex および ceAlarmDescrAlarmType によってインデックスが作成されます。

次に、ASR1000 モジュールのエンティティの温度タイプすべてに定義されているすべてのアラーム タイプのサンプル出力を示します。 インデックス 9 は、前の項の ceAlarmDescrMapTable から取得されます。

```
ptolemy-225->getmany -v2c 9.0.0.56 public ceAlarmDescrTable | grep "\.9\."
ceAlarmDescrSeverity.9.0 = 1
ceAlarmDescrSeverity.9.1 = 1
ceAlarmDescrSeverity.9.2 = 1
ceAlarmDescrSeverity.9.3 = 2
ceAlarmDescrSeverity.9.4 = 3
ceAlarmDescrSeverity.9.5 = 1
ceAlarmDescrSeverity.9.6 = 1
ceAlarmDescrSeverity.9.7 = 2
ceAlarmDescrSeverity.9.8 = 3
ceAlarmDescrText.9.0 = Faulty Temperature Sensor
ceAlarmDescrText.9.1 = Temp Above Normal (Shutdown)
ceAlarmDescrText.9.2 = Temp Above Normal
ceAlarmDescrText.9.3 = Temp Above Normal
ceAlarmDescrText.9.4 = Temp Above Normal
ceAlarmDescrText.9.5 = Temp Below Normal (Shutdown)
ceAlarmDescrText.9.6 = Temp Below Normal
ceAlarmDescrText.9.7 = Temp Below Normal
ceAlarmDescrText.9.8 = Temp Below Normal
```

『Bellcore Technical Reference TR-NWT-000474 Issue 4, December 1993, OTGR Section 4. Network Maintenance: Alarm and Control - Network Element』を参照してください。重大度は次のように定義されます。

- critical(1)
- major(2)
- minor(3)
- info(4)

次に、センサーに定義されているアラームのリストを示します。

```
Alarm type 0 is for faulty sensor
Alarm type 1 is for crossing  the shutdow threshold  (above normal range).
Alarm type 2 is for crossing  the critical threshold  (above normal range).
Alarm type 3 is for crossing  the major  threshold (above normal range).
Alarm type 4 is for crossing  the minor  threshold (above normal range).
Alarm type 5 is for crossing  the shutdow threshold (below normal range).
Alarm type 6 is for crossing  the critical threshold (below normal range).
Alarm type 7 is for crossing  the major  threshold (below normal range).
Alarm type 8 is for crossing  the minor  threshold (below normal range).
```

これらのアラーム タイプは、すべてのセンサー物理エンティティ タイプに対して定義されます。唯一の違いは、センサー物理タイプごとに ceAlarmDescrText が異なる点です。アラームの説明テキストで、温度センサーには「TEMP」、電圧センサーには「Volt」があります。

次に、すべてのアラーム タイプのサンプル出力を示します。cevPowerSupplyASR1002AC がベンダー タイプ OID であり、ceAlarmDescrIndex 19 にマッピングされた ASR1002 AC 電源に対して定義されます。

```
ptolemy-237->getmany -v2c 9.0.0.56 public ceAlarmDescrTable | grep "\.19\."
ceAlarmDescrSeverity.19.0 = 1
ceAlarmDescrSeverity.19.1 = 1
ceAlarmDescrSeverity.19.2 = 1
ceAlarmDescrSeverity.19.3 = 2
ceAlarmDescrSeverity.19.4 = 2
ceAlarmDescrSeverity.19.5 = 2
```

```
ceAlarmDescrText.19.0 = Power Supply Failure
ceAlarmDescrText.19.1 = All Fans Failed
ceAlarmDescrText.19.2 = Multiple Fan Failures
ceAlarmDescrText.19.3 = Fan 0 Failure
ceAlarmDescrText.19.4 = Fan 1 Failure
ceAlarmDescrText.19.5 = Fan 2 Failure
```

アラーム テーブル

アラーム テーブルは、システムに含まれている各物理エンティティに関するアラームの制御およびステータス情報を示します。テーブルには、アラームを生成できる各物理エンティティによって現在アサートされているアラームが含まれます。アラームを生成できるエンティティ物理テーブル内の物理エンティティごとに、このテーブルにエントリがあります。アラーム エントリ（ceAlarmEntry）は、エンティティ物理インデックス（entPhysicalIndex）によってインデックスが作成されます。次に、アラーム エントリの MIB オブジェクトのリストを示します。

- **ceAlarmFilterProfile**
  アラーム フィルタ プロファイル オブジェクトには、対応する物理エンティティに関連付けられたアラーム フィルタ プロファイルを一意に識別する整数値が含まれます。アラーム フィルタ プロファイルは、エージェントが対応する物理エンティティについてモニタおよびシグナリングするアラーム タイプを制御します。このオブジェクトのデフォルト値は 0 で、エージェントは、対応する物理エンティティに関連付けられたすべてのアラームをモニタおよびシグナリングします。
- **ceAlarmSeverity**
  このオブジェクトは、対応する物理エンティティによって現在アサートされている最大の重大度のアラームを示します。
  値が「0」の場合、対応する物理エンティティは現在アラームをアサートしていません。
- **ceAlarmList**
  このオブジェクトは、対応する物理エンティティによって現在アサートされているアラームを示します。 アラームが物理エンティティによってアサートされている場合、アラーム リスト内の対応するビットは 1 に設定されます。アラーム リストは、オクテット文字列として定義され、そのサイズの範囲は 0 ～ 32 です。
  - 物理エンティティが現在アラームをアサートしていない場合、リストの長さはゼロです。それ以外の場合、長さは 32 です。
  - オクテット文字列はアラーム リストを表し、各ビットはアラーム タイプを表します。

オクテット 1：

```
    7 6 5 4 3 2 1 0
   +-+-+-+-+-+-+-+-+
   |               |
   +-+-+-+-+-+-+-+-+
    | | | | | | | |
    | | | | | | | +- Alarm type 0
    | | | | | | +--- Alarm type 1
    | | | | | +----- Alarm type 2
    | | | | +------- Alarm type 3
    | | | +--------- Alarm type 4
    | | +----------- Alarm type 5
    | +------------- Alarm type 6
+---------------- Alarm type 7
```

オクテット 2：

```
    7 6 5 4 3 2 1 0
   +-+-+-+-+-+-+-+-+
   |               |
   +-+-+-+-+-+-+-+-+
```

```
    | | | | | | | |
    | | | | | | | +- Alarm type 8
    | | | | | | +--- Alarm type 9
    | | | | | +----- Alarm type 10
    | | | | +------- Alarm type 11
    | | | +--------- Alarm type 12
    | | +----------- Alarm type 13
    | +------------- Alarm type 14
+---------------- Alarm type 15
```

オクテット xx

オクテット 32：

```
    7 6 5 4 3 2 1 0
   +-+-+-+-+-+-+-+-+
   |               |
   +-+-+-+-+-+-+-+-+
    | | | | | | | |
    | | | | | | | +- Alarm type 248
    | | | | | | +--- Alarm type 249
    | | | | | +----- Alarm type 250
    | | | | +------- Alarm type 251
    | | | +--------- Alarm type 252
    | | +----------- Alarm type 253
    | +------------- Alarm type 254
+---------------- Alarm type 255
```

エンティティ物理テーブル（ENTITY-MIB の entPhysicalTable）から、電源ベイ 0 の ASR1002 AC 電源の entPhysicalIndex が 4 であることがわかります。

次に、PS ベイ 0 内の電源のアラーム リストのサンプル出力を示します。

```
ptolemy-248->getone -v2c 9.0.0.56 public ceAlarmList.4
ceAlarmList.4 =
09 00  00 00   00 00  00 00    00 00  00 00   00 00  00 00
00 00  00 00   00 00  00 00    00 00  00 00   00 00  00 00
```

オクテット 1：09

```
    7 6 5 4 3 2 1 0
   +-+-+-+-+-+-+-+-+
    0 0 0 0 1 0 0 1
   +-+-+-+-+-+-+-+-+
    | | | | | | | |
    | | | | | | | +- Alarm type 0
    | | | | | | +--- Alarm type 1
    | | | | | +----- Alarm type 2
    | | | | +------- Alarm type 3
    | | | +--------- Alarm type 4
    | | +----------- Alarm type 5
    | +------------- Alarm type 6
+---------------- Alarm type 7
```

「アラーム説明テーブル」の項のサンプル出力およびアラーム マッピング テーブルより、ベイ 0 の ASR1002 AC 電源で次のアラームがアサートされています。

タイプ タイプ 0：電源の障害

タイプ タイプ 3：ファン 0 の障害

entPhysiclIndex が 14 であるベイ 1 の ASR1002 AC 電源：

```
ptolemy-247->getone -v2c 9.0.0.56 public ceAlarmList.14
```

ceAlarmList.14 =

ベイ 1 の電源について戻されたアラーム リストの長さが 0 であるため、ベイ 1 の電源にはアサートされるアラームがありません。

次に、「show facility-alarm status」CLI コマンドの出力を示します。デバイスで現在アサートされているすべてのアラームが表示されます。

```
R5-mcp-2ru-1#sh facility-alarm status
System Totals  Critical: 2  Major: 1  Minor: 0
Source              Severity      Description [Index]
------              --------      -------------------
Cisco ASR1002 AC Power Sup CRITICAL      Power Supply Failure [0]
Cisco ASR1002 AC Power Sup MAJOR         Fan 0 Failure [3]
xcvr container 0/0/1       INFO          Transceiver Missing [0]
xcvr container 0/0/2       CRITICAL      Transceiver Missing - Link Down [1]
xcvr container 0/0/3       INFO          Transceiver Missing [0]
```

