Cisco Virtual Network Management Center 2.1 XML API ガイド

2013 年 6 月 3 日

【注意】シスコ製品をご使用になる前に、安全上の注意（www.cisco.com/jp/go/safety_warning/）をご確認ください。

本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。

あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。

また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

目次

Cisco VNMC XML API の概要

ここでは、Cisco Virtual Network Management Center（VNMC）および XML Application Programming Interface（API）の一般的な情報を提供します。

VNMC および XML API の概要

ここでは、VNMC および XML API の概要を示します。

VNMC

VNMC は、Cisco 仮想サービスのデバイスおよびセキュリティ ポリシーを一元管理できる仮想アプライアンスです。エンタープライズおよびマルチテナント クラウドでの展開に対応するよう設計された VNMC は、仮想化されたデータセンターおよびクラウド環境をセキュリティ保護するために、トランスペアレントでシームレス、かつスケーラブルな管理を実現します。

GUI および XML API を搭載した VNMC では、データセンター全体に中央から仮想サービスを設定、導入、管理することができます。

VNMC は、各管理対象デバイスがパラメータで定義されたそのサブコンポーネント（つまり、オブジェクト）により表される、情報モデル主導のアーキテクチャに基づいて構築されています。このモデル中心のアプローチは、コンピュート ファイアウォールおよびエッジ ファイアウォールで仮想化インフラストラクチャを保護する柔軟でシンプルなメカニズムを提供します。

VNMC は、独自の仮想コンピュータ、ネットワーク、およびストレージ リソースを所有し、共有された物理インフラストラクチャに展開される、複数のクライアント組織またはテナントをサポートします。複数のテナントは同じインフラストラクチャ上で共存でき、各テナントは仮想リソースの管理権限を保持することができます。この複数のテナントの設計では、コンピュータ、ネットワーク、ストレージ、およびセキュリティ ポリシーを含む、各テナントに対して指定されたサービス レベル契約（SLA）を満たすことができます。

VNMC 管理情報モデル

VNMC サービス コンポーネントを構成するすべての物理および論理コンポーネントは、階層管理情報モデルで表されます。このモデルを管理情報ツリーといいます。階層構造は、最上部から始まり、親ノードと子ノードを含みます。ツリー内の各ノードは管理対象オブジェクト（またはオブジェクトのグループ）を表し、オブジェクトの管理および動作ステートを表示します。各オブジェクトは、オブジェクトとツリー内の位置を示す一意の識別名（DN）を持ちます。

管理対象オブジェクトは、ポリシー、ルール、セキュリティ プロファイル、コンピュート ファイアウォール、およびエッジ ファイアウォールなどの VNMC 管理対象エンティティを抽象化したものです。API を呼び出すことにより、オブジェクトは管理情報ツリーから読み取られ、管理情報ツリーに書き込まれます。個々の VNMC サービス コンポーネントの情報モデルは、データ管理エンジン（DME）によって一元的に保存および管理されます。ユーザが VNMC サービス コンポーネントに対して管理上の変更（エッジ ファイアウォール

プロファイルと ASA 1000V の関連付けなど）を開始すると、DME により最初にその変更が情報モデルに適用され、続いてその変更が実際の ASA 1000V に適用されます。この方法を、*モデル方式フレームワーク*といいます。

VNMC のコンポーネント

VNMC は、管理、テナント管理、ポリシー管理、リソース管理などを行うモジュール形式の機能を提供する複数のサービス コンポーネントで構成されます。これらのコンポーネントは、サービス プロバイダーまたはアプリケーションとも呼ばれます。各コンポーネントは、一意の URL を介してアクセスされます。

ここでは、VNMC コンポーネントについて説明します。

- Management Controller
- Service Registry
- Resource Manager
- Policy Manager
- VM Manager

各コンポーネントは、独自のデータ モデルと、モデル方式サービス要求を処理する DME を持ちます。各コンポーネントは、独自の管理情報ツリー ストレージ（メモリ内とパーシスタンス ストアの両方）を保持します。異なるサービス コンポーネント間でのデータ共有は、アドホックのサービス間 API 通信、パブリッシュまたはサブスクライブ メソッドによって実現されます。

Management Controller

Management Controller（コア サービスとも呼ばれます）は、VNMC 仮想マシンにシステム関連サービスを提供します。提供されるサービスは次のとおりです。

- ローカルまたは LDAP モードでユーザ ログインを許可および認証します。
- ロケール、ロール、トラスト ポイントなどのアクセス コントロールを提供します。
- IP アドレス、サブネット マスク、ゲートウェイ、ホスト名などのシステム情報を保持します。
- ユーザ入力時に、データベース バックアップ、データ エクスポート、データ インポートなどのシステム保守操作を実行します。
- 監査ログ、障害、イベント ログ、コア ダンプ ファイルなどのシステム診断情報を保持します。

Management Controller のタイプは mgmt-controller です。Management Controller に関連するすべての要求に対して API URL でこのサービス タイプを使用します。

Service Registry

Service Registry は、登録されたすべての管理対象エンドポイント（ASA 1000V または VSG など）およびサービス プロバイダー（Policy Manager または Resource Manager）に関する情報を持つ中央サービスのリポジトリです）。

（注） サービス エンドポイントは、GUI のクライアントと呼ばれます。

サービス エンドポイントとサービス プロバイダーは、Service Registry に動的に登録され、Service Registry からサービス コンポーネントに関する情報を取得します。また Service Registry はテナント管理も行い、次のサービスを提供します。

- ユーザ入力時に、組織（テナント、データセンター、アプリケーション、および階層）を作成、削除、および更新します。ポリシーとリソースが、対象となる組織に割り当てられるときに、組織の変更は Policy Manager と Resource Manager に自動的に伝播されます。
- 登録されたサービス（プロバイダー、エンドポイント、管理コントローラなど）に関する情報を保持します。
- 監査ログ、障害、イベント ログなどの診断情報を保持します。

Service Registry のタイプは service-reg です。Service Registry に関連するすべての要求に対して API URL でこのサービス タイプを使用します。Resource Manager は、論理的なコンピュート ファイアウォールとエッジ ファイアウォール、VSG との関連付けを管理します。

Resource Manager

Resource Manager は、論理的なコンピュート ファイアウォールとエッジ ファイアウォール、VSG および ASA 1000V との関連付けを管理します。エッジ ファイアウォールが ASA 1000V に関連付けられている場合、（エッジ ファイアウォールにより定義された）デバイス設定プロファイル情報は ASA 1000V にプッシュされます。今度はこれにより ASA 1000V が起動され、Policy Manager からセキュリティ プロファイルおよびポリシーをダウンロードします。Resource Manager は、次のサービスを実施します。

- ASA 1000V、VSG、および Cisco Virtual Supervisor Module（VSM）のインベントリを保持します。
- ユーザ入力によりコンピュート ファイアウォールを定義し、プロビジョニングに向けた VSG と関連付けます。
- ユーザ入力によりエッジ ファイアウォールを定義し、プロビジョニングに向けた ASA 1000V と関連付けます。
- 仮想マシン（VM）属性を取得するために VMware vCenter インスタンスと対話します。
- 検出された VM のインベントリを保持し、次の情報を VSG に配信します。
  - 名前、ハイパーバイザ、親 vApp、およびクラスタなどの VM 属性
  - ポート プロファイル名や IP アドレスなどの vNIC 属性
- VSG および ASA 1000V のプールを管理します。
- VSG および ASA 1000V のヘルス状態および障害を維持します。
- 監査ログ、障害、イベント ログなどの診断情報を保持します。

Resource Manager タイプは resource-mgr です。Resource Manager に関連するすべての要求に対して API URL でこのサービス タイプを使用します。

Policy Manager

Policy Manager はデバイス設定プロファイル、サービス ポリシー、サービス プロファイル、およびすべての関連するアーティファクトの中央リポジトリです。コンピュート ファイアウォールが Resource Manager から VSG に関連付けられている場合、VSG は Policy Manager を照会して、デバイス プロファイル、サービス プロファイル、および参照されるすべてのポリシーを解決します。その後 VSG は、Policy Manager から取得した情報に従って自身を設定します。同じプロセスがエッジ ファイアウォールが ASA 1000V に関連付けられている場合に使用されます。

Policy Manager は、次のサービスを提供します。

- ユーザ入力を使用して次の内容を定義します。
  - ファイアウォール デバイス プロファイル
  - オブジェクト グループ
  - ポリシー
  - 次の条件のポリシー ルール：
    - プロトコル、送信元または宛先、IP アドレスおよびポートなどのネットワーク属性
    - インスタンス名、ゲスト OS、ゾーン、親アプリケーション、ポート プロファイル、クラスタ、リソース プール、ホスト名、およびハイパーバイザなどの VM 属性
    - カスタム属性
    - ポリシー セット
    - セキュリティ プロファイル辞書とカスタム属性
    - サービス プロファイル
    - NTP、DNS、syslog、および障害に対する VNMC システム管理デバイス プロファイルおよびポリシー。
    - 仮想ゾーン（vZone）
- ユーザ入力を使用して、ポリシーおよびポリシー セットを割り当てます。
- サービス ポリシー、サービス プロファイル、デバイス プロファイルおよび関連オブジェクトをエッジ ファイアウォールおよびコンピュート ファイアウォール インスタンスに配布します。
- 監査ログ、障害、イベント ログなどの診断情報を保持します。

Policy Manager サービス タイプは policy-mgr です。Policy Manager に関連するすべての要求に対して API URL でこのサービス タイプを使用します。

VM Manager

VM Manager は、VMWare vCenter と対話し、vCenter から取得された VM 情報を保持するために使用します。VM Manager は、ユーザがアクセスできるサービスがないバックエンド サービスです。VM Manager サービス タイプは vm-mgr です。

VNMC XML API

VNMC XML API は、VNMC との統合または対話をプログラミングによって実行できます。API インターフェイスは、HTTPS プロトコルを使用して XML ドキュメントを受け取ります。開発者は、任意のプログラミング言語を使用して API メソッドを含む XML ドキュメントを生成できます。設定およびステータス情報は、管理情報ツリー（XML API を介して完全に公開されます）という階層ツリー構造に格納されます。

API モデルは、再帰的に駆動され、アプリケーション開発のために主要な機能を提供します。たとえば、変更は、単一のオブジェクト、オブジェクトのサブツリー、またはオブジェクト ツリー全体に対して行うことができます。また、変更は、オブジェクトの単一の属性に行ったり、単一の API コールで VNMC 構造全体に適用したりできます。

API は*寛容*モードで動作します。不明な属性は、DME で保持されているデフォルト値（該当する場合）で置き換えられます。複数の管理対象オブジェクト（つまり、ポリシー）を設定するときにその管理対象オブジェクトのいずれかを設定できない場合、API はその動作を停止し、設定を元の状態に戻し、API プロセスを停止して障害を通知します。

API は、スケーラビリティとパフォーマンスを向上させるために非同期動作モデルを使用します。完了するのに時間がかかるプロセスは*ノンブロッキング*または*非同期*です（高速な API プロセスはブロックされないか、さらに時間がかかることによって遅れています）。プロセスは、有効な要求時に成功メッセージを受け取り、タスクが完了すると完了メッセージを受け取ります。

完全なイベント サブスクリプションがサポートされます。VNMC は、発生したイベント（管理対象オブジェクトの変更など）に関する通知をすべてのサブスクライバに送信し、ステータス変更のタイプを示します。

管理対象オブジェクト データ モデルの今後の更新では、後方互換性を維持するために既存のオブジェクト モデルに準拠します。製品アップグレード時に既存のプロパティが変更された場合、変更はアップグレード後のデータベース ロード中に処理されます。新しいプロパティにはデフォルト値が割り当てられます。

VNMC では、モデル方式アーキテクチャを使用しています。このアーキテクチャでは、変更は最初に管理対象オブジェクトの形式の論理構造に適用されます。次に、管理対象オブジェクトは、エンドポイントに変更を適用し、必要な状態（つまり、設定）のエンドポイントを実現します。

API の動作はトランザクション型で、単一のデータ モデルで終了します。VNMC は、すべてのエンドポイント（ASA 1000V、VSG、および VSM）通信（ステータス更新など）を行います。API ユーザはエンドポイントと直接通信できないため、ユーザは分離された個々のコンポーネント設定を管理する必要がありません。

VNMC API モデルには、次のプログラミング エンティティが含まれます。

- クラス：管理情報ツリーのオブジェクト、プロパティおよびステータス
- メソッド：1 つまたは複数のオブジェクトに対して API が実行するアクション
- タイプ：オブジェクトのプロパティの許容値の収集または範囲。一般的な要求はサービス コンポーネントの DME に送信され、トランザクタ キューにファーストイン ファーストアウト（FIFO）の順序で配置されます。トランザクタはこのキューから要求を取得し、要求を解釈して認可チェックを実行します。要求が確認されると、トランザクタは管理情報ツリーを更新します。このプロセスは、単一のトランザクションで実行されます。

VNMC のデータ モデル スキーマ

すべての VNMC サービス プロバイダーに対するデータ モデル HTML ドキュメントは VNMC サーバでパッケージ化され、次の URL のいずれかを使用してアクセスできます。

- https://vnmc-ip-address/doc、次の内容が含まれます。
  - core
  - policy-mgr
  - resource-mgr
  - service-reg
- https://vnmc-ip-address/schema、次の内容が含まれます。
  - core.in.xsd：Management Controller 設定 API とデータ モデル

- core.out.xsd：Management Controller クエリー API とデータ モデル
- policy-mgr.in.xsd：Policy Manager 設定 API とデータ モデル
- policy-mgr.out.xsd：Policy Manager クエリー API とデータ モデル
- resource-mgr.in.xsd：Resource Manager 設定 API とデータ モデル
- resource-mgr.out.xsd：Resource Manager クエリー API とデータ モデル
- service-reg.in.xsd：Service Registry 設定 API とデータ モデル
- service-reg.out.xsd：Service Registry クエリー API とデータ モデル

VNMC サービスへのアクセス

VNMC サービスには、HTTPS プロトコルを介した XML API 要求を使用してアクセスできます。HTTPS 要求の URL 形式は次のとおりです。

`https://vnmc-ip-address/xmlIM/service-type`

service-type は、「VNMC のコンポーネント」で説明されている、該当するサービス プロバイダーのタイプです。たとえば、Policy Manager に要求を送信するには、次の例に表示されているサービス タイプ policy-mgr を使用します。

`https://vnmc-ip-address/xmlIM/policy-mgr`

オブジェクトの命名

特定のオブジェクトは、次の項で説明されるように、識別名（DN）または相対名（RN）で識別できます。

- 識別名
- 相対名

識別名

DN を使用すると、ターゲット オブジェクトを明確に識別できます。DN は、一連の相対名から構成される次の形式を持ちます。

dn = rn/rn/rn/rn...

たとえば、一連の相対的な名前は次の例と類似している可能性があります。

org-root/org-tenant1/zone-trustedServers-0

次の例では、DN は、オブジェクト ツリーの最上部から Zone（GUI の vZone）オブジェクト（zone-trustedServers-0）までの完全修飾パスを提供しています。DN は、API コールが動作する管理対象オブジェクトを指定します。

< …dn = "org-root/org-tenant1/zone-trustedServers-0" />

相対名

RN は、親オブジェクトのコンテキスト内でオブジェクトを識別します。オブジェクトの DN は、次の形式で親 DN と RN で構成されます。

dn = parent-dn/rn

たとえば、テナント Tenant1 の下にある名前 trustedServers-0 の Zone（GUI の vZone）オブジェクトには、次の関連する RN および DN があります。

- オブジェクトの RN は、zone-trustedServers-0 です。
- オブジェクトの親テナント DN は、org-root/org-Tenant1 です。
- オブジェクトの DN は、org-root/org-Tenant1/zone-trustedServers-0 です。

API メソッド カテゴリ

VNMC との対話に 4 つのメソッド カテゴリが使用されます。各 API はメソッドであり、各メソッドは XML ドキュメントに対応します。ここでは、メソッド カテゴリについて詳細に説明し、また必要な情報を取得するためにクエリー フィルタを使用する方法について説明します。

- 認証方法

- クエリー メソッド
- クエリー フィルタ
- 設定メソッド
- イベント サブスクリプション メソッド

（注） このマニュアルのいくつかのコード例では、用語 <real_cookie> は 1217377205/85f7ff49-e4ec-42fc-9437-da77a1a2c4bf などの実際の Cookie に置き換えられます。VNMC の Cookie は 47 文字の文字列です。これは、Web ブラウザがセッション情報を保持するためにローカルに保存する種類の Cookie ではありません。

認証方法

認証方法は、VNMC アクティブ セッションを開始し、維持します。他の API コールが許可される前に、正常な認証を実行する必要があります。API 要求は Cookie により認証されます。

接続セッションが確立および認証されると、応答で Cookie が返されます。これは 7200 秒（120 分）有効です。Cookie は、有効期限が切れないようセッション期間中に更新する必要があります。各更新操作により、デフォルトの期間有効な Cookie が作成されます。

VNMC に対して一度に最大 256 個のセッションを開くことができます。最大セッション制限に到達すると、後続するログイン要求が拒否されます。

操作は HTTP POST 方式を使用して実行されます。VNMC は、ポート 443 で HTTPS プロトコルだけをサポートします。HTTP のエンベロープには XML の設定が含まれます。

表 1 に使用可能な認証方式を示します。

表 1. 使用可能な認証方式

方式	説明
aaaLogin	VNMC にログインする最初の方式です。 接続セッションを確立し、有効なクッキーを取得します。
aaaRefresh	セッションを維持し、現在の認証 Cookie をリフレッシュします。
aaaLogout	現在のセッションを終了し、現在の認証 Cookie を非アクティブにします。

クエリー メソッド

クエリー メソッドは、VNMC オブジェクトの現在の設定ステータスに関する情報を取得します。

ほとんどのクエリー メソッドは、引数 inHierarchical、ブール値 true/yes または false/no を持ちます。true の場合、inHierarchical 引数はすべての子オブジェクトを返します。次に例を示します。

<configResolveDn … inHierarchical="false"></> <configResolveDn … inHierarchical="true"></>

また、API クエリー メソッドは、コールを再帰的にするかどうか（つまり、他のオブジェクトまたは親オブジェクトを参照し返すオブジェクトに従うかどうか）を指定する inRecursive 引数を持つ場合があります。

表 2 に使用可能なクエリー メソッドを示します。

表 2. クエリー メソッド

フィルタ	説明
configResolveDns	DN のセットによりオブジェクトを取得します。
configResolveDns	複数のクラスのオブジェクトを取得します。
configResolveChildren	オブジェクトの子オブジェクトを取得します。
configResolveDns	DN のセットによりオブジェクトを取得します。
configResolveDns	複数のクラスのオブジェクトを取得します。
configResolveChildren	オブジェクトの子オブジェクトを取得します。
configResolveParent	オブジェクトの親オブジェクトを取得します。
configScope	管理情報ツリーの DN に対してクラス クエリーを実行します。

クエリー フィルタ

この API には、クエリー メソッドの有用性を高めるためのフィルタが含まれます。これらのフィルタは、クエリーの一部として渡すことができ、必要な結果セットを特定するために使用されます。ここでは、使用可能なフィルタについて説明します。

- Simple フィルタ
- Property フィルタ
- Composite フィルタ
- Modifier フィルタ

Simple フィルタ

Simple フィルタには true と false の 2 つがあります。これらのフィルタは、次のようにそれぞれ true または false の単純なステータスに反応します。

- true フィルタ：true ブール条件を持つオブジェクトの結果セット。
- false フィルタ：false ブール条件を持つオブジェクトの結果セット。

詳細については、「Simple フィルタ」を参照してください。

Property フィルタ

Property フィルタは、結果セットに含める基準としてオブジェクトのプロパティの値を使用します。ほとんどの場合、Property フィルタを作成するには、比較の値とともにターゲット オブジェクトとプロパティの classId と propertyId が必要です。

表 3 に、Property フィルタのタイプについて説明します。

表 3. Property フィルタ

フィルタ	説明
Equality	結果セットは、特定のプロパティが、指定したプロパティ値と等しいオブジェクトを含みます。
Not equal	結果セットは、特定のプロパティが、指定したプロパティ値と等しくないオブジェクトを含みます。
Greater than	結果セットは、特定のプロパティが、指定したプロパティ値よりも大きいオブジェクトを含みます。
Greater than or equal	結果セットは、特定のプロパティが、指定したプロパティ値以上であるオブジェクトを含みます。
Less than	結果セットは、特定のプロパティが、指定したプロパティ値未満であるオブジェクトを含みます。
Less than or equal	結果セットは、特定のプロパティが、指定したプロパティ値以下であるオブジェクトを含みます。
Wildcard	結果セットは、特定のプロパティがワイルドカード基準を満たすオブジェクトを含みます。サポートされるワイルドカードは、次のとおりです。 % : 単一文字に一致するパーセント記号。 * : 0 文字以上に一致するアスタリスク。 ? : 0 文字以上に一致する疑問符。 - : 一致する範囲を示すダッシュ。たとえば、[a-e] は a ～ e の任意の文字に一致します。 + : 1 文字以上に一致するプラス記号。
Any bits	結果セットは、特定のプロパティが、渡されたビット セットの少なくとも 1 つを持つオブジェクトを含みます。（ビットマスク プロパティにだけ使用します）。
All bits	結果セットは、特定のプロパティが、渡されたビット セットのすべてを持つオブジェクトを含みます。（ビットマスク プロパティにだけ使用します）。

詳細については、「Property フィルタ」を参照してください。

Composite フィルタ

Composite フィルタは、2 つ以上のコンポーネント フィルタから構成され、柔軟に結果セットを作成できます。たとえば、Composite フィルタによって、含まれているフィルタの少なくとも 1 つで受け入れられたオブジェクトだけに結果セットを限定できます。

表 4 に使用可能な Composite フィルタを示します。

表 4. Composite フィルタ

フィルタ	説明
AND	結果セットは、各コンポーネント フィルタのフィルタリング基準を満たす必要があります。たとえば、totalMemory が 64 メガバイトを超えた動作可能なすべてのコンピュータ ブレードを取得する場合、フィルタは 1 つの Greater than フィルタと 1 つの Equality フィルタから構成されます。
Between	結果セットは、最初の指定値と 2 番めの指定値の範囲内にあるオブジェクトを含みます。たとえば、障害などがあります。
OR	結果セットは、少なくとも 1 つのコンポーネント フィルタのフィルタリング基準を満たす必要があります。たとえば、2 つの Equality フィルタで構成されるフィルタを持つすべてのサービス プロファイルを取得します。
XOR	結果セットは、たった 1 つの Composite コンポーネント フィルタのフィルタリング基準を満たすオブジェクトです。