アラーム履歴テーブル

アラーム履歴テーブル ceAlarmHistTable には、エージェントによって生成されたアラームのアサートおよびクリアの履歴が含まれます。ceAlarmHistTableSize は、アラーム履歴テーブルのサイズを制御するために使用されます。値が 0 の場合、履歴はこのテーブルに保持されません。ceAlarmHistTable の容量がこのオブジェクトで指定された値に達すると、エージェントは新しいエントリを格納するために最も古いエンティティを削除します。

ceAlarmHistLastIndex オブジェクトには、デバイスの snmp のエージェントによってテーブルに追加された最後のエントリに対応する最後のインデックスが含まれます。管理クライアントは、CISCO-ENTITY-ALARM-MIB モジュールで定義された付録 A「アラーム通知」に示されている通知を使用する場合、このオブジェクトをポーリングして、エージェントが送信した通知を見失ったかどうかを確認できます。

次に、ceAlarmHistIndex でインデックスが作成された ceAlarmHistEntry に定義されている MIB オブジェクトのリストを示します。

- **ceAlarmHistIndex**
  これは、テーブルのエントリを一意に識別する整数値です。このオブジェクトの値は「1」から始まり、アラームがアラーム履歴テーブルに対して追加（アサートまたはクリア）されるごとに単調に増加します。このオブジェクトの値が「4294967295」の場合、次のアラーム状態遷移をモニタするときに「1」にリセットされます。
- **ceAlarmHistType**
  このオブジェクトは、アラームがアサートまたはクリアされた結果としてエントリが追加されたことを示します。
- **ceAlarmHistEntPhysicalIndex**
  このオブジェクトには、アラームを生成した物理エンティティの entPhysicalIndex が含まれます。
- **ceAlarmHistAlarmType**
  このオブジェクトは、生成されたアラームのタイプを示します。
- **ceAlarmHistSeverity**
  このオブジェクトは、生成されたアラームの重大度を示します。
- **ceAlarmHistTimeStamp**
  このオブジェクトは、アラームが生成されると、sysUpTime オブジェクトの値を示します。

例 A-4　アラーム履歴のサンプル出力の表示

```
ptolemy-257->getnext -v2c 9.0.0.56 public ceAlarmHistory
ceAlarmHistTableSize.0 = 200  →  the size of alarm history table
ptolemy-258->getnext -v2c 9.0.0.56 public ceAlarmHistTableSize.0
ceAlarmHistLastIndex.0 = 21  →  the index for the last alarm added
```

例 A-5　アラーム履歴テーブルに対して（アサートまたはクリア）追加された最後のアラーム アクションの表示

```
ptolemy-259->getmany -v2c 9.0.0.56 public ceAlarmHistTable | grep "\.21 "
ceAlarmHistType.21 = cleared(2) → alarm cleared
ceAlarmHistEntPhysicalIndex.21=4 → it is for physical entity indexed by 4
ceAlarmHistAlarmType.21 = 3 → alarm type is 3
ceAlarmHistSeverity.21 = major(2) → the alarm severity is major(2)
ceAlarmHistTimeStamp.21 = 7506193
```

この時点で、EMS アプリケーションには、物理エンティティおよび物理エンティティに対して定義されたエンティティ アラーム タイプに関するすべての情報をすでに持っている必要があります。

例 A-6　entPhysicalIndex の値が 4 である物理エンティティの表示

```
entPhysicalDescr.4 = Cisco ASR1002 AC Power Supply
entPhysicalVendorType.4 = cevPowerSupplyASR1002AC
entPhysicalContainedIn.4 = 3
entPhysicalClass.4 = powerSupply(6)
entPhysicalParentRelPos.4 = 0
entPhysicalName.4 = Power Supply Module 0
entPhysicalHardwareRev.4 = V01
entPhysicalFirmwareRev.4 =
entPhysicalSoftwareRev.4 =
entPhysicalSerialNum.4 = ART1132U00C
entPhysicalMfgName.4 =
entPhysicalModelName.4 = ASR1002-PWR-AC
```

例 A-7　cevPowerSupplyASR1002AC に定義されているアラーム タイプの表示

```
ceAlarmDescrSeverity.19.0 = 1
ceAlarmDescrSeverity.19.1 = 1
ceAlarmDescrSeverity.19.2 = 1
ceAlarmDescrSeverity.19.3 = 2
ceAlarmDescrSeverity.19.4 = 2
ceAlarmDescrSeverity.19.5 = 2
ceAlarmDescrText.19.0 = Power Supply Failure
ceAlarmDescrText.19.1 = All Fans Failed
ceAlarmDescrText.19.2 = Multiple Fan Failures
ceAlarmDescrText.19.3 = Fan 0 Failure
ceAlarmDescrText.19.4 = Fan 1 Failure
ceAlarmDescrText.19.5 = Fan 2 Failure
```

cevPowerSupplyASR1002AC に定義されているアラーム タイプから、アプリケーションはアラーム履歴テーブルの最後のエントリを次のように容易に解釈できます。ファン 0 の障害アラームが電源ベイ 0 の Cisco ASR1002 AC 電源でクリアされました

アラーム通知

CISCO-ENTITY-ALARM-MIB は、アラーム アサート通知（ceAlarmAsserted）およびアラーム クリア通知（ceAlarmCleared）をサポートします。通知は、snmp SET による ceAlarmNotifiesEnable オブジェクトの設定でイネーブルにできます。ceAlarmNotifiesEnable には、アラーム通知の重大度または値 0 が含まれます。

```
severity 1: critical       Service affecting Condition
severity 2: major          Immediate action needed
severity 3: minor          Minor warning conditions
severity 4: informational  Informational messages
```

重大度 4 では、すべての重大度の通知がイネーブルになります。

重大度 3 では、重大度 1、2、および 3 の通知がイネーブルになります。

重大度 2 では、重大度 1 および 2 の通知がイネーブルになります。

重大度 1 では、重大度 1 の通知のみがイネーブルになります。

値 0 では、アラーム通知がディセーブルになります。

アラーム通知は CLI コマンドでイネーブルまたはディセーブルにできます。アラーム通知をディセーブルにするには、「NO」形式を使用します。

```
snmp-server enable traps alarm [critical, major, minor, information]
no snmp-server enable traps alarm [critical, major, minor, information]
```

アラーム通知には、アラーム履歴エントリで説明したものとまったく同じ情報が含まれます。MIB オブジェクトおよび受信したアラーム通知の解釈については、「アラーム履歴テーブル」の項を参照してください。

例 A-8　受信したサンプル通知の表示

```
Received SNMPv2c Trap:
Community: public
From: 9.0.0.56
sysUpTimeInstance = 7500792
snmpTrapOID.0 = ceAlarmCleared
ceAlarmHistEntPhysicalIndex.19 = 4
ceAlarmHistAlarmType.19 = 0
ceAlarmHistSeverity.19 = critical(1)
ceAlarmHistTimeStamp.19 = 7500792


Received SNMPv2c Trap:
Community: public
From: 9.0.0.56
sysUpTimeInstance = 7504592
snmpTrapOID.0 = ceAlarmAsserted
ceAlarmHistEntPhysicalIndex.20 = 4
ceAlarmHistAlarmType.20 = 3
ceAlarmHistSeverity.20 = major(2)
ceAlarmHistTimeStamp.20 = 7504592


Received SNMPv2c Trap:
Community: public
From: 9.0.0.56
sysUpTimeInstance = 7506193
snmpTrapOID.0 = ceAlarmCleared
ceAlarmHistEntPhysicalIndex.21 = 4
ceAlarmHistAlarmType.21 = 3
ceAlarmHistSeverity.21 = major(2)
ceAlarmHistTimeStamp.21 = 7506193
```

エンティティの包含ツリー

次に、ASR1002 デバイスのサンプル エンティティ階層、Mib Variables printed : <entPhysicalName entPhysicalClass> を示します。

```
ENTITY-MIB containment tree:
    |
    \-1 (cevChassisASR1002) : Chassis : chassis
```

```
|
+-2 (cevContainerASR1000FPSlot) : slot F0 : container
|    |
|    \-9000 (cevModuleASR1000ESP10) : module F0 : module
|         |
|         +-9001 (cevSensorModuleDeviceVoltage) : V1: VMA F0/0 : sensor
|         |
|         +-9002 (cevSensorModuleDeviceVoltage) : V1: VMB F0/1 : sensor
|         |
|         +-9003 (cevSensorModuleDeviceVoltage) : V1: VMC F0/2 : sensor
|         |
|         +-9004 (cevSensorModuleDeviceVoltage) : V1: VMD F0/3 : sensor
|         |
|         +-9005 (cevSensorModuleDeviceVoltage) : V1: VME F0/4 : sensor
|         |
|         +-9006 (cevSensorModuleDeviceVoltage) : V1: 12v F0/5 : sensor
|         |
|         +-9007 (cevSensorModuleDeviceVoltage) : V1: VDD F0/6 : sensor
|         |
|         +-9008 (cevSensorModuleDeviceVoltage) : V1: GP1 F0/7 : sensor
|         |
|         +-9009 (cevSensorModuleDeviceVoltage) : V2: VMA F0/8 : sensor
|         |
|         +-9010 (cevSensorModuleDeviceVoltage) : V2: VMB F0/9 : sensor
|         |
|         +-9011 (cevSensorModuleDeviceVoltage) : V2: VMC F0/10 : sensor
|         |
|         +-9012 (cevSensorModuleDeviceVoltage) : V2: VMD F0/11 : sensor
|         |
|         +-9013 (cevSensorModuleDeviceVoltage) : V2: VME F0/12 : sensor
|         |
|         +-9014 (cevSensorModuleDeviceVoltage) : V2: VMF F0/13 : sensor
|         |
|         +-9015 (cevSensorModuleDeviceVoltage) : V2: 12v F0/14 : sensor
|         |
|         +-9016 (cevSensorModuleDeviceVoltage) : V2: VDD F0/15 : sensor
|         |
|         +-9017 (cevSensorModuleDeviceVoltage) : V2: GP1 F0/16 : sensor
|         |
|         +-9018 (cevSensorModuleDeviceTemp) : Temp: Inlet F0/17 : sensor
|         |
|         +-9019 (cevSensorModuleDeviceTemp) : Temp: Asic1 F0/18 : sensor
|         |
|         +-9020 (cevSensorModuleDeviceTemp) : Temp: Exhaust1 F0/19 : sensor
|         |
|         +-9021 (cevSensorModuleDeviceTemp) : Temp: Exhaust2 F0/20 : sensor
|         |
|         \-9022 (cevSensorModuleDeviceTemp) : Temp: Asic2 F0/21 : sensor
|
+-3 (cevContainerASR1000PowerSupplyBay) : Power Supply Bay 0 : container
|    |
|    \-4 (cevPowerSupplyASR1002AC) : Power Supply Module 0 : powerSupply
|         |
|         +-5 (cevSensorModuleDeviceCurrent) : PEM Iout P0/0 : sensor
|         |
|         +-6 (cevSensorModuleDeviceVoltage) : PEM Vout P0/1 : sensor
|         |
|         +-7 (cevSensorModuleDeviceVoltage) : PEM Vin P0/2 : sensor
|         |
|         +-8 (cevSensorModuleDeviceTemp) : Temp: PEM P0/3 : sensor
|         |
|         \-9 (cevSensorModuleDeviceTemp) : Temp: FC  P0/4 : sensor
|
+-13 (cevContainerASR1000PowerSupplyBay) : Power Supply Bay 1 : container
```

```
|    |
|    \-14 (cevPowerSupplyASR1002AC) : Power Supply Module 1 : powerSupply
|         |
|         +-15 (cevSensorModuleDeviceCurrent) : PEM Iout P1/0 : sensor
|         |
|         +-16 (cevSensorModuleDeviceVoltage) : PEM Vout P1/1 : sensor
|         |
|         +-17 (cevSensorModuleDeviceVoltage) : PEM Vin P1/2 : sensor
|         |
|         +-18 (cevSensorModuleDeviceTemp) : Temp: PEM P1/3 : sensor
|         |
|         \-19 (cevSensorModuleDeviceTemp) : Temp: FC  P1/4 : sensor
|
+-1000 (cevModuleASR1002SIP10) : module 0 : module
|    |
|    +-1001 (cevSensorModuleDeviceVoltage) : V1: VMA 0/0 : sensor
|    |
|    +-1002 (cevSensorModuleDeviceVoltage) : V1: VMB 0/1 : sensor
|    |
|    +-1003 (cevSensorModuleDeviceVoltage) : V1: VMC 0/2 : sensor
|    |
|    +-1004 (cevSensorModuleDeviceVoltage) : V1: VMD 0/3 : sensor
|    |
|    +-1005 (cevSensorModuleDeviceVoltage) : V1: VME 0/4 : sensor
|    |
|    +-1006 (cevSensorModuleDeviceVoltage) : V1: VMF 0/5 : sensor
|    |
|    +-1007 (cevSensorModuleDeviceVoltage) : V1: 12v 0/6 : sensor
|    |
|    +-1008 (cevSensorModuleDeviceVoltage) : V1: VDD 0/7 : sensor
|    |
|    +-1009 (cevSensorModuleDeviceVoltage) : V1: GP1 0/8 : sensor
|    |
|    +-1010 (cevSensorModuleDeviceVoltage) : V1: GP2 0/9 : sensor
|    |
|    +-1011 (cevSensorModuleDeviceVoltage) : V2: VMB 0/10 : sensor
|    |
|    +-1012 (cevSensorModuleDeviceVoltage) : V2: 12v 0/11 : sensor
|    |
|    +-1013 (cevSensorModuleDeviceVoltage) : V2: VDD 0/12 : sensor
|    |
|    +-1014 (cevSensorModuleDeviceVoltage) : V2: GP2 0/13 : sensor
|    |
|    +-1015 (cevSensorModuleDeviceTemp) : Temp: Left 0/14 : sensor
|    |
|    +-1016 (cevSensorModuleDeviceTemp) : Temp: Center 0/15 : sensor
|    |
|    +-1017 (cevSensorModuleDeviceTemp) : Temp: Asic1 0/16 : sensor
|    |
|    +-1018 (cevSensorModuleDeviceTemp) : Temp: Right 0/17 : sensor
|    |
|    +-1026 (cevModuleCpuType) : cpu 0/0 : other
|    |
|    +-1027 (cevContainerSPABay) : subslot 0/1 : container
|    |
|    +-1028 (cevContainerSPABay) : subslot 0/2 : container
|    |
|    +-1029 (cevContainerSPABay) : subslot 0/3 : container
|    |
|    \-1040 (cevModuleASR1002Spa4pGe) : SPA subslot 0/0 : module
|         |
|         +-1066 (cevSensorModuleDeviceTemp) : subslot 0/0 temperature Sensor 0
|         |
|         +-1067 (cevSensorModuleDeviceTemp) : subslot 0/0 temperature Sensor 1
```

```
|         |
|         +-1091 (cevContainerSFP) : subslot 0/0 transceiver container 0 : cont+
|         |    |
|         |    \-1092 (cevSFP1000BaseT) : subslot 0/0 transceiver 0 : module
|         |         |
|         |         \-1093 (cevPortGe) : GigabitEthernet0/0/0 : port
|         |
|         +-1103 (cevContainerSFP) : subslot 0/0 transceiver container 1 : cont+
|         |
|         +-1115 (cevContainerSFP) : subslot 0/0 transceiver container 2 : cont+
|         |
|         \-1127 (cevContainerSFP) : subslot 0/0 transceiver container 3 : cont+
|
\-7000 (cevModuleASR1002RP1) : module R0 : module
     |
     +-7001 (cevSensorModuleDeviceVoltage) : V1: VMA R0/0 : sensor
     |
     +-7002 (cevSensorModuleDeviceVoltage) : V1: VMB R0/1 : sensor
     |
     +-7003 (cevSensorModuleDeviceVoltage) : V1: VMC R0/2 : sensor
     |
     +-7004 (cevSensorModuleDeviceVoltage) : V1: VMD R0/3 : sensor
     |
     +-7005 (cevSensorModuleDeviceVoltage) : V1: VME R0/4 : sensor
     |
     +-7006 (cevSensorModuleDeviceVoltage) : V1: VMF R0/5 : sensor
     |
     +-7007 (cevSensorModuleDeviceVoltage) : V1: 12v R0/6 : sensor
     |
     +-7008 (cevSensorModuleDeviceVoltage) : V1: VDD R0/7 : sensor
     |
     +-7009 (cevSensorModuleDeviceVoltage) : V1: GP1 R0/8 : sensor
     |
     +-7010 (cevSensorModuleDeviceVoltage) : V1: GP2 R0/9 : sensor
     |
     +-7011 (cevSensorModuleDeviceTemp) : Temp: CPU R0/10 : sensor
     |
     +-7012 (cevSensorModuleDeviceTemp) : Temp: Outlet R0/11 : sensor
     |
     +-7013 (cevSensorModuleDeviceTemp) : Temp: Inlet R0/12 : sensor
     |
     +-7014 (cevSensorModuleDeviceTemp) : Temp: Asic1 R0/13 : sensor
     |
     +-7026 (cevModuleCpuType) : cpu R0/0 : other
     |
     +-7027 (cevPortUSB) : usb R0/0 : port
     |
     \-7029 (cevPortGe) : NME R0 : port

Mib Variables printed : <entPhysicalName entPhysicalClass>
```