詳細については、「Composite フィルタ」を参照してください。

Modifier フィルタ

Modifier フィルタは、含まれているフィルタの結果を変更します。NOT Modifier フィルタだけがサポートされています。このフィルタは、含まれているフィルタの結果を逆にします。含まれている基準に一致しないオブジェクトを取得する場合は、このフィルタを使用します。

詳細については、「Modifier フィルタ」を参照してください。

設定メソッド

複数のメソッドでは、管理対象オブジェクトの設定を変更できます。使用する設定メソッドに応じて、これらの変更は、ツリー全体、サブツリー、または個々のオブジェクトに適用できます。

設定メソッドの例には、次の内容が含まれます。

- configConfMo：単一のサブツリー（つまり、DN）に影響を与えます。
- configConfMos：複数のサブツリー（つまり、複数の DN）に影響を与えます。
- configConfMoGroup：複数のサブツリー構造または管理対象オブジェクトに対して同じ設定の変更を加えます。

ほとんどの設定メソッドは、引数 inHierarchical を使用します。子オブジェクトが XML ドキュメントに含まれ、DME が寛容モードで動作するため、設定時にこれらの値は重大な役割を担いません。

データの検証

VNMC は、データ型（整数、ブール値、または文字列など）および他の制約（範囲または正規表現など）を含む各プロパティの API によって入力されているすべての設定情報を検証します。API によって提供される情報が指定されたプロパティの要件を満たしていない場合、VNMC はトランザクションが停止し、失敗の応答を発行します。

失敗の応答に関する詳細情報については、「成功または失敗の応答」を参照してください。

イベント サブスクリプション メソッド

ユーザまたはシステムにより開始されたアクションによって、オブジェクトが作成、変更、または削除されると、イベントが生成されます。アプリケーションは、定期的なポーリングまたはイベントに登録することによ

り、VNMC のステータス変更情報を受信できます。ポーリングはリソースを大量に消費するため、イベント サブスクリプションが推奨される通知方法です。

イベント サブスクリプションでは、クライアント アプリケーションが VNMC からのイベント通知を受けるように登録できます。イベントが発生すると、VNMC はイベントおよびそのタイプの通知をサブスクライブしているクライアント アプリケーションに送信します。実際の変更イベントだけが送信され、影響を受けないオブジェクトの属性は送信されません。このプロセスは、システム内のすべてのオブジェクトの変更に適用されます。

イベント通知への登録

イベント通知を登録するには、次の手順を実行します。

ステップ 1. イベント サブスクリプションの有効な Cookie を取得するために、VNMC にログインします。ログインしていない場合、イベント サブスクリプション要求は拒否され、エラー応答が発生します。

ステップ 2. HTTP セッションを開き、このセッションを開いたままにします。

ステップ 3. 次のように、HTTP セッションを介して eventSubcribe 要求を送信します。

```
<eventSubscribe cookie="<real_cookie>"></eventSubscribe>
```

eventSubscribe は、次の URL に転送される必要があります。

```
https://<VNMCIP>/xmlIM/<DMEModule>
```

クライアントがイベント通知を要求するモジュールに応じて、DME モジュールは resource-mgr、policy-mgr、service-reg、または core のいずれかになる可能性があります。eventSubscribe 要求が VNMC により受け取られると、VNMC は HTTP セッションを介して発生するすべての新しいイベントを送信します。

各イベントはイベントの固有識別子（ID）を持ちます。イベント ID はカウンタとして動作し、すべてのイベント通知に含まれます。イベントが生成されると、イベント ID カウンタが増加し、新しいイベントに新しいイベント ID が割り当てられます。このプロセスにより、イベントを追跡することが可能になり、イベントを見逃すことがなくなります。クライアントがイベントを見逃した場合は、loggingSyncOcns を使用して、見逃したイベントを取得できます。

次の方法の非同期イベントの通知は、他の方法と同じ形式を使用しません。代わりに、これらの方法のイベント通知は、例に示す形式と構文を使用します。

configMoChangeEvent

```
<xs:element name="configMoChangeEvent" type="configMoChangeEvent"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configMoChangeEvent" mixed="true">
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

methodVessel

```
<xs:element name="methodVessel" type="methodVessel"
substitutionGroup="externalMethod"/>
   <xs:complexType name="methodVessel" mixed="true">
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignednInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

次に、VNMC がクライアントにイベント通知を送信する例を示します。configMoChangeEvent は、methodVessel 内でラップされます。これは、イベント通知専用で使用されます。

```
<methodVessel cookie="<real_cookie>"
```

```
    commCookie=""
    srcExtSys="172.21.63.106"
    destExtSys="0.0.0.0"
    srcSvc="resource-mgr_dme"
    destSvc="sam_extXMLApi">
        <inStimuli>
            <configMoChangeEvent
                cookie=""
                commCookie=""
                srcExtSys="0.0.0.0"
                destExtSys="0.0.0.0"
                srcSvc="resource-mgr_dme"
                destSvc="sam_extXMLApi"
                inEid="52269">
                <inConfig>
                    <vmInst
                    dn="vmmEp/inst-5008b230-da65-54fb-76d4-8e1a462492cf"
                    powerState="off"
                    status="modified"/>
                </inConfig>
            </configMoChangeEvent>
      </inStimuli>
</methodVessel>
```

eventSendHeartbeat

VNMC は、イベント サブスクリプション イベントに応答または通知を送信しません。代わりに、ソケット接続が失われないようにするために、クライアントに eventHeartBeat（デフォルトは 120 秒）を定期的に送信します。クライアントがこれらのハート ビートを定期的に受信する場合、確実に VNMC のイベントを欠落させないことができます。VNMC がクライアントに送信する eventHeartBeat の例をここに示します。

```
<eventSendHeartbeat cookie="0/0/0/2a76"
       commCookie=""
       srcExtSys="0.0.0.0"
       destExtSys="0.0.0.0"
       srcSvc=""
       destSvc=""
       response="yes"
       outSystemTime="2010-11-12T20:38:19.630">
</eventSendHeartbeat>
```

UML ダイアグラム

統一モデリング言語（UML）ダイアグラムは、VNMC に使用するオブジェクト指向開発手法を補完します。これらのダイアグラムは、クラス、クラス間の関連のほか、属性とメソッドを表示するので、VNMC クラスの全体像およびクラス間がどのように相互に関連しているかを把握できます。

VNMC UML ダイアグラムは、以降のトピックで説明するように、次の属性を含みます。

- クラス
- 関連
- 継承
- 集約および合成

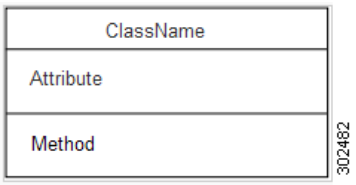
サンプルの VNMC UML ダイアグラムを表示するには、「UML の図」を参照してください。

クラス

UML では、*クラス*はオブジェクトを抽象化したものです。VNMC では、オブジェクトの例であるクラスは、テナント、コンピュート ファイアウォール、エッジ ファイアウォール、ポリシー、およびプロファイルとなります。一般的に、VNMC で作成できるものはクラスとみなすことができます。

クラスは UML の主要コンポーネントで、図 1 に表示されるように、UML ダイアグラムで四角形で表示されます。

図 1　UML のクラス表記

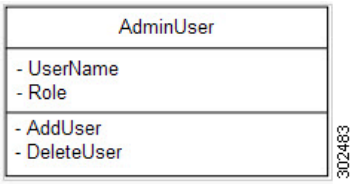


クラス表記には、次の 3 つのセクションが含まれます。

- クラス名：オブジェクトの名前。
- 属性：オブジェクトに関して保存される情報。
- メソッド：オブジェクトまたはクラスが実行できる操作。

図 2 は、各セクションでエントリの例でクラスの例を示します。

図 2　UML のクラス例



クラス ダイアグラムのすべてがすべてのセクションの内容を含むとは限りません。たとえば、メソッドがなくクラス名と属性を持つ UML クラス ダイアグラムが表示されることがあります。この場合、3 つのセクションが表示されますが、メソッドのセクションには情報が含まれません。

関連

UML ダイアグラムの関連は、2 つ以上のクラス間のリンクまたは依存関係を示します。関連は図 3 に示すようにさまざまな方法で表され、表 5 で説明します。

図 3UML ダイアグラムの関連

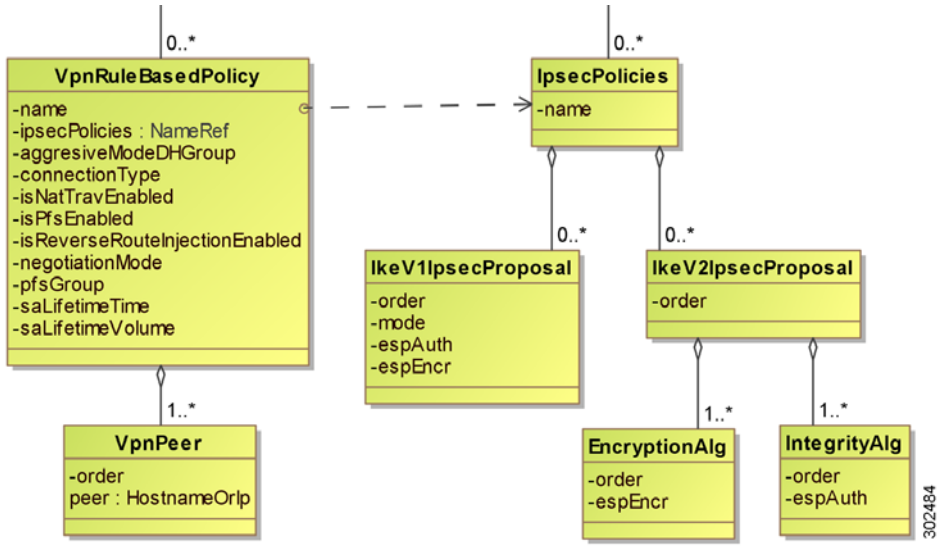


表 5.　UML 関連の表記法

表記法	説明
線	
実線	• 閉じて塗りつぶされていない矢印を使用する場合、上位のクラスのオブジェクト（またはスーパークラス）を指す矢印で継承を示します。 • 開いた矢印を使用する場合、既知のクラスの関連を示します。
破線	参照オブジェクトから発信する矢印で名前付きの参照先を示します。

矢印	
閉じて塗りつぶされていない矢印	実線を使用する場合、スーパークラスを指す矢印で継承を示します。
塗りつぶされた矢印	入れ子になった制御フローまたは手続呼出しを示します。
開いた矢印	実線を使用する場合、スーパークラスを指す矢印で単方向の関連を示します。
1 本の矢印	• 実線を使用する場合、スーパークラスを指す矢印で継承を示します。 • 破線を使用する場合、矢印の方向で単方向の関連を示します。
2 本の矢印または矢印なし	実線で使用した場合、双方向の関連を示します。
ダイヤモンド	
空のダイヤモンド	親の末尾にある空のダイヤモンドでオブジェクト間の基本的な集約を示します。
塗りつぶされたダイヤモンド	親の末尾にある塗りつぶされたダイヤモンドでオブジェクト間の合成集約を示します。
多重性の指定 接続されたオブジェクトに関連付けることのできるオブジェクトのインスタンスの数を示します。	
0..1	0 または 1。
1	必ず 1。
0..*	0 以上。
1..*	1 以上。
n	必ず n（n > 1）。
0..n	必ず n（n > 1）。
1..n	必ず n（n > 1）。

継承

類似していたり、密接に関連しているクラスは、多くの場合、同じ属性またはメソッドを使用します。複数のクラスの反復的なコーディングを防ぐために、UML は既存の属性とメソッドを再利用できる継承メカニズムを採用しています。