SNMP 通知の生成

ここでは、ルータのイベントや条件に応じて生成される SNMP 通知に関する情報を提供し、通知を受信するホストを指定する方法について説明します。

- 通知を受信するホストの指定
- 設定の変更
- FRU ステータスの変更

通知を受信するホストの指定

CLI または SNMP を使用して、SNMP 通知を受信するホストを指定したり、受信する通知のタイプを指定したりできます。CLI の手順については、「通知のイネーブル化」（P.4-2）を参照してください。SNMP を使用してこの情報を設定するには、次の MIB オブジェクトを使用します。

ターゲット ホストを選択し、そのホストのために生成する通知のタイプを指定するには、次のような SNMP-NOTIFICATION-MIB オブジェクトを使用します。

- snmpNotifyTable：ホストと通知タイプを選択するオブジェクトが含まれます。
  - snmpNotifyTag は、SNMP 通知を受信するホストを指定するために使用される任意のオクテット文字列（タグ値）です。ターゲット ホストに関する情報は snmpTargetAddrTable（SNMP-TARGET-MIB）で定義され、各ホストに 1 つ以上のタグ値が関連付けられます。snmpTargetAddrTable のホストにこの snmpNotifyTag 値と一致するタグ値がある場合、ホストは snmpNotifyType で指定された通知タイプを受信するように選択されます。
  - snmpNotifyType は、送信する SNMP 通知のタイプ（notification(1) または inform(2)）です。
- snmpNotifyFilterProfileTable および snmpNotifyFilterTable：通知フィルタを作成してターゲット ホストに送信される通知のタイプを制限するには、これらのテーブルのオブジェクトを使用します。

通知を受信するホストに関する情報を設定するには、SNMP-TARGET-MIB オブジェクトを使用します。

- snmpTargetAddrTable：SNMP 通知を受信するホストの転送アドレス。各エントリは、タグ値のリストを含むホスト アドレスに関する情報を提供します。
  - snmpTargetAddrTagList：ホスト アドレスに関連付けられた一連のタグ値。ホストのタグ値が snmpNotifyTag と一致する場合、ホストは snmpNotifyType で定義された通知タイプを受信するように選択されます。
- snmpTargetParamsTable：SNMP 通知を生成するときに使用する SNMP パラメータ。

特定の SNMP 通知をイネーブルおよびディセーブルにするには、適切な MIB の通知イネーブル オブジェクトを使用します。たとえば、mplsLdpSessionUp または mplsLdpSessionDown 通知を生成するには、MPLS-LDP-MIB オブジェクト mplsLdpSessionUpDownTrapEnable を enabled(1) に設定する必要があります。

設定の変更

エンティティ通知がイネーブルの場合、ルータは、次のテーブル内のいずれかの情報が変更されたときに（ルータ コンフィギュレーションの変更を示す）entConfigChange 通知（ENTITY-MIB）を生成します。

- entPhysicalTable
- entAliasMappingTable
- entPhysicalContainsTable

（注） 設定変更を追跡する管理アプリケーションは、スロットリングまたは送信ロスの結果として失われた entConfigChange 通知を検出するために、entLastChangeTime オブジェクトの値を確認します。

設定変更の通知のイネーブル化

設定変更のたびに entConfigChange 通知を生成するようにルータを設定するには、CLI から次のコマンドを入力します。通知をディセーブルにするには、コマンドの **no** 形式を使用します。

```
Router(config)# snmp-server enable traps entity
Router(config)# no snmp-server enable traps entity
```

FRU ステータスの変更

FRU 通知がイネーブルの場合、ルータは FRU ステータスの変更に応じて次の通知を生成します。

- cefcModuleStatusChange : FRU の動作ステータス（cefcModuleOperStatus）が変更されました。
- cefcFRUInserted : FRU がシャーシに挿入されました。通知は、FRU の entPhysicalIndex および FRU が挿入されたコンテナを示します。
- cefcFRURemoved : FRU がシャーシから取り外されました。通知は、FRU の entPhysicalIndex および FRU が取り外されたコンテナを示します。

（注） これらの通知の詳細については、CISCO-ENTITY-FRU-CONTROL-MIB を参照してください。

FRU 通知のイネーブル化

FRU イベントの通知を生成するようにルータを設定するには、CLI から次のコマンドを入力します。通知をディセーブルにするには、コマンドの **no** 形式を使用します。

```
Router(config)# snmp-server enable traps fru-ctrl
Router(config)# no snmp-server enable traps fru-ctrl
```

SNMP を介して FRU 通知をイネーブルにするには、cefcMIBEnableStatusNotification を true(1) に設定します。通知をディセーブルにするには、cefcMIBEnableStatusNotification を false(2) に設定します。

Quality of Service のモニタリング

ここでは、設定の Quality of Service（QoS）の使用に関する次の情報を提供します。

- CISCO-CLASS-BASED-QOS-MIB の概要
- CISCO-CLASS-BASED-QOS-MIB を使用した QoS 設定の表示
- CISCO-CLASS-BASED-QOS-MIB を使用した QoS のモニタリング
- QoS 統計情報の処理に関する考慮事項
- サンプル QoS アプリケーション

CISCO-CLASS-BASED-QOS-MIB の概要

CISCO-CLASS-BASED-QOS-MIB は、モジュラ Quality of Service コマンドライン インターフェイス（モジュラ QoS CLI）をサポートするシスコ プラットフォームに、Quality of Service（QoS）の設定情報および統計情報への読み取り専用アクセスを提供します。

CISCO-CLASS-BASED-QOS-MIB のオブジェクト関係

CISCO-CLASS-BASED-QOS-MIB テーブル内の移動方法を理解するには、さまざまな QoS オブジェクト間の関係を理解することが重要です。QoS オブジェクトの構成要素は次のとおりです。

- match ステートメント：分類の目的でパケットを識別する特定の一致基準。
- クラス マップ:さまざまなカテゴリにパケットを分類するために使用する 1 つまたは複数の match ステートメントが含まれているユーザ定義のトラフィック クラス。
- 機能アクション：QoS 機能。機能には、ポリシング、トラフィック シェーピング、キューイング、ランダム検出、およびパケット マーキングが含まれます。トラフィックの分類後、各トラフィック クラスにアクションを適用します。
- ポリシー マップ：ユーザ定義のクラス マップに QoS 機能のアクションを関連付けるユーザ定義のポリシー。
- サービス ポリシー：インターフェイスに関連付けられたポリシー マップ。

MIB は、次のインデックスを使用して QoS 機能を識別し、その機能のインスタンスを区別します。

- cbQosObjectsIndex：ルータの各 QoS 機能を識別します。
- cbQoSConfigIndex：QoS 設定のタイプを識別します。このインデックスは、同一の設定を持つ QoS オブジェクトで共有されます。
- cbQosPolicyIndex：固有のサービス ポリシーを識別します。

QoS MIB 情報ストレージ

CISCO-CLASS-BASED-QOS-MIB 情報は、次の場所に保存されます。

- 設定インスタンス：すべてのクラス マップ、ポリシー マップ、match ステートメント、および機能アクションの設定パラメータが含まれます。同一のインスタンスが複数ある場合があります。同じ QoS 機能の複数のインスタンスは、cbQosConfigIndex で識別される 1 つの設定オブジェクトを共有します。
- 実行時統計情報インスタンス：任意の設定済み QoS ポリシーが適用される前後のトラフィック クラスによるサマリーおよびレートが含まれます。また、詳細な機能固有の統計情報をポリシー マップ機能の選択に使用できます。それぞれに一意の実行時インスタンスがあります。QoS 機能の複数のインスタンスに個別の統計情報オブジェクトがあります。QoS オブジェクトの実行時インスタンスそれぞれに、一致する設定を持つ複数のオブジェクトを区別する一意の識別子（cbQosObjectsIndex）が割り当てられます。

CISCO-CLASS-BASED-QOS-MIB を使用した QoS 設定の表示

ここでは、QoS 設定が CISCO-CLASS-BASED-QOS-MIB テーブルに保存される方法の例を示します。サンプルは、QoS オブジェクトによってグループ化された情報を示します。ただし、SNMP クエリーの実際の出力は、次のような QoS 情報を示すことがあります。