継承の状況では、情報を継承するクラスはサブクラスと呼ばれ、継承されるクラスはスーパークラスと呼ばれます。1 つのクラスのすべての属性とメソッドが別のクラスによって継承される状況は、純粋な継承と呼ばれます。

UML ダイアグラムでは、継承はサブクラスからスーパークラスを指す閉じた矢印の実線で示されます。

集約および合成

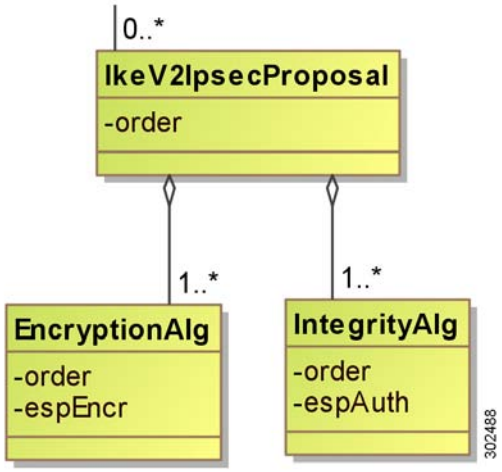
集約および合成には、固有の特性を持つ関連のタイプがあります。UML ダイアグラムを使用する場合、集約と合成の違いを理解して、どのように図に表示されているかを理解することが重要です。

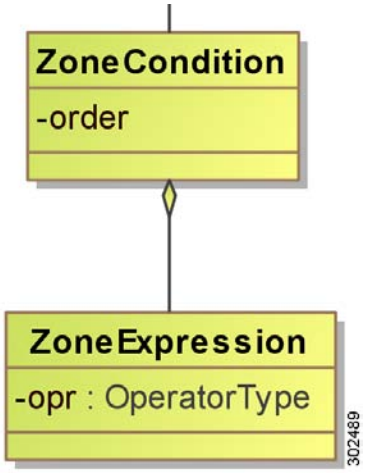
表 6 で、集約と合成の関連の違いを説明します。

表 6. 集約および合成の特性

集約	合成
集約オブジェクトは、コンポーネント オブジェクトの存在には影響しません。 つまり、集約オブジェクトがシステムから削除されると、コンポーネント オブジェクトはそのまま存在します。	集約オブジェクトは、コンポーネント オブジェクトの存在に影響します。 つまり、集約オブジェクトがシステムから削除されると、すべてのコンポーネント オブジェクトも削除されます。
コンポーネント オブジェクトは、システムの他のクラスから集約できます。	コンポーネント オブジェクトは、集約（または親）オブジェクト以外のオブジェクトでは使用できません。
コンポーネント オブジェクトの変更内容は、残りのシステムに伝播させることができます。	コンポーネント オブジェクトの変更内容は、残りのシステムに伝播させることができません。

UML ダイアグラムでは、集約は親の末尾にある空のダイアモンドで実線によって表されます。	UML ダイアグラムでは、合成は親の末尾にある塗りつぶされたダイアモンドで実線によって表されます。





VNMC GUI とサーバ間の XML 交換の取得

VNMC GUI は、Adobe Flex GUI フレームワークを使用して開発された Web ベースのアプリケーションです。その結果、Adobe Flash Player のデバッグ バージョンをインストールして GUI と VNMC サーバ間で XML 交換を取得できます。Adobe Flash Player のデバッグ バージョンのインストール後、ログ出力がホーム ディレクトリの下のログ ファイルに保存されます。（Windows 7 では、C:\Users*username*\AppData\Roaming\Macromedia\Flash Player\Logs\flashlog.txt の下にログ ファイルがあります）。

「VNMC XML API メソッド」の例のセクションに含まれるサンプル要求と応答のほとんどの内容がこの方法で取得されます。

成功または失敗の応答

VNMC は、どのような API 要求に対してもほとんど即時に応答します。要求を完了できない場合は、失敗が返されます。クエリーまたはログイン メソッドの場合は、実際の結果が返されます。設定メソッドの場合、成功の応答は要求が有効であることを示しますが、操作が完了したことを示しません。たとえば、バックアップ要求が受け入れられ、VNMC から成功の応答があった場合でも、VNMC サーバで実際のバックアップ ジョブを完了するのに長い時間がかかることがあります。

ここでは、応答タイプを詳細に説明します。

- 成功の応答
- 失敗の応答
- 空の結果の応答

成功の応答

成功の応答時、XML ドキュメントが、要求された情報または変更が行われたことを示す確認とともに返されます。

次に、DN org-root/org-tenant d3337/pol-pl を持つポリシーに対する configResolveDn 要求の例を示します。

```
<configResolveDn cookie="<real_cookie>"
     dn="org-root/org-tenant_d3337/pol-p1"
     inHierarchical="false"/>
```

この応答には次の詳細が含まれます。

```
<configResolveDn
    dn="org-root/org-tenant_d3337/pol-p1"
    cookie="<real_cookie>"
    commCookie="7/13/0/a3c"
```

```
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfig>
        <policyRuleBasedPolicy
            descr=""
            dn="org-root/org-tenant_d3337/pol-p1"
            intId="10811"
            name="p1"/>
    </outConfig>
</configResolveDn></configResolveDn>
```

失敗の応答

失敗した要求に対する応答には、errorCode と errorDescr に対する XML 属性が含まれます。次に、システム内にすでに存在する別のポリシーと同じ名前のポリシーを作成しようとした失敗した要求に対する応答の例を示します。

```
<configConfMo
    dn="org-root/org-tenant_d3337/pol-p1"
    cookie="<real_cookie>"
    commCookie="7/13/0/2038"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes"
    errorCode="103"
    invocationResult="unidentified-fail"
    errorDescr="can't create; object already exists.">
</configConfMo>
```

特定の種類の解析エラーでは、記載されたプロパティが無効または解析不能の場合、応答はエラー XML です。次の例は、Wildcard フィルタに使用された無効な正規表現の応答を示します。

```
<error cookie="<real_cookie>"
    response="yes"
    errorCode="ERR-xml-parse-error"
    invocationResult="594"
errorDescr="XML PARSING ERROR: Invalid regular expression pattern"/>
```

空の結果の応答

存在しないオブジェクトに対するクエリー要求は、失敗として扱われません。オブジェクトが存在しない場合は、成功メッセージが返されますが、XML ドキュメントには、要求されたオブジェクトが見つからないことを示す空のデータ フィールド <outConfig> </outConfig> が含まれます。次に、DN による存在しないポリシーの解決の例を示します。

```
<configResolveDn
    dn="org-root/org-tenant_d3337/pol-p1"
    cookie="<real_cookie>"
    commCookie="7/13/0/203e"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfig>
    </outConfig>
</configResolveDn>
```

共通の API メソッドおよび規制

ここでは、共通の VNMC API メソッドについて特定し、API 規則について説明し、要求と応答の例を示します。

- 方式およびフィルタ
- API の例

方式およびフィルタ

次の各トピックでは、VNMC が使用する方式およびフィルタについて説明します。

- 認証方法
- 情報収集用クエリー メソッド
- ポリシーに関するクエリー メソッド
- 障害に関するクエリー メソッド
- フィルタ

認証方法

認証により、API は VNMC と対話できるようになります。また、認証を使用すると、権限を設定し、実行できる操作を制御できます。

（注） セッション Cookie は 47 文字の文字列であり、Web ブラウザがセッション情報を保持するためにローカルに保存する種類の Cookie ではありません。ほとんどのコード例では、<real_cookie> が 1217377205/85f7ff49-e4ec-42fc-9437-da77a1a2c4bf などの実際の Cookie の代わりに使用されています。

次の各トピックでは、認証方式について詳細に説明します。

- ログイン
- セッションの更新
- セッションからのログアウト
- 失敗したログインに対する応答

ログイン

次のコード サンプルは、HTTPS を使用して VNMC に接続するためのログイン要求および認証要求を投稿する Linux POST コマンドを示します。

```
POST https://10.193.34.70/xmlIM/mgmt-controller
Please enter content (application/x-www-form-urlencoded) to be POSTed:
<aaaLogin
    inName="admin"
    inPassword="Company@123"/>
```

（注） XML バージョンと DOCTYPES は、aaaLogin に含まれていません。これらを使用しないでください。inName 属性と inPassword 属性はパラメータです。

各 XML API ドキュメントは、実行する操作を表します。要求が XML API ドキュメントとして受け取られると、VNMC は要求を読み取り、メソッドで指定されているアクションを実行します。VNMC は、XML ドキュメント形式のメッセージで応答し、要求の成否を示します。

次のコード サンプルは、ログイン要求の一般的な成功の応答を表しています。

```
<aaaLogin
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin,read-only"
    outDomains="mgmt02-dummy"
    outChannel="fullssl"
    outEvtChannel="fullssl">
```

```
</aaaLogin>
```

（注） VNMC イベント チャネルは HTTP が使用されるため、SSL で暗号化または送信されません。

セッションの更新

セッションは、aaaLogin 応答または以前の更新から取得された 47 文字の Cookie を使用して、aaaRefresh メソッドで更新されます。

次のコード サンプルは、セッションをリフレッシュする方法を表しています。

```
<aaaRefresh
    inName="admin"
    inPassword="mypasword"
    inCookie="<real_cookie>"/>
```

セッションからのログアウト

次のコード サンプルは、セッションからログアウトする方法を表しています。

```
<aaaLogout
    inCookie="<real_cookie>"/>
```

失敗したログインに対する応答

次のコード サンプルは、失敗したログインの応答を表しています。

```
<aaaLogin
    response="yes"
    cookie=""
    errorCode="551"
    invocationResult="unidentified-fail"
    errorDescr="Authentication failed">
</aaaLogin>
```

情報収集用クエリー メソッド

クエリー メソッドは、階層、ステータス、およびスコープを含む情報を取得します。

ここでは、使用可能なクエリー メソッドについて説明します。

- configResolveDn
- configResolveDns
- configResolveClass
- configResolveClasses
- configFindDnsByClassId
- configResolveChildren
- configResolveParent
- configScope
- configResolveDn

configResolveDn

configResolveDn を設定する場合は、次の点に注意してください。

- DN により指定されたオブジェクトが取得されている。
- 指定された DN が、解決するオブジェクト インスタンスを識別している。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。

「configResolveDn」に示す要求または応答の例を参照してください。

configResolveDns

複数の DN を解決するために configResolveDns を使用するときは、次のことに注意してください。

- DN により指定されたオブジェクトが取得されている。
- 指定された DN が、解決するオブジェクト インスタンスを識別している。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。
- 要求の順序によって応答の順序が決まるわけではない。
- 未知の DN が、outUnresolved 属性の一部として返されている。

「configResolveDns」に示す要求または応答の例を参照してください。

configResolveClass

クラスを解決するために configResolveClass を使用するときは、次のことに注意してください。

- 指定されたクラス タイプのオブジェクトが取得されている。
- classId 属性により、返されるオブジェクト クラス名が指定されている。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。列挙値、クラス ID、およびビット マスクが文字列として表示されている。
- 結果セットが大きくなることがある。結果セットは正確に定義してください。たとえば、装置のすべてのインスタンスを取得するには、equipmentItem クラスを問い合わせます。サーバのリストだけを取得する場合は、クエリーで classId の属性値として computeBlade を使用します。

「configResolveClass」に示す要求または応答の例を参照してください。

configResolveClasses

複数のクラスを解決するために configResolveClasses を使用するときは、次のことに注意してください。

- 指定されたクラス タイプのオブジェクトが取得されている。
- classId 属性により、返されるオブジェクト クラス名が指定されている。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。