（注） これはすべての QoS 情報の一部の表示です。

```
getmany -v2c 9.0.0.55 ciscoCBQosMIB
cbQosIfType.64 = mainInterface(1)
cbQosIfType.66 = mainInterface(1)
cbQosPolicyDirection.64 = input(1)
cbQosPolicyDirection.66 = output(2)
```

```
cbQosIfIndex.64 = 4
cbQosIfIndex.66 = 4
cbQosFrDLCI.64 = 0
cbQosFrDLCI.66 = 0
cbQosAtmVPI.64 = 0
cbQosAtmVPI.66 = 0
cbQosAtmVCI.64 = 0
cbQosAtmVCI.66 = 0
cbQosEntityIndex.64 = 0
cbQosEntityIndex.66 = 0
cbQosConfigIndex.64.64 = 15348192
cbQosConfigIndex.64.7282691 = 12103539
cbQosConfigIndex.64.15123441 = 1593
cbQosConfigIndex.64.15755442 = 1594
cbQosConfigIndex.66.66 = 15889568
cbQosConfigIndex.66.1907619 = 15971699
cbQosConfigIndex.66.9319458 = 1594
cbQosConfigIndex.66.15082481 = 1593
cbQosObjectsType.64.64 = policymap(1)
cbQosObjectsType.64.7282691 = police(7)
cbQosObjectsType.64.15123441 = classmap(2)
cbQosObjectsType.64.15755442 = matchStatement(3)
cbQosObjectsType.66.66 = policymap(1)
cbQosObjectsType.66.1907619 = queueing(4)
cbQosObjectsType.66.9319458 = matchStatement(3)
cbQosObjectsType.66.15082481 = classmap(2)
cbQosParentObjectsIndex.64.64 = 0
cbQosParentObjectsIndex.64.7282691 = 15123441
cbQosParentObjectsIndex.64.15123441 = 64
cbQosParentObjectsIndex.64.15755442 = 15123441
cbQosParentObjectsIndex.66.66 = 0
cbQosParentObjectsIndex.66.1907619 = 15082481
cbQosParentObjectsIndex.66.9319458 = 15082481
cbQosParentObjectsIndex.66.15082481 = 66
cbQosPolicyMapName.15348192 = policy-police
cbQosPolicyMapName.15889568 = policy-bw
cbQosPolicyMapDesc.15348192 =
cbQosPolicyMapDesc.15889568 =
cbQosCMName.1593 = class-default
cbQosCMDesc.1593 =
cbQosCMInfo.1593 = matchAny(3)
.....
.....
```

CISCO-CLASS-BASED-QOS-MIB を使用した QoS のモニタリング

ここでは、CISCO-CLASS-BASED-QOS-MIB テーブルの QoS 統計情報を確認してルータの QoS をモニタする方法について説明します。

（注） CISCO-CLASS-BASED-QOS-MIB には、CLI の **show** コマンドの出力に表示される内容よりも多くの情報が含まれている場合があります。

表 A-1 に、QoS 統計情報テーブルのタイプを示します。

表 A-1　QoS 統計情報テーブル

QoS テーブル	統計
cbQosCMStatsTable	クラス マップ：QoS ポリシーの実行前後のパケット数、バイト数、およびビット レート。ドロップされたパケット数およびバイト数。
cbQosMatchStmtStatsTable	match ステートメント：QoS ポリシーを実行する前のパケット数、バイト数、およびビット レート。
cbQosPoliceStatsTable	ポリシング アクション：ポリシー アクションに適合、違反、ポリシー アクションを超過したパケット数、バイト数、およびビット レート。
cbQosQueueingStatsTable	キューイング：廃棄されたパケット数とバイト数、およびキューの深さ。
cbQosTSStatsTable	トラフィック シェーピング：遅延およびドロップされたパケット数とバイト数、機能の状態、およびキュー サイズ。
cbQosREDClassStatsTable	ランダム早期検出：キューがいっぱいになったときにドロップされたパケット数とバイト数、および送信されたバイト数とオクテット数。

QoS 統計情報の処理に関する考慮事項

ルータは、ほとんどの QoS 統計情報の 64 ビット カウンタを保持します。ただし、一部の QoS カウンタは、1 ビット オーバーフロー フラグが設定された 32 ビット カウンタとして実装されます。以降のサンプルでは、これらのカウンタは 33 ビット カウンタとして示されています。

QoS カウンタ統計情報にアクセスするときは、次の点を考慮します。

- SNMPv2c または SNMPv3 アプリケーション：*cbQosxxx64* MIB オブジェクトを通じて QoS カウンタの 64 ビット全体にアクセスします。
- SNMPv1 アプリケーション：次のように MIB の QoS 統計情報にアクセスします。
  - cb*Qosxxx* MIB オブジェクトを通じてカウンタの下位 32 ビットにアクセスします。
  - *cbQosxxxOverflow* MIB オブジェクトを通じてカウンタの上位 32 ビットにアクセスします。

サンプル QoS 統計情報テーブル

ここに挙げるサンプルは、CISCO-CLASS-BASED-QOS-MIB 統計情報テーブルのカウンタを示しています。

- 図 A-2 は、cbQosCMStatsTable のカウンタおよびこれらのカウンタや他の統計情報にアクセスするためのインデックスを示しています。
- 図 A-3 は、cbQosMatchStmtStatsTable、cbQosPoliceStatsTable、cbQosQueueingStatsTable、cbQosTSStatsTable、および cbQosREDClassStatsTable のカウンタを示しています。

使いやすいように、次の図は、カウンタが 3 個のオブジェクトとして実装されている場合でも、そのカウンタを 1 つのオブジェクトとして示しています。たとえば、`cbQosCMPrePolicyByte` は次のオブジェクトとして実装されています。

- cbQosCMPrePolicyByteOverflow
- cbQosCMPrePolicyByte

- cbQosCMPrePolicyByte64

図 A-2　QoS クラス マップの統計情報およびインデックス

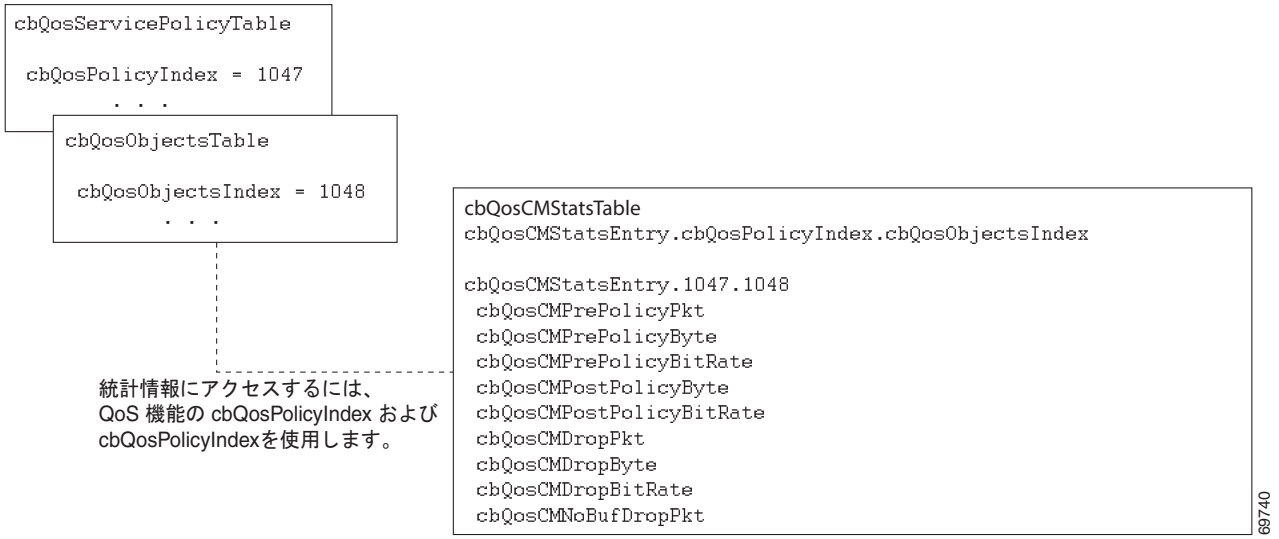

図 A-3　QoS 統計情報テーブル

サンプル QoS アプリケーション

ここでは、CISCO-CLASS-BASED-QOS-MIB から QoS 課金操作で使用する情報を取得する方法を示すコード例を示します。課金アプリケーションを開発する場合には、これらの例を使用できます。内容は、次のとおりです。

- サービス ポリシーのカスタマー インターフェイスの確認

- QoS 課金情報の取得

サービス ポリシーのカスタマー インターフェイスの確認

ここでは、CISCO-CLASS-BASED-QOS-MIB でサービス ポリシーが設定されたカスタマー インターフェイスを確認し、さらにアプリケーションで処理（QoS サービスの課金など）するようにこれらのインターフェイスにマークを付けるサンプル アルゴリズムを示します。

アルゴリズムでは、カスタマー インターフェイスごとに 2 つの SNMP **get-next** 要求を使用します。たとえば、ルータに 2000 のカスタマー インターフェイスがある場合、これらのインターフェイスに送受信サービス ポリシーが関連付けられているかどうかを確認するには、4000 の SNMP **get-next** 要求が必要です。