（注） 無効なクラス名が inIds 属性で指定された場合は、XML 解析エラーが生成されます。クエリーは実行できません。

「configResolveClasses」に示す要求または応答の例を参照してください。

configFindDnsByClassId

指定されたクラスの識別名の検出時に、次のことを確認してください。

- 指定されたクラスの DN が取得されている。
- classId 属性により、取得するオブジェクト タイプが指定されている。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。

「configFindDnsByClassId」に示す要求または応答の例を参照してください。

configResolveChildren

管理情報ツリーの子オブジェクトの解決時に、次のことを確認してください。

- このメソッドが、名前付きクラスのインスタンスである名前付きオブジェクトのすべての子オブジェクトを取得している。

  （注）　クラス名を省略すると、名前付きオブジェクトのすべての子オブジェクトが返されます。

- inDn 属性により、子オブジェクトが取得される名前付きオブジェクトが指定されている。
- classId 属性により、返される子オブジェクト クラスの名前が指定されている。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。

「configResolveChildren」に示す要求または応答の例を参照してください。

configResolveParent

オブジェクトの親オブジェクトの解決時に、次のことを確認してください。

- このメソッドが、指定された DN の親オブジェクトを取得している。
- dn 属性が子オブジェクトの DN である。
- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。

「configResolveParent」に示す要求または応答の例を参照してください。

configScope

クエリーのスコープを制限すると、粒度が細かくリソースを大量に消費しない要求を実現できます。クエリーは、管理情報ツリーでルート以外の場所に固定できます。

クエリー スコープの設定時に、次のことを確認してください。

- メソッドが、指定された DN にクエリーのルート（スコープ）を設定し、指定されたクラス タイプのオブジェクトを返している。
- dn が、クエリーのスコープが設定された名前付きオブジェクトである。
- inClass 属性により、返されるオブジェクト クラスの名前が指定されている。

  （注）　クラス名が指定されない場合、クエリーは configResolveDn と同じように動作します。

- 認証 Cookie を aaaLogin または aaaRefresh から受け取っている。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、クラス ID、およびビット マスクが文字列として表示されている。

「configScope」に示す要求または応答の例を参照してください。

ポリシーに関するクエリー メソッド

ポリシーは、さまざまな組織で利用できます。多くのポリシーがある大規模なシステムでは、同時にすべてのポリシーを問い合わせるとリソースが大量に消費されます。代わりに、ポリシーのタイプと、ポリシーが割り当てられる親組織を指定してください。

次のコード サンプルは、ルールベースのポリシーのクエリーを表しています。

```
<configScope
    cookie="<real_cookie>"
    inHierarchical="false"
    dn="org-root/org-tenant_d3338"
    inClass="policyRuleBasedPolicy" />
```

応答は次のとおりです。

```
<configScope
```

```
    dn="org-root/org-tenant_d3338"
    cookie="<real_cookie>"
    commCookie="7/13/0/24e7"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <policyRuleBasedPolicy
            descr=""
            dn="org-root/org-tenant_d3338/pol-p9"
            intId="10274"
            name="p9"/>
        <policyRuleBasedPolicy
            descr=""
            dn="org-root/org-tenant_d3338/pol-p1"
            intId="10301"
            name="p1"/>
    </outConfigs>
</configScope>
```

DN を使用する次のクエリーはさらに効率的です。

```
<configResolveDn
    inHierarchical="false"
    cookie="<real_cookie>"
    dn="sys org-root/org-tenant_d3338/pol-p1">
</configResolveDn>
```

ポリシー オブジェクトとそのルール データを取得するには、次のように inHierarchical を true に変更します。

```
<configResolveDn
    inHierarchical="true"
    cookie="<real_cookie>"
    dn=" org-root/org-tenant_d3338/pol-p1">
</configResolveDn>
```

障害に関するクエリー メソッド

次のコード サンプルは、障害のクエリーを表しています。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="faultInst"/>
```

次に、すべての重大な障害（メジャーおよびクリティカル）のリストを取得する例を示します（フィルタ <inFilter>eq class= "faultInst" を含む）。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="faultInst">
    <inFilter>
        <eq class="faultInst"
            property="highestSeverity"
            value="major" />
    </inFilter>
</configResolveClass>
```

フィルタ

目的の特定のクエリーを作成するには、次の各トピックで説明されているようにフィルタを使用します。

- Simple フィルタ
- Property フィルタ
- Composite フィルタ
- Modifier フィルタ

Simple フィルタ

次の各トピックで説明されているように、Simple フィルタは true および false 条件を使用します。

- false 条件
- true 条件

inHierarchical（次の例に表示されます）に加えて、inRecursive および inSingleLevel を true または false に設定することもできます。

false 条件

次に、false 条件を使用した Simple フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    classId="topSystem"
    inHierarchical="false">
    <inFilter>
    </inFilter>
</configResolveClass>
```

true 条件

次に、true 条件を使用した Simple フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    classId="topSystem"
    inHierarchical="true">
    <inFilter>
    </inFilter>
</configResolveClass>
```

Property フィルタ

ここでは、Property フィルタを使用する方法について説明します。

- Equality フィルタ
- Not equal フィルタ
- Greater than フィルタ
- Greater than or Equal to フィルタ
- Less than フィルタ
- Less Than or Equal to フィルタ
- Wildcard フィルタ
- Any Bits フィルタ
- All Bits フィルタ

Equality フィルタ

次に、*Equality* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="fwComputeFirewall">
    <inFilter>
        <eq class="fwComputeFirewall"
```

```
            property="assocState"
            value="associated" />
    </inFilter>
</configResolveClass>
```

Not equal フィルタ

次に、*Not equal* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="fwComputeFirewall">
    <inFilter>
        <ne class="fwComputeFirewall"
            property="assocState"
            value="associated" />
    </inFilter>
</configResolveClass>
```

Greater than フィルタ

次に、*Greater than* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    classId="eventRecord"
    inHierarchical="false">
    <inFilter>
        <gt class="eventRecord"
            property="txId"
            value="36520" />
    </inFilter>
</configResolveClass>
```

Greater than or Equal to フィルタ

次に、*Greater than or Equal to* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="sysdebugCore">
    <inFilter>
        <ge class="sysdebugCore"
            property="size"
            value="2097152" />
    </inFilter>
</configResolveClass>
```

Less than フィルタ

次に、*Less than* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="sysdebugCore">
    <inFilter>
        <lt class="sysdebugCore"
            property="size"\
            value="2097152" />
    </inFilter>
</configResolveClass>
```

Less Than or Equal to フィルタ

次に、*Less Than or Equal to* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="sysdebugCore">
    <inFilter>
        <le class="sysdebugCore"
            property="size"
            value="2097152" />
    </inFilter>
</configResolveClass>
```

Wildcard フィルタ

次に、名前がプレフィックス「edge」で始まるすべての仮想ファイアウォールを検索する *Wildcard* フィルタの例を示します。

要求：

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="fwInstance">
    <inFilter>
        <wcard class="fwInstance"
            property="name"
            value="edge*" />
    </inFilter>
</configResolveClass>
```

応答：

```
<configResolveClass
    cookie="<real_cookie>"
    commCookie="5/12/0/14dc"
    srcExtSys=""
    destExtSys=""
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes"
    classId="fwInstance">
    <outConfigs>
        <fwInstance
        assignedToDn="org-root/org-Tenant-1/efw-NewSampleEDgeFirewall"
        assoc="associated"
        capability="infra-fw"
        descr=""
        dn="fw/inst-1007"
        fltAggr="0"
        fsmDescr=""
        fsmPrev="AssociateSuccess"
        fsmProgr="100"
        fsmRmtInvErrCode="none"
        fsmRmtInvErrDescr=""
        fsmRmtInvRslt=""
        fsmStageDescr=""
        fsmStamp="2013-05-21T23:27:57.016"
        fsmStatus="nop"
        fsmTry="0"
        intId="10259"
        mgmtIp="<FirewallIP>"
        model=""
        name="edge-firewall"
        pooled="0"
        registeredClientDn="extpol/reg/clients/client-1007"
```

```
        revision="0"
        serial=""
        svcId="1007"
        vendor=""/>
    </outConfigs>
</configResolveClass>
```

Wildcard フィルタの場合、ユーザが無効なフィルタを指定すると、VNMC はエラー タグで応答を返します。次の例には、無効な Wildcard フィルタではなく、ワイルドカード「*」があります。

要求

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="orgTenant">
    <inFilter>
        <wcard class="orgTenant"
        property="name" value="*" />
    </inFilter>
</configResolveClass>
```

応答

```
<error cookie="<real_cookie>"
    response="yes"
    errorCode="ERR-xml-parse-error"
    invocationResult="594"
errorDescr="XML PARSING ERROR: Invalid regular expression pattern"/>
```

Any Bits フィルタ

次に、*Any Bits* フィルタの例を示します。

```
<configResolveClass
    cookie="null"
    inHierarchical="false"
    classId="extpolClient">
    <inFilter>
        <anybit class="extpolClient"
            property="capability"
            value="vm-fw,vm-vasw" />
    </inFilter>
</configResolveClass>
```

All Bits フィルタ

次に、*All Bits* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="extpolClient">
    <inFilter>
        <allbits class="extpolClient"
            property="capability"
            value="vm-fw,vm-vasw" />
    </inFilter>
</configResolveClass>
```

Composite フィルタ

次の各トピックでは、Composite フィルタについて説明します。

- AND フィルタ
- OR フィルタ

- Between フィルタ
- AND OR NOT Composite フィルタ

AND フィルタ

次に、設定に関連付けられたファイアウォールを検索する *AND* フィルタの例を示します。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="fwComputeFirewall">
    <inFilter>
        <and>
            <eq class="fwComputeFirewall"
                property="assocState"
                value="associated" />
            <eq class="fwComputeFirewall"
                property="configState"
                value="applied" />
        </and>
    </inFilter>
</configResolveClass>
```

OR フィルタ

次に、操作ステータスが可視性なし、または未登録であるすべての管理対象エンドポイントを返す *OR* フィルタの例を示します。

```
<configResolveClass
    inHierarchical="false"
    cookie="<real_cookie>"
    classId="extpolClient">
    <inFilter>
        <or>
            <eq class="extpolClient"
                property="operState"
                value="unregistered"/>
                <eq class="extpolClient"
                property="operState"
                value="lost-visibility"/>
        </or>
    </inFilter>
</configResolveClass>
```

Between フィルタ

次の例は、フィルタ *間*を示しています。

```
<configResolveClass
    cookie="<real_cookie>"
    classId="eventRecord"
    inHierarchical="false">
    <inFilter>
        <bw class="eventRecord"
            property="txId"
            firstValue="4"
            secondValue="5"/>
    </inFilter>
</configResolveClass>
```

AND OR NOT Composite フィルタ

次に、[unassociated] の関連づけの状態、[not-applied] の設定状態を持っているが、[reachability] の補助プロパティ値を持たないタイプ fwComputeFirewall のすべてのオブジェクトを返す *AND OR NOT* Composite フィルタの例を示します。

```
<configScope
    cookie="<real_cookie>"
    dn="org-root"
    inClass="fwComputeFirewall"
    inHierarchical="false"
    inRecursive="false">
    <inFilter>
        <and>
            <or>
                <eq class="fwComputeFirewall" property="assocState"
                value="unassociated" />
                <eq class="fwComputeFirewall" property="configState" value="not-
                applied" />
            </or>
            <not>
                <eq class="fwComputeFirewall" property="auxProps"
                value="reachability"/>
            </not>
        </and>
    </inFilter>
</configScope>
```

Modifier フィルタ

NOT フィルタ

次の例は、NOT Modifier フィルタを示しています。

```
<configResolveClass
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="extpolClient">
    <inFilter>
        <not>
            <anybit class="extpolClient"
                property="capability"
                value="vm-fw" />
        </not>
    </inFilter>
</configResolveClass>
```

API の例

ここでは、API の設定および管理例について説明します。

- Management Controller を使用した認証
- Service Registry を使用したテナント管理
- ポリシー管理
- リソース管理

ここでは、MO（またはオブジェクト）の作成、変更、または削除に configConfMos がどのように使用されているかを説明します。configConfMos API が単一の設定オブジェクトを指定するために単一の XML 要素 <pair> を使用する各例では、同じ結果を得るために configConfMo API を使用できます。

Management Controller を使用した認証

VNMC へのアクセスはセッションベースであり、ログイン要求で最初に認証する必要があります。デフォルトのセッションの長さは 120 分です。aaaKeepAlive メソッドと aaaRefresh メソッドを使用して、セッションを延長できます。アクティビティが完了したときに、aaaLogout を使用してユーザがセッションから手動でログアウトすることが推奨されます。

VNMC にログイン要求を送信する場合は、サービス タイプ mgmt-controller を使用します。

次の各トピックでは、認証要求および応答の例を示します。

- 認証要求
- 認証応答

認証要求

次に、認証要求の例を示します。

```
POST URL: https://10.193.33.221/xmlIM/mgmt-controller
XML API payload:
<aaaLogin
    inName="admin"
    inPassword="Nbv12345"/>
```

認証応答

次に、認証応答の例を示します。

```
<aaaLogin
    cookie=""
    commCookie=""
    srcExtSys="0.0.0.0"
    destExtSys="0.0.0.0"
    srcSvc="" destSvc=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin,read-only"
    outDomains=""
    outChannel="fullssl"
    outEvtChannel="fullssl"
    outSessionId="web_13528"
    outVersion="1.0">
</aaaLogin>
```

Service Registry を使用したテナント管理

Service Registry（サービス タイプが service-reg）は、ポリシーとリソースを整理するために使用されるテナントとサブ組織を作成および管理します。これらのテナントとサブ組織は、ボトムアップ ポリシーおよびリソース解決のためにツリー階層で配置されます。テナントとサブ組織の作成で使用されるクラス名は次の 4 つのレベルをサポートします。

- テナント、クラス orgTenant：最上位の組織を定義します。
- データセンター、クラス orgDatatcenter：テナント下のデータセンターを定義します。
- アプリケーション、クラス orgApp：データセンター下のアプリケーションを定義します。
- 階層、クラス orgTier：アプリケーション下の階層を定義します。

新しい組織を作成したり、既存の組織を更新したりする場合は、以下の情報を指定します。

- オブジェクト DN
- 属性値
- created または modified に等しいステータス
- 組織を削除するには、要求で status = deleted を使用します。

次の各トピックでは、要求および応答を作成または更新する例を示します。

- 組織の作成または更新の要求

- 組織の作成または更新の応答

組織の作成または更新の要求

次に、demoTenant という名前のテナントを作成する例を示します。

```
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/org-demoTenant">
            <orgTenant dn="org-root/org-demoTenant"
                name="demoTenant"
                status="created"/>
        </pair>
    </inConfigs>
</configConfMos>
```

組織の作成または更新の応答

次に、demoTenant という名前のテナントの作成後に受け取る応答の例を示します。

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="2/15/0/839"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="service-reg_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/org-demoTenant">
            <orgTenant
                descr=""\
                dn="org-root/org-demoTenant"
                fltAggr="0"
                level="1"
                name="demoTenant"
                status="created"/>
        </pair>
    </outConfigs>
</configConfMos>
```

ポリシー管理

ここでは、ポリシーを管理する方法について説明します。

- デバイス ポリシー
- デバイス プロファイル
- ゾーン
- オブジェクト グループ
- 属性ディクショナリ
- ポリシー
- PolicySet
- Compute Security Profile

デバイス ポリシー

ここでは、デバイス ポリシーでの作業に関する例を示します。

- syslog ポリシー
- SNMP ポリシー
- LogProfile ポリシー

syslog ポリシー

syslog ポリシー オブジェクトは、システム内で実行されるすべてのアクションを追跡し、デバイス プロファイルのロギング レベルを設定します。次の例では、mysyslog という名前の Syslog ポリシーが作成されます。

要求

```
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/syslog-mysyslog">
            <commSyslog dn="org-root/syslog-mysyslog">
            <commSyslogConsole
                adminState="enabled"
                severity="emergencies"/>
            <commSyslogClient
                name="primary"
                adminState="enabled"
                hostname="5.6.7.8"
                severity="notifications"
                forwardingFacility="local7"/>
            <commSyslogClient
                name="secondary"
                adminState="enabled"
                hostname="123.23.53.123"
                severity="warnings"
                forwardingFacility="local5"/>
            </commSyslog>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/15/0/1b9"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
        <outConfigs>
            <pair key="org-root/syslog-mysyslog">
            <commSyslog
                adminState="enabled"
                descr=""
                dn="org-root/syslog-mysyslog"
                intId="25301"
                name="mysyslog"
                port="514"
                proto="udp"
                severity="warnings"
                status="created"/>
            </pair>
    </outConfigs>
</configConfMos>
```

SNMP ポリシー

次の例では、mysnmp という名前の SNMP ポリシーが作成されます。作成されたポリシーは、デバイス プロファイル リストでこの名前で利用できます。

要求

```
<configConfMos
    cookie="<real_cookie>">
```

```
    <inConfigs>
        <pair key="org-root/snmp-mysnmp">
        <commSnmp dn="org-root/snmp-mysnmp"
            adminState="enabled"
            descr="The default SNMP policy"
            sysContact="Andrew Jackson"
            sysLocation="San Jose" >
        <commSnmpCommunity
            community="public"
            role="read-only"/>
        <commSnmpTrap
            hostname="nms-23.host123.com"
            community="private"/>
        <commSnmpTrap
            hostname="nms-42.host123.com"
            community="private"/>
        </commSnmp>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/15/0/1bc"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/snmp-mysnmp">
        <commSnmp
            adminState="enabled"
            descr="The default SNMP policy"
            dn="org-root/snmp-mysnmp"
            intId="25281"
            name="mysnmp"
            port="161"
            proto="all"
            sysContact="Andrew Jackson"
            sysLocation="San Jose"/>
        </pair>
    </outConfigs>
</configConfMos>
```

LogProfile ポリシー

次の例では、debugLog という名前の LogProfile ポリシーに対してデバッグ レベルとロギング レベルが設定されます。

要求

```
Post URL：https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/logprof-debugLog">
        <policyLogProfile
            dn="org-root/logprof-debugLog"
            name="debugLog"
            level="debug1"
            size="10000000"
            backupCount="4"/>
```

```
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/12/0/17d0"
    srcExtSys="172.20.101.150"
    destExtSys="172.20.101.150"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/logprof-debugLog">
            <policyLogProfile
            adminState="enabled"
            backupCount="4"
            descr=""
            dn="org-root/logprof-debugLog"
            intId="10514"
            level="debug1"
            name="debugLog"
            size="10000000"
            status="created"/>
        </pair>
    </outConfigs>
</configConfMos>
```

デバイス プロファイル

このアクションにより、myDeviceProfile という名前のデバイス プロファイルが作成されます。これにより、プロファイルが有効になり、プロファイルの内容が指定されます。

要求

```
Post URL：https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/org-Cola/fwdevprofile-myDeviceProfile">
        <fwpolicyFirewallDeviceProfile
            dn="org-root/org-Cola/fwdevprofile-myDeviceProfile"
            snmpPolicy="mysnmp"
            syslogPolicy="mysyslog"
            timezone="America/Los_Angeles" >
        <commDns
            name="somedns.com"
            domain="somedns.com"/>
<!-- The order attribute means the device should first use the DNS provider with
thesmallest "order" value. If that DNS server is not responsive, use the DNS
provider with the next smaller number. -->
        <commDnsProvider
            hostip="171.70.168.183"
            order="1"/>
        <commDnsProvider
            hostip="171.68.226.120"
            order="2"/>
        <commDnsProvider
            hostip="64.102.6.247"
            order="3"/>
        <commNtpProvider
```

```
            name="somentp.com"
            order="1"/>
        <commNtpProvider
            name="north-america.pool.ntp.org"
            order="2"/>
        </fwpolicyFirewallDeviceProfile>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/15/0/1ba"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/org-Cola/fwdevprofile-myDeviceProfile">
        <fwpolicyFirewallDeviceProfile
            coreFilePolicy="
            descr=""
            dn="org-root/org-Cola/fwdevprofile-myDeviceProfile"
            dnsPolicy=""
            enablePolicyDecisionLog="no"
            faultPolicy=""
            httpPolicy=""
            httpsPolicy=""
            intId="25326"
            logProfilePolicy=""
            name="myDeviceProfile"
            snmpPolicy="mysnmp"
            status="created"
            syslogPolicy="mysyslog"
            telnetPolicy=""
            timezone="America/Los_Angeles"/>
        </pair>
    </outConfigs>
</configConfMos>
```

ゾーン

次の例では、trustedClients-0 と trustedServers-0 の 2 つのゾーンが作成されます。また、値を持つ属性セットも割り当てられます。

（注） VNMC UI では、ゾーンは vZone（仮想ゾーン）で表します。

要求

```
Post URL：https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/org-tenant1/zone-trustedClients-0">
<!-- Create a zone of all VMs in the 192.168.1.0/24 subnet -->
          <policyZone dn="org-root/org-tenant1/zone-trustedClients-0">
             <policyZoneCondition id="1" order="1">
                <policyZoneExpression opr="prefix">
                   <policyIPAddress id="1" value="192.168.1.0"
                      <policyIPSubnet id="1" value="255.255.255.0" />
                </policyZoneExpression>
```

```
            </policyZoneCondition>
         </policyZone>
      </pair>
      <pair key="org-root/org-tenant1/zone-trustedServers-0">
<!-- Create a zone of all VMs attached to a VNSP where the "appType" is set to
"BuildServer" -->
         <policyZone dn="org-root/org-tenant1/zone-trustedServers-0">
            <policyZoneCondition id="1" order="1">
               <policyZoneExpression opr="eq">
                  <policyParentAppName id="1" value="BuildServer" />
               </policyZoneExpression>
            </policyZoneCondition>
         </policyZone>
      </pair>
   </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
   cookie="<real_cookie>"
   commCookie="7/15/0/1a6"
   srcExtSys="10.193.33.221"
   destExtSys="10.193.33.221"
   srcSvc="sam_extXMLApi"
   destSvc="policy-mgr_dme"
   response="yes">
   <outConfigs>
      <pair key="org-root/org-tenant0/zone-zone0">
         <policyZone
            descr=""
            dn="org-root/org-tenant0/zone-zone0"
            intId="24295"
            name="zone0"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/zone-trustedServers-0">
         <policyZone
            descr=""
            dn="org-root/org-tenant1/zone-trustedServers-0"
            intId="24404"
            name="trustedServers-0"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/zone-trustedClients-0">
         <policyZone
            descr=""
            dn="org-root/org-tenant1/zone-trustedClients-0"
            intId="24408"
            name="trustedClients-0"
            status="created"/>
      </pair>
   </outConfigs>
</configConfMos>
```

オブジェクト グループ

次の例では、ftpPorts0 という名前のオブジェクト グループが作成され、属性セットが割り当てられます。

要求