（注） このアルゴリズムは情報提供だけを目的としています。アプリケーションのニーズとは異なる可能性があります。

MIB を確認して、どのインターフェイスがカスタマーに関連付けられているかを調べます。サービス ポリシーがカスタマー インターフェイスの送信および受信方向に関連付けられているかどうかを示すフラグのペアを作成します。非カスタマー インターフェイスに TRUE のマークを付けます（したがって、これらのインターフェイスにこれ以上の処理は不要です）。

```
FOR each ifEntry DO
  IF (ifEntry represents a customer interface) THEN
     servicePolicyAssociated[ifIndex].transmit = FALSE;
     servicePolicyAssociated[ifIndex].receive = FALSE;
  ELSE
     servicePolicyAssociated[ifIndex].transmit = TRUE;
     servicePolicyAssociated[ifIndex].receive = TRUE;
  END-IF
END-FOR
```

cbQosServicePolicyTable を調べてし、それに関連付けられたサービス ポリシーがある各カスタマー インターフェイスにマークを付けます。また、インターフェイスの方向に注意してください。

```
x = 0;
done = FALSE;
WHILE (!done)
  status = snmp-getnext (
           ifIndex = cbQosIfIndex.x,
           direction = cbQosPolicyDirection.x
  );
  IF (status != 'noError') THEN
     done = TRUE
  ELSE
     x = extract cbQosPolicyIndex from response;
     IF (direction == 'output') THEN
       servicePolicyAssociated[ifIndex].transmit = TRUE;
     ELSE
       servicePolicyAssociated[ifIndex].receive = TRUE;
     END-IF
  END-IF
END-WHILE
```

カスタマー インターフェイスにサービス ポリシーが関連付けられていないケースを管理します。

```
FOR each ifEntry DO
  IF (!servicePolicyAssociated[ifIndex].transmit) THEN
    Perform processing for customer interface without a transmit service policy.
  END-IF
```

```
  IF (!servicePolicyAssociated[ifIndex].receive) THEN
    Perform processing for customer interface without a receive service policy.
  END-IF
END-FOR
```

QoS 課金情報の取得

ここでは、QoS 課金操作に CISCO-CLASS-BASED-QOS-MIB を使用するサンプル アルゴリズムについて説明します。アルゴリズムは、定期的にポストポリシー入出力統計情報を取得し、それらを組み合わせて、結果を課金データベースに送信します。

このアルゴリズムでは、次が使用されます。

- カスタマー インターフェイスごとに 1 つの SNMP **get** 要求：ifAlias を取得するため。
- カスタマー インターフェイスごとに 2 つの SNMP **get-next** 要求:サービス ポリシー インデックスを取得するため。
- ポリシーの各オブジェクトのカスタマー インターフェイスごとに 2 つの SNMP **get-next** 要求：ポストポリシー バイトを取得するため。たとえば、ポリシーに 100 個のインターフェイスと 10 個のオブジェクトがある場合、アルゴリズムは 2000 個の **get-next** 要求（2 x 100 x 10）を必要とします。

（注） このアルゴリズムは情報提供だけを目的としています。アプリケーションのニーズとは異なる可能性があります。

顧客の課金情報を設定します。

```
FOR each ifEntry DO
  IF (ifEntry represents a customer interface) THEN
     status = snmp-getnext (id = ifAlias.ifIndex);
     IF (status != 'noError') THEN
        Perform error processing.
     ELSE
        billing[ifIndex].isCustomerInterface = TRUE;
        billing[ifIndex].customerID = id;
        billing[ifIndex].transmit   = 0;
        billing[ifIndex].receive    = 0;
     END-IF
  ELSE
     billing[ifIndex].isCustomerInterface = FALSE;
  END-IF
END-FOR
```

課金情報を取得します。

```
x = 0;
done = FALSE;
WHILE (!done)
  response = snmp-getnext (
             ifIndex = cbQosIfIndex.x,
             direction = cbQosPolicyDirection.x
  );
  IF (response.status != 'noError') THEN
     done = TRUE
  ELSE
     x = extract cbQosPolicyIndex from response;
     IF (direction == 'output') THEN
        billing[ifIndex].transmit = GetPostPolicyBytes (x);
     ELSE
        billing[ifIndex].receive = GetPostPolicyBytes (x);
```

```
      END-IF
    END-IF
  END-WHILE
```

ポストポリシー バイトの数を課金目的で決定します。

```
GetPostPolicyBytes (policy)
  x = policy;
  y = 0;
  total = 0;
  WHILE (x == policy)
     response = snmp-getnext (type = cbQosObjectsType.x.y);
     IF (response.status == 'noError')
        x = extract cbQosPolicyIndex from response;
        y = extract cbQosObjectsIndex from response;
        IF (x == policy AND type == 'classmap')
           status = snmp-get (bytes = cbQosCMPostPolicyByte64.x.y);
           IF (status == 'noError')
                   total += bytes;
           END-IF
        END-IF
     END-IF
  END-WHILE
RETURN total;
```

ルータ インターフェイスのモニタリング

ここでは、インターフェイスのサービスに影響を及ぼす可能性のある問題または条件があるかどうかを確認するためにルータ インターフェイスのステータスをモニタする方法に関する情報を提供します。インターフェイスがダウンしているか、または問題が発生しているかどうかを確認するには、次の操作を実行します。

インターフェイスの運用ステータスおよび管理ステータスの確認

インターフェイスのステータスを確認するには、インターフェイスの次の IF-MIB オブジェクトを表示します。

- ifAdminStatus：インターフェイスの管理上設定された（望ましい）状態。ifAdminStatus を使用して、インターフェイスをイネーブルまたはディセーブルにします。
- ifOperStatus：インターフェイスの現在の動作状態。

linkDown および linkUp 通知のモニタリング

インターフェイスに障害が発生したかどうかを確認するには、インターフェイスの linkDown および linkUp 通知をモニタできます。これらの通知をイネーブルにする方法については、「インターフェイスの linkUp/linkDown 通知のイネーブル化」（P.A-33）を参照してください。

- linkDown：インターフェイスに障害が発生したか、障害が発生しそうであることを示します。
- linkUp：インターフェイスがダウン状態ではなくなったことを示します。

インターフェイスの linkUp/linkDown 通知のイネーブル化

ルータ インターフェイスの状態がアップ（ready）またはダウン（not ready）に変わったときに通知を送信するように SNMP を設定するには、次の手順を実行して、linkUp および linkDown 通知をイネーブルにします。

ステップ 1　次の CLI コマンドを発行して、ほとんど（ただし、すべてとは限らない）インターフェイスの linkUp および linkDown 通知をイネーブルにします。

```
Router(config)# snmp-server enable traps snmp linkdown linkup
```

ステップ 2　linkUp および linkDown 通知がそのインターフェイスに対してイネーブルになっているか、またはディセーブルになっているかを確認するには、各インターフェイスの ifLinkUpDownTrapEnable オブジェクト（ifXTable IF-MIB）の設定を表示します。

ステップ 3　インターフェイスで linkUp および linkDown 通知をイネーブルにするには、ifLinkUpDownTrapEnable を enabled(1) に設定します。インターフェイスの最下位レイヤに対してだけ linkDown 通知を送信するようにルータを設定するには、「linkDown 通知の SNMP 通知フィルタリング」（P.A-34）を参照してください。

ステップ 4　linkUp および linkDown 通知についてインターネット技術特別調査委員会（IETF）標準をイネーブルにするには、次のコマンドを発行します。（IETF 標準は RFC 2233 に基づいています）。

```
Router(config)# snmp-server trap link ietf
```

ステップ 5　通知をディセーブルにするには、適切なコマンドの **no** 形式を使用します。

linkDown 通知の SNMP 通知フィルタリング

メイン インターフェイスがダウンした場合にだけ SNMP が linkDown 通知を送信するように linkDown 通知をフィルタリングするには、SNMP 通知フィルタリング機能を使用します。インターフェイスがダウンすると、そのすべてのサブインターフェイスがダウンするため、サブインターフェイスごとに多数の linkDown 通知が発生します。この機能では、これらのサブインターフェイス通知が除外されます。

この機能はデフォルトでオフになっています。SNMP 通知フィルタリング機能をイネーブルにするには、次の CLI コマンドを実行します。機能をディセーブルにするには、コマンドの **no** 形式を入力します。

```
[no] snmp ifmib trap throttle
```

お客様へのトラフィックの課金

ここでは、SNMP インターフェイス カウンタおよび QoS データ情報を使用して、お客様に対するトラフィックの課金額を決定する方法について説明します。インターフェイスに関連付けられた QoS サービス ポリシーが、そのインターフェイスのトラフィックのポリシングであることを示すシナリオもあります。

ここでは、次の内容について説明します。

入力および出力インターフェイス カウント

ルータは、入力インターフェイス上で受信され、出力インターフェイスで送信されるパケットとバイトの数に関する情報を保持します。

IF-MIB カウンタのサポートに関する詳細な制約事項については、「IF-MIB（RFC 2863）」（P.3-110）を参照してください。

IF-MIB カウンタのサポートに関する次の重要な情報を参照してください。

- 特に明記していない限り、すべての IF-MIB カウンタは、Cisco ASR 1000 シリーズ ルータ インターフェイスでサポートされています。
- IF-MIB の大容量カウンタのサポートについては、シスコは RFC 2863 標準に準拠しています。RFC 2863 標準では、次のように動作するインターフェイスについて規定しています。
  - 20,000,000 bps 以下で、32 ビット バイトおよびパケット カウンタがサポートされている*必要があります*。
  - 20,000,000 bps 以下、650,000,000 bps 未満で、32 ビット パケット カウンタおよび 64 ビット オクテット カウンタがサポートされている*必要があります*。
  - 650,000,000 bps 以上で、64 ビット パケット カウンタ*および* 64 ビット オクテット カウンタがサポートされている*必要があります*。
- QoS サービス ポリシーがインターフェイスに関連付けられている場合、ルータはインターフェイス上のトラフィックにポリシー ルールを適用し、インターフェイスのパケットおよびバイト カウントを増やします。