```
Post URL : https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
```

```
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/org-tenant0/objgrp-ftpPorts0">
            <policyObjectGroup dn="org-root/org-tenant0/objgrp-ftpPorts0">
                <policyObjectGroupExpression id="1" order="1" opr="eq">
                    <policyNetworkPort id="1" value="20" />
                </policyObjectGroupExpression>
                <policyObjectGroupExpression id="2" order="2" opr="eq">
                    <policyNetworkPort id="1" value="21" />
                </policyObjectGroupExpression>
            </policyObjectGroup>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/15/0/1a4"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/org-tenant0/objgrp-ftpPorts0">
            <policyObjectGroup
                attributeName=""
                descr=""
                dn="org-root/org-tenant0/objgrp-ftpPorts0"
                intId="24265"
                name="ftpPorts0"
                status="created"/>
        </pair>
    </outConfigs>
</configConfMos>
```

属性ディクショナリ

次の例では、さまざまな属性 ID と名前を定義するディクショナリが作成されます。

要求

```
Post URL : https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
<!-- Create Sample VnspCustomDictionary under org-tenant1 -->
        <pair key="org-root/attr-dict-custom-userAttrs">
            <policyVnspCustomDictionary dn="org-root/attr-dict-custom-userAttrs">
                <policyVnspCustomAttr dataType="string" id="1" name="userAttr1" />
                <policyVnspCustomAttr dataType="string" id="2" name="userAttr2" />
                <policyVnspCustomAttr dataType="string" id="3" name="dept" />
                <policyVnspCustomAttr dataType="string" id="4" name="production" />
            </policyVnspCustomDictionary>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
```

```
    cookie="<real_cookie>"
    commCookie="7/15/0/1a3"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/attr-dict-custom-userAttrs">
            <policyVnspCustomDictionary
                descr=""
                dn="org-root/attr-dict-custom-userAttrs"
                intId="24245"
                name="userAttrs"
                status="created"/>
        </pair>
    </outConfigs>
</configConfMos>
```

ポリシー

バージョン 2.0 から、VSG コンピュート ファイアウォールはこのプロパティで無効の値をサポートせずに、デフォルトの値が有効なので、VNMC は ACL ポリシーに対して adminState プロパティを使用しなくなりました。ただし、VNMC は ASA 1000V で使用されるような他のポリシー タイプで adminState のプロパティを使用し続けます。

ACL ポリシーに対して API で adminState プロパティを設定する場合、応答には次のエラー メッセージが含まれます。

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/12/0/55"
    srcExtSys="10.193.76.15"
    destExtSys="10.193.76.15"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes"
    errorCode="170"
    invocationResult="unidentified-fail"
    errorDescr="Admin implicit props cannot be modified, prop=adminState>
```

次の例では、trustedHosts という名前のポリシーが作成され、使用できるルールが設定されます。

要求

```
Post URL：https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
        <pair key="org-root/org-tenant1/pol-trustedHosts">
            <policyRuleBasedPolicy dn="org-root/org-tenant1/pol-trustedHosts">
                <policyRule name="allowSsh" order="1">
<!-- This rule allows all VMs in zone "trustedClients" to initiate an SSH
connection to VMs in zone "trustedServers" -->
                <policyRuleCondition id="100" order="1">
                    <policyNetworkExpression opr="eq">
                    <policyNwAttrQualifier attrEp="source"/>
                    <policyZoneNameRef id="1" value="trustedClients-0" />
                    </policyNetworkExpression>
                </policyRuleCondition>
                <policyRuleCondition id="101" order="20">
                    <policyNetworkExpression opr="eq">
                    <policyNwAttrQualifier attrEp="destination"/>
                    <policyZoneNameRef id="1" value="trustedServers-0" />
                    </policyNetworkExpression>
```

```
            </policyRuleCondition>
            <policyRuleCondition id="103" order="30">
               <policyNetworkExpression opr="eq">
               <policyNwAttrQualifier attrEp="destination"/>
               <policyNetworkPort id="1" placement="0" value="22" />
               </policyNetworkExpression>
            </policyRuleCondition>
            <fwpolicyAction actionType="permit"/>
            </policyRule>
            <policyRule name="allowTacacs" order="2">
            <fwpolicyAction actionType="permit"/>
            </policyRule>
         </policyRuleBasedPolicy>
      </pair>
   </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
   cookie="<real_cookie>"
   commCookie="7/15/0/1b5"
   srcExtSys="10.193.33.221"
   destExtSys="10.193.33.221"
   srcSvc="sam_extXMLApi"
   destSvc="policy-mgr_dme"
   response="yes">
   <outConfigs>
      <pair key="org-root/org-tenant1/pol-trustedHosts">
         <policyRuleBasedPolicy
            descr=""
            dn="org-root/org-tenant1/pol-trustedHosts"
            intId="25131"
            name="trustedHosts"
            status="created"/>
      </pair>
   </outConfigs>
</configConfMos>
```

PolicySet

次の例では、ACL-PolicySet が作成され、ポリシーが適用される順序が設定されます。

要求

```
<configConfMos
   cookie="<real_cookie>"
   inHierarchical="false">
   <inConfigs>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test">
         <policyPolicyNameRef
            dn="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test"
            order="100"
            policyName="Test"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet">
         <policyPolicySet
            descr=""
            dn="org-root/org-tenant1/pset-ACL-PolicySet"
            name="ACL-PolicySet"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant/pset-ACL-PolicySet/polref-Test-03">
         <policyPolicyNameRef
```

```
            dn="org-root/org-tenant/pset-ACL-PolicySet/polref-Test-03"
            order="300"
            policyName="Test-03"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test-02">
         <policyPolicyNameRef
            dn="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test-02"
            order="200"
            policyName="Test-02"
            status="created"/>
      </pair>
   </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
   cookie="<real_cookie>"
   commCookie="7/12/0/187c"
   srcExtSys="172.20.101.150"
   destExtSys="172.20.101.150"
   srcSvc="sam_extXMLApi"
   destSvc="policy-mgr_dme"
   response="yes">
   <outConfigs>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet">
         <policyPolicySet
            adminState="enabled"
            descr="" dn="org-root/org-tenant1/pset-ACL-PolicySet"
            intId="10666"
            name="ACL-PolicySet"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test">
         <policyPolicyNameRef
            dn="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test"
            order="100"
            policyName="Test"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test-02">
         <policyPolicyNameRef
            dn="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test-02"
            order="200"
            policyName="Test-02"
            status="created"/>
      </pair>
      <pair key="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test-03">
         <policyPolicyNameRef
            dn="org-root/org-tenant1/pset-ACL-PolicySet/polref-Test-03"
            order="300"
            policyName="Test-03"
            status="created"/>
      </pair>
   </outConfigs>
</configConfMos>
```

Compute Security Profile

次の例では、vnsp-sp1 という名前のセキュリティ プロファイルが作成され、VNSP がそのセキュリティ プロファイルに割り当てられます。

要求

```
Post URL : https://10.193.33.221/xmlIM/policy-mgr
XML API payload:
<configConfMos
cookie="<real_cookie>">
<inConfigs>
<pair key="org-root/org-tenant0/vnsp-sp1">
   <policyVirtualNetworkServiceProfile
      dn="org-root/org-tenant0/vnsp-sp1"
      policySetNameRef="myPolicySet0">
      <policyVnspAVPair id="1">
         <policyAttributeDesignator attrName="dept"/>
         <policyAttributeValue value="DEV" />
      </policyVnspAVPair>
      </policyVirtualNetworkServiceProfile>
      </pair>
   </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
   cookie="<real_cookie>"
   commCookie="7/15/0/1ac"
   srcExtSys="10.193.33.221"
   destExtSys="10.193.33.221"
   srcSvc="sam_extXMLApi"
   destSvc="policy-mgr_dme"
   response="yes">
   <outConfigs>
      <pair key="org-root/org-tenant0/vnsp-sp1">
         <policyVirtualNetworkServiceProfile
            descr=""
            dn="org-root/org-tenant0/vnsp-sp1"
            intId="24512"
            name="sp1"
            policySetNameRef="myPolicySet0"
            status="created"
            vnspId="2"/>
      </pair>
   </outConfigs>
</configConfMos>
```

リソース管理

リソース管理コンポーネントは、次の機能を実行します。

- エッジ ファイアウォールを作成します。
- ポリシーにコンピュート ファイアウォールとエッジ ファイアウォールを割り当てます。
- 利用可能なファイアウォール インスタンスがないか問い合わせます。
- ファイアウォール インスタンスを管理対象オブジェクトに関連付けます。
- 組織をリソース プールにまとめることができます。
- 仮想センターと対話して、VM 属性を取得します。

ここでは、ファイアウォールでの作業に関する例を示します。

- エッジ ファイアウォールの作成
- コンピュート ファイアウォールの作成
- コンピュート ファイアウォールへのデバイス プロファイルの割り当て
- ファイアウォール インスタンスの問い合わせ

エッジ ファイアウォールの作成

次の例では、データ インターフェイス内とデータ インターフェイス外に 1 つずつエッジ ファイアウォールを作成します。

要求

```
<configConfMos
    cookie="<real_cookie">
    <inConfigs>
        <pair key="org-root/efw-default">
            <fwEdgeFirewall
                dn="org-root/efw-default"
                rn="efw-default"
                name="default"
                haMode="standalone"
                status="created"/>
        </pair>
        <pair key="org-root/efw-default/interface-inside">
            <fwDataInterface
                dn="org-root/efw-default/interface-inside"
                name="inside"
                role="inside"
                ipSubnet="255.255.255.0"
                ipAddressPrimary="10.10.10.1"
                status="created"/>
        </pair>
        <pair key="org-root/efw-default/interface-outside">
            <fwDataInterface
                dn="org-root/efw-default/interface-outside"
                name="outside"
                role="outside"
                ipSubnet="255.255.255.0"
                ipAddressPrimary="10.10.20.1"
                status="created"/>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="5/12/0/35da"
    srcExtSys="172.20.101.150"
    destExtSys="172.20.101.150"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/efw-default">
            <fwEdgeFirewall
                assocState="unassociated"
                auxProps=""
                configState="not-applied"
                descr=""
                dn="org-root/efw-default"
                fltAggr="0"
                haMode="standalone"
                hostname="edge-firewall"
                intId="10467"
                name="default"
                status="created"/>
        </pair>
        <pair key="org-root/efw-default/interface-inside">
            <fwDataInterface
```

```
                descr=""
                dn="org-root/efw-default/interface-inside"
                intId="10468"
                ipAddressPrimary="10.10.10.1"
                ipAddressSecondary="0.0.0.0"
                ipSubnet="255.255.255.0"
                isIpViaDHCP="no"
                name="inside"
                role="inside"
                status="created"/>
        </pair>
        <pair key="org-root/efw-default/interface-outside">
            <fwDataInterface
                descr=""
                dn="org-root/efw-default/interface-outside"
                intId="10469"
                ipAddressPrimary="10.10.20.1"
                ipAddressSecondary="0.0.0.0"
                ipSubnet="255.255.255.0"
                isIpViaDHCP="no"
                name="outside"
                role="outside"
                status="created"/>
        </pair>
    </outConfigs>
</configConfMos>
```