次の CISCO-CLASS-BASED-QOS-MIB オブジェクトは、インターフェイス カウントを提供します。

- cbQosCMDropPkt および cbQosCMDropByte（cbQosCMStatsTable）：ユーザがサービス ポリシーによって設定された制限を超過したためにドロップされたパケットおよびバイトの総数。これらのカウントには、サービス ポリシーの制限を超過したためにドロップされたパケットおよびバイトのみが含まれます。他の理由でドロップされたパケットとバイトは含まれません。
- cbQosPoliceConformedPkt および cbQosPoliceConformedByte（cbQosPoliceStatsTable）：サービス ポリシーの制限に適合し、送信されたパケットおよびバイトの総数。

お客様に課金するトラフィック量の決定

特定のお客様に課金可能なインターフェイス上のトラフィック量を判断するには、次の手順を実行します。

ステップ 1　お客様に適用するインターフェイス上のサービス ポリシーを決定します。

ステップ 2　お客様のトラフィックを定義するために使用されるサービス ポリシーとクラス マップのインデックス値を決定します。次の手順でこの情報が必要です。

ステップ 3　トラフィック ジェネレータを使用してトラフィックを生成します。データ レートは、そのポリシーの適合バースト（bc）/超過バースト（BE）に設定された値より大きくする必要があります。

ステップ 4　（任意）サービス ポリシー数制限を超過したためにドロップお客様のトラフィックの量を判断するには、お客様の cbQosCMDropPkt オブジェクト（cbQosCMStatsTable）にアクセスします。

QoS トラフィック ポリシングを示すシナリオ

ここでは、SNMP QoS 統計情報を使用して特定のお客様に課金可能なインターフェイス上のトラフィック量を判断するシナリオについて説明します。サービス ポリシーをインターフェイス上のトラフィックに適用するとパケット カウントがどのような影響を受けるかも示します。

シナリオを作成するには、以降の項で説明する次の手順に従ってください。

1. サービス ポリシーを作成し、インターフェイスに関連付けます。
2. サービス ポリシーをインターフェイス上のトラフィックに適用する前にパケット カウントを表示します。
3. **ping** コマンドを発行して、インターフェイスのトラフィックを生成します。サービス ポリシーがトラフィックに適用されることに注意してください。
4. お客様に課金するトラフィック量を判断するには、サービス ポリシーが適用された後のパケット カウントを表示します。
   - 適合したパケット：サービス ポリシーで設定された範囲内のパケット数で、これに対してお客様に課金できます。
   - 超過またはドロップされたパケット：サービス ポリシーの範囲外であるために送信されなかったパケットの数。これらのパケットは、お客様に課金できません。

（注） 上記のシナリオでは、Cisco ASR 1000 シリーズ ルータは中間デバイスとして使用されます（つまり、トラフィックは他の場所で発生し、別のデバイスを宛先とします）。

サービス ポリシー コンフィギュレーション

このシナリオは、次のポリシーマップ コンフィギュレーションを使用します。ポリシー マップを作成する方法については、『*Cisco ASR 1000 Series Router Software Configuration Guide*』の「Configuring Quality of Service」を参照してください。

```
Policy Map test-police
    Class class-default
     police cir 1000000 bc 10000 be 20000
       conform-action transmit
       exceed-action drop
       violate-action drop

interface GigabitEthernet1/1/5
 ip address 15.1.0.52 255.0.0.0
 no negotiation auto
 service-policy output test-police
end
```

サービス ポリシーの適用前のパケット カウント

次の CLI および SNMP の出力は、サービス ポリシーが適用される前のインターフェイスの出力トラフィックを示しています。

CLI コマンド出力

```
Router#sh policy-map interface gi 1/1/5

 GigabitEthernet1/1/5

  Service-policy output: test-police
```

```
  Class-map: class-default (match-any)
    0 packets, 0 bytes
    5 minute offered rate 0 bps, drop rate 0 bps
    Match: any
    police:
        cir 1000000 bps, bc 10000 bytes, be 20000 bytes
      conformed 0 packets, 0 bytes; actions:
        transmit
      exceeded 0 packets, 0 bytes; actions:
        drop
      violated 0 packets, 0 bytes; actions:
        drop
      conformed 0 bps, exceed 0 bps, violate 0 bps
```

SNMP 出力

```
ptolemy:4> getmany 9.0.0.52 cbQosIfIndex
cbQosIfIndex.290 = 18
ptolemy:5> getone 9.0.0.52 ifDescr.18
ifDescr.18 = GigabitEthernet1/1/5
ptolemy:6>

getmany 9.0.0.52 cbQosCMDropPkt cbQosCMDropByte
cbQosCMDropPkt.290.9756705 = 0
cbQosCMDropByte.290.9756705 = 0
ptolemy:77>
```

サービス ポリシーの適用後のパケット カウント

トラフィック ジェネレータを使用してトラフィックを生成したら、**police** コマンドで設定された認定情報レート（CIR）を超過およびこれに適合したパケットの数を確認します。

- 19351 個のパケットがポリシング レートに適合し、送信されました
- 80 個のパケットがポリシング レートを超過し、ドロップされました
- 16066130 個のパケットがポリシング レートに違反し、ドロップされました

次の CLI および SNMP 出力は、サービス ポリシーが適用された後のインターフェイス上のカウントを示しています。オブジェクト cbQosCMDropPkt は、超過および違反したパケットの合計を示し、cbQosCMDropByte は超過および違反したバイトの合計を示します。（出力では、超過および違反したパケット カウントが太字で示されています）。

CLI コマンド出力

```
Router#sh show policy-map int gi 1/1/5

 GigabitEthernet1/1/5

  Service-policy output: test-police

    Class-map: class-default (match-any)
      16085561 packets, 1994609369 bytes

      5 minute offered rate 16051000 bps, drop rate 16032000 bps
      Match: any
      police:
          cir 1000000 bps, bc 10000 bytes, be 10000 bytes
        conformed 19351 packets, 2399329 bytes; actions:
          transmit
```

```
        exceeded 80 packets, 9920 bytes; actions:
          drop
        violated 16066130 packets, 1992200120 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 16032000 bps
Router#
```

SNMP 出力

```
getmany 9.0.0.52 cbQosCMDropPkt cbQosCMDropByte
cbQosCMDropPkt.290.9756705 = 16066210
cbQosCMDropByte.290.9756705 = 1992210040
ptolemy:77>
        . . .
```

IF-MIB カウンタの使用

この項では、IF-MIB カウンタと、さまざまなインターフェイスおよびサブインターフェイス上でそれらのカウンタを使用する方法について説明します。サブインターフェイス カウンタはプロトコル固有です。ここでは、ATM インターフェイスの IF-MIB カウンタを扱います。

IF-MIB カウンタは、下位層と上位層に関して定義されます。

- ifInDiscards：上位層のプロトコルへの配信を妨げるエラーが検出されなくても、廃棄された着信パケットの数。このようなパケットを廃棄する理由の 1 つは、バッファ スペースを空けることです。
- IfInErrors：エラー、パケット指向インターフェイスの上位層プロトコルへの配信を妨げるエラーが含まれている着信パケットの数。
- ifInUnknownProtos：パケット指向インターフェイスのプロトコルが不明またはサポート対象外であることが原因で廃棄された、インターフェイス経由で受信したパケットの数。
- ifOutnDiscards：送信を妨げるエラーが検出されなくても、廃棄された発信パケットの数。このようなパケットを廃棄する理由の 1 つは、バッファ スペースを空けることです。
- ififOutErrors：パケット指向インターフェイスのエラーのために送信できなかった送信パケットの数。

カウンタの論理フローは次のとおりです。

1. パケットがインターフェイスに到着したら、次のことを確認します。
   a. パケット内のエラー：エラーが検出された場合、ifInErrors を増分し、パケットをドロップします。
   b. プロトコル エラー：エラーが検出された場合、ifInUnknownProtos を増分し、パケットをドロップします。
   c. リソース（バッファ）：リソースを取得できない場合は、ifInDiscards を増分し、パケットをドロップします。
   d. ifInUcastPkts/ifInNUcastPkts を増分し、パケットを処理します（この時点で、ifInOctets をパケットのサイズで増分します）。
2. パケットがインターフェイスから送信される場合：
   a. ifOutUcasePkts/ifOutNUcastPkts を増分します（ここで、ifOutOctets もパケットのサイズで増分します）。
   b. パケット内のエラーをチェックし、パケットにエラーがある場合は、ifOutErrors を増分し、パケットをドロップします。

c. リソース（バッファ）をチェックし、リソースを取得できない場合は、ifOutDiscards を増分し、パケットをドロップします。

次の出力は、IF-MIB エントリの例です。

IfXEntry ::=

```
SEQUENCE {
    ifName                  DisplayString,
    ifInMulticastPkts       Counter32,
    ifInBroadcastPkts       Counter32,
    ifOutMulticastPkts      Counter32,
    ifOutBroadcastPkts      Counter32,
    ifHCInOctets            Counter64,
    ifHCInUcastPkts         Counter64,
    ifHCInMulticastPkts     Counter64,
    ifHCInBroadcastPkts     Counter64,
    ifHCOutOctets           Counter64,
    ifHCOutUcastPkts        Counter64,
    ifHCOutMulticastPkts    Counter64,
    ifHCOutBroadcastPkts    Counter64,
    ifLinkUpDownTrapEnable  INTEGER,
    ifHighSpeed             Gauge32,
    ifPromiscuousMode       TruthValue,
    ifConnectorPresent      TruthValue,
    ifAlias                 DisplayString,
    ifCounterDiscontinuityTime TimeStamp
```

サンプル カウンタ

大容量カウンタは、基本的な ifTable カウンタの 64 ビット バージョンです。基本的なセマンティクスは、32 ビット対応と同じです。構文は 64 ビットに拡張されます。

表 A-2 に、容量カウンタのオブジェクト ID（OID）を示します。

表 A-2　容量カウンタのオブジェクト ID

名前	オブジェクト ID（OID）
ifHCInOctets	::= { ifXEntry 6 }
ifHCInUcastPkts	::= { ifXEntry 7 }
ifHCInMulticastPkts	::= { ifXEntry 8 }
ifHCInBroadcastPkts	::= { ifXEntry 9 }
ifHCOutOctets	::= { ifXEntry 10 }
ifHCOutUcastPkts	::= { ifXEntry 11 }
ifHCOutMulticastPkts	::= { ifXEntry 12 }
ifHCOutBroadcastPkts	::= { ifXEntry 13 }
ifLinkUpDownTrapEnable	::= { ifXEntry 14 }
ifHighSpeed	::= { ifXEntry 15 }
ifPromiscuousMode	::= { ifXEntry 16 }
ifConnectorPresent	::= { ifXEntry 17 }
ifAlias	::= { ifXEntry 18 }
ifCounterDiscontinuityTime	::= { ifXEntry 19 }

関連情報および有益なリンク

次の URL から、シスコ IF-MIB カウンタに関する有用な情報にアクセスできます。

- SNMP カウンタに関する FAQ：

  http://www.cisco.com/en/US/customer/tech/tk648/tk362/technologies_q_and_a_item09186a00800b69ac.shtml

- Cisco IOS MIB ツールへのアクセス：

  http://tools.cisco.com/ITDIT/MIBS/servlet/index

SIP および SPA の概要

次に、Cisco SIP および SPA（共有ポート アダプタ）の一般的な特性について説明します。

- Cisco ASR 1000 シリーズ SPA インターフェイス プロセッサ（SIP）は、次のようなキャリア カードです。
  - ラインカードのようにルータのスロットに挿入されます。このカード自体にネットワーク接続機能はありません。
  - 1 つまたは複数の SPA を装着するためのサブスロットが 1 つまたは複数装備されています。SPA にはネットワーク接続用のインターフェイス ポートがあります。
  - 通常動作時にすべてのサブスロットに動作する SPA を取り付けるか、またはすべての空のサブスロットにブランク フィラー プレート（SPA-BLANK=）を取り付けた状態で、ルータに装着されます。
  - サブスロットに SPA を装着した状態で、活性挿抜（OIR）を実行できます。SPA も活性挿抜をサポートするので、SIP とは無関係に着脱可能です。
- 共有ポート アダプタ（SPA）は次のようなモジュラ タイプのポート アダプタです。
  - 互換性のある SIP キャリア カードのサブスロットに搭載され、ネットワーク接続を提供するとともにインターフェイスのポート密度を向上させます。SIP のタイプに応じて、1 つまたは複数の SPA を SIP に搭載できます。
  - ネットワーク接続以外のサービスを提供し、互換性のあるカードのサブスロットに搭載されます。たとえば、IPSec VPN SPA は、IP セキュリティ（IPSec）暗号化/復号化、総称ルーティング カプセル化（GRE）、インターネット キー交換（IKE）キー生成などのサービスを提供します。
  - シングルハイト（1 つの SIP サブスロットに挿入）およびダブルハイト（縦に並んだ 2 つの単一 SIP サブスロットに挿入）で使用できます。

（注） SPA-1X10GE-WL-V2 は Cisco IOS XE Release 3.3.0 S および Cisco IOS Release 15.1(2)S 以降の Cisco ASR1K プラットフォームでサポートされています。

（注） 1 ポート 10GE LAN/WAN-PHY 共有ポート アダプタ（SPA-1X10GE-WL-V2）は、両側で同じモード（LAN モードまたは WAN モード）にする必要があります。

（注）　SPA-1X10GE-WL-V2（LAN モードで設定）は、SPA-1X10GE-L-V2（LAN SPA）と互換性があります。

LAN-PHY モードの設定

1 ポート 10GE LAN/WAN-PHY 共有ポート アダプタ（SPA-1X10GE-WL-V2）で LAN-PHY モードを設定するには、次のコマンドを使用します。

```
show controllers wanphy interface-path-id [alarms | all | registers]
configure terminal
controller wanphy interface-path-id
lanmode on
end
hw-module subslot interface-path-id reload
show controllers wanphy interface-path-id [alarms | all | registers]
```

（注）　LAN-PHY モードを設定し、SPA をリロードした後、すべてのリンクがアップ状態になります。

（注）　Cisco IOS Release 15.1(2)S から、1 ポート 10GE LAN/WAN-PHY 共有ポート アダプタ（SPA-1X10GE-WL-V2）は、LAN モードと WAN モードの両方をサポートします。

SIP ハードウェア タイプの表示

Cisco ASR 1000 シリーズ ルータに搭載された SIP ハードウェア タイプを確認するには、show platform コマンドを使用できます。Cisco ASR 1000 シリーズ ルータには、SIP ハードウェア情報を提供するコマンドがあります。各 SIP/SPA カードの詳細を出力するサブコマンドは他にもあります。次の例は、このようなコマンドの一部のリストです。

例 A-9　show platform コマンドの例

次の例に、Cisco ASR 1000 シリーズ ルータにおける **show platform** コマンドの出力を示します。

```
Router#sh platform
Chassis type: ASR1006

Slot      Type                State                 Insert time (ago)
--------- ------------------- --------------------- -----------------
0         ASR1000-SIP10       ok                    06:19:03
 0/0      SPA-1XOC12-POS      ok                    06:17:25
 0/1      SPA-2XCT3/DS0       ok                    06:17:25
 0/2      SPA-2XT3/E3         ok                    06:17:25
 0/3      SPA-8X1GE-V2        ok                    06:17:34
1         ASR1000-SIP10       ok                    06:19:03
 1/0      SPA-1X10GE-L-V2     ok                    06:17:36
 1/1      SPA-5X1GE-V2        ok                    06:17:25
```

```
 1/2      SPA-8X1FE-TX-V2     ok                    06:17:36
2         ASR1000-SIP10       ok                    06:19:03
 2/0      SPA-2X1GE-V2        ok                    06:17:36
 2/1      SPA-10X1GE-V2       ok                    06:17:36
 2/2      SPA-2XOC3-POS       ok                    06:17:36
R0        ASR1000-RP1         ok, active            06:19:03
R1                            unknown               06:19:03
F0        ASR1000-ESP10       ok, active            06:19:03
P0        ASR1006-PWR-AC      ok                    06:18:25
P1        ASR1006-FAN         ok                    06:18:25

Slot      CPLD Version        Firmware Version
--------- ------------------- ----------------------------------------
0         06120701            12.2(20070802:195019) [gschnorr-mcp_...
1         06120701            12.2(20070802:195019) [gschnorr-mcp_...
2         06120701            12.2(20070802:195019) [gschnorr-mcp_...
R0        0706210B            12.2(20070807:170946) [gschnorr-mcp_...
R1        N/A                 N/A
F0        07021400            12.2(20070802:195019) [gschnorr-mcp_...
```

```
Router#sh platform ?
hardware  Show platform hardware information
  software  Show platform software information
  |         Output modifiers
  <cr>

Router#sh platform har
Router#sh platform hardware ?
cpp        Cisco packet processor
  interface  Interface information
  port       port information
  slot       Slot information
  subslot    Subslot information

Router#sh platform hardware slot ?
0   SPA-Inter-Processor slot 0
  1   SPA-Inter-Processor slot 1
  2   SPA-Inter-Processor slot 2
  F0  Embedded-Service-Processor slot 0
  F1  Embedded-Service-Processor slot 1
```

```
  R0  Route-Processor slot 0
  R1  Route-Processor slot 1

Router#sh platform hardware slot 0 ?
dram     MCP85xx DRAM commands
  eobc     Show EOBC
  fan      Fan commands
  io-port  IO Port information
  led      LED-related commands
  mcu      MCU related commands
  plim     PLIM information
  sensor   Sensor information
  serdes   Serdes information
  spa      SPA related information
  voltage  Voltage commands

Router#
```