コンピュート ファイアウォールの作成

次の例では、test5 という名前のコンピュート ファイアウォールを作成します。

要求

```
<configConfMos
    cookie="<real_cookie>"
    inHierarchical="false">
    <inConfigs>
        <pair key="org-root/cfw-test5">
            <fwComputeFirewall
                dn="org-root/cfw-test5"
                name="test5"
                dataIpAddress="5.5.5.5"
                dataIpSubnet="255.255.255.0"
                status="created"/>
        </pair>
    </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="5/12/0/1545"
    srcExtSys="172.25.97.246"
    destExtSys="172.25.97.246"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="org-root/cfw-test5">
            <fwComputeFirewall
                assocState="unassociated"
                auxProps=""
                configState="not-applied"
                dataIpAddress="5.5.5.5"
                dataIpSubnet="255.255.255.0"
                descr=""
```

```
            devicePolicy=""
            dn="org-root/cfw-test5"
            fltAggr="0"
            hostname="firewall"
            intId="10583"
            name="test5"
            status="created"/>
      </pair>
   </outConfigs>
</configConfMos>
```

コンピュート ファイアウォールへのデバイス プロファイルの割り当て

VNMC 2.0 から、fw:ComputeFirewall.devicePolicy は廃止されているため、XML API で変更しないでください。代わりに、タイプ logical:DeviceProfileAssociation の子 MO を作成し、logical:DeviceProfileAssociation.profileRef を設定してください。logical:DeviceProfileAssociation は fw:ComputeFirewall の子です。

fw:ComputeFirewall.devicePolicy を変更する場合は、fw:ComputeFirewall.devicePolicy および logical:DeviceProfileAssociation 間の不整合な動作、および UI に表示される内容を参照してください。

次に、デバイス プロファイル my-profile-ref をコンピュート ファイアウォール test5 に割り当てる例を示します。これは、P. 2-28 の「コンピュート ファイアウォールの作成」で作成されました。

要求

```
<configConfMos
   cookie="<real_cookie>"
   inHierarchical="false">
   <inConfigs>
      <pair key="org-root/cfw-test5/device-profile">
         <logicalDeviceProfileAssociation
            dn="org-root/cfw-test5/device-profile"
            profileRef="my-profile-ref"
            status="created"/>
      </pair>
   </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
   cookie="<real_cookie>"
   commCookie="5/12/0/1571"
   srcExtSys="172.25.97.246"
   destExtSys="172.25.97.246"
   srcSvc="sam_extXMLApi"
   destSvc="resource-mgr_dme"
   response="yes">
   <outConfigs>
      <pair key="org-root/cfw-test5/device-profile">
         <logicalDeviceProfileAssociation
            descr=""
            dn="org-root/cfw-test5/device-profile"
            intId="10586"
            name=""
            profileRef="my-profile-ref"
            status="created"/>
      </pair>
   </outConfigs>
</configConfMos>
```

ファイアウォール インスタンスの問い合わせ

次の例は、管理 IP アドレスが 10.193.33.221 のすべてのファイアウォール インスタンスを照会し、その属性を取得します。

要求

```
POST URL：https://10.193.33.221/xmlIM/resource-mgr
XML API payload:
<configResolveClass
    cookie="<real_cookie>"
    classId="fwInstance"
    inHierarchical="false">
    <inFilter>
        <eq class="fwInstance"
            property="mgmtIp"
            value="10.193.33.221" />
    </inFilter>
</configResolveClass>
```

応答

```
configResolveClass
    cookie=<real_cookie>"
    commCookie="5/15/0/1d9"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes"
    classId="fwInstance">
    <outConfigs>
        <fwInstance
            assignedToDn=""
            assoc="none"
            descr=""
            dn="fw/inst-1005"
            fltAggr="0"
            fsmDescr=""
            fsmPrev="DisassociateSuccess"
            fsmProgr="100"
            fsmRmtInvErrCode="none"
            fsmRmtInvErrDescr=""
            fsmRmtInvRslt=""
            fsmStageDescr=""
            fsmStamp="2010-12-03T23:18:13.304"
            fsmStatus="nop"
            fsmTry="0"
            intId="10155"
            mgmtIp="10.193.33.221"
            model=""
            name="firewall"
            pooled="0"
            registeredClientDn="extpol/reg/clients/client-1005"
            revision="0"
            serial=""
            svcId="1005"
            vendor=""/>
    </outConfigs>
</configResolveClass>
```

VNMC XML API メソッド

この項では、次のトピックについて取り上げます。

- サポートされていないメソッド
- サポートされているメソッド

サポートされていないメソッド

シスコでは VNMC サーバのメソッド リストに表示されている場合でも次のメソッドをサポートしていません。

注意：次のメソッドを使用しないことを強く推奨します。これは、VNMC サーバで予期せぬ結果を引き起こす可能性があります。

- aaaCheckComputeAuthToken
- aaaCheckComputeExtAccess
- aaaGetComputeAuthToken
- cliviewConfMos
- configFindDependencies
- configMoChangeEvent
- fsmDebugAction
- methodVessel
- orgResolveLogicalParents
- policyEstimateImpact
- statsClearInterval
- syntheticFSObjInventory
- syntheticTestTx

サポートされているメソッド

ここでは、次のメソッドの説明、構文（要求および応答）、使用例について説明します。

- aaaGetRemoteUserRoles
- aaaGetUserLocales
- aaaKeepAlive
- aaaLogin
- aaaLogout
- aaaRefresh
- configConfFiltered
- configConfMo
- configConfMoGroup
- configConfMos
- configFindDnsByClassId
- configResolveChildren
- configResolveClass
- configResolveClasses
- configResolveDn
- configResolveDns
- configResolveParent
- configScope
- eventSendHeartbeat
- eventSubscribe
- eventSubscribeApps
- faultAckFault
- faultAckFaults
- faultResolveFault
- loggingSyncOcns
- orgResolveElements
- orgResolveInScope
- poolResolveInScope

これらのメソッドは、GUI コンソールからも呼び出されます。

aaaGetRemoteUserRoles

この API メソッドは、リモート ロケーションのユーザ特権を返します。

要求構文

```
<xs:element name="aaaGetRemoteUserRoles" type="aaaGetRemoteUserRoles"
substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaGetRemoteUserRoles" mixed="true">
      <xs:attribute name="inRemoteUserName">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="[a-zA-Z][a-zA-Z0-9_.@-]{0,31}"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="aaaGetRemoteUserRoles" type="aaaGetRemoteUserRoles"
substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaGetRemoteUserRoles" mixed="true">
      <xs:attribute name="outRemoteUserPriv">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:pattern
value="((policy|aaa|read-only|admin|tenant|operations|res config|fault),){0,7}
(policy|aaa|read-only|admin|tenant|operations|res-config|fault){0,1}"/>
               </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
         </xs:complexType>
```

例

要求

```
<aaaGetRemoteUserRoles
   cookie="<real_cookie>"
   inRemoteUserName="adminuser"
   outRemoteUserPriv/>
```

応答

```
<aaaGetRemoteUserRoles
   cookie="<real_cookie>"
   commCookie="11/15/0/2964"
   srcExtSys="10.193.33.109"
   destExtSys="10.193.33.109"
   srcSvc="sam_extXMLApi"
   destSvc="mgmt-controller_dme"
   response="yes"
   outRemoteUserPriv="admin">
</aaaGetRemoteUserRoles>
```

aaaGetUserLocales

この API メソッドは、認可されたユーザ ロケーションのリストを返します。

要求構文

```
<xs:element name="aaaGetUserLocales" type="aaaGetUserLocales"
substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaGetUserLocales" mixed="true">
      <xs:attribute name="inUserName">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="[a-zA-Z][a-zA-Z0-9_.@-]{0,31}"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inIsUserRemote">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
```

```
                        <xs:enumeration value="no"/>
                        <xs:enumeration value="yes"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:union>
        </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="cookie" type="xs:string"/>
    <xs:attribute name="response" type="YesOrNo"/>
</xs:complexType>
```

応答構文

```
<xs:element name="aaaGetUserLocales" type="aaaGetUserLocales"
substitutionGroup="externalMethod"/>
    <xs:complexType name="aaaGetUserLocales" mixed="true">
        <xs:attribute name="outUserLocales">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:minLength value="0"/>
                    <xs:maxLength value="512"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

```
<aaaGetUserLocales
    cookie="<real_cookie>"
    inUserName="john"
    inIsUserRemote="no"
    outUserLocales/>
```

応答

```
<aaaGetUserLocales
    cookie="<real_cookie>"
    commCookie="11/15/0/2962"
    srcExtSys="10.193.33.109"
    destExtSys="10.193.33.109"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes"
    outUserLocales="TestSanity">
</aaaGetUserLocales>
```

aaaKeepAlive

次の例は、デフォルトのセッション時間が経過するまでセッションをアクティブなままにし、メソッド呼び出し後に同じ Cookie を使用します。

要求構文

```
<xs:element name="aaaKeepAlive" type="aaaKeepAlive"
substitutionGroup="externalMethod"/>
    <xs:complexType name="aaaKeepAlive" mixed="true">
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
```

```
</xs:complexType>
```

応答構文

```
<xs:element name="aaaKeepAlive" type="aaaKeepAlive"
substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaKeepAlive" mixed="true">
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<aaaKeepAlive
   cookie="<real_cookie>" />
```

応答

```
<aaaKeepAlive
   cookie="<real_cookie>"
   commCookie="11/15/0/2969"
   srcExtSys="10.193.33.109"
   destExtSys="10.193.33.109"
   srcSvc="sam_extXMLApi"
   destSvc="mgmt-controller_dme"
   response="yes">
</aaaKeepAlive>
```

aaaLogin

次の例は、セッションを開始するために必要で、クライアントと VNMC 間で認証された HTTPS セッションを確立するログイン プロセスを示しています。

要求構文

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaLogin" mixed="true">
      <xs:attribute name="inName">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inPassword">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="0"/>
               <xs:maxLength value="512"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaLogin" mixed="true">
```

```
      <xs:attribute name="outCookie" type="xs:string"/>
      <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
      <xs:attribute name="outPriv">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:pattern
value="((policy|aaa|read-only|admin|tenant|operations|res
config|fault),){0,7}(policy|aaa|read-only|admin|tenant|operations|res-
config|fault){0,1}"/>
         </xs:restriction>
      </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="outDomains" type="xs:string"/>
      <xs:attribute name="outChannel">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration value="fullssl"/>
               <xs:enumeration value="noencssl"/>
               <xs:enumeration value="plain"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="outEvtChannel">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration value="fullssl"/>
               <xs:enumeration value="noencssl"/>
               <xs:enumeration value="plain"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="outSessionId">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="1"/>
               <xs:maxLength value="64"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="outVersion" type="xs:string"/>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<aaaLogin
   inName="admin"
   inPassword="Nbv12345"/>
```

応答

```
<aaaLogin cookie=""
   commCookie=""
   srcExtSys="0.0.0.0"
   destExtSys="0.0.0.0"
   srcSvc="" destSvc=""
   response="yes"
   outCookie="<real_cookie>"
```

```
    outRefreshPeriod="600"
    outPriv="admin"
    outDomains=""
    outChannel="fullssl"
    outEvtChannel="fullssl"
    outSessionId="web_49019"
    outVersion="1.0(0.39938)">
</aaaLogin>
```

aaaLogout

次に、現在のセッションを終了するログアウト プロセスの例を示します。デフォルトのセッション時間が経過すると、このプロセスは自動的に呼び出されます。

要求構文

```
<xs:element name="aaaLogout" type="aaaLogout"
substitutionGroup="externalMethod"/>
    <xs:complexType name="aaaLogout" mixed="true">
        <xs:attribute name="inCookie" type="xs:string"/>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
    </xs:complexType>
```

応答構文

```
<xs:element name="aaaLogout" type="aaaLogout"
substitutionGroup="externalMethod"/>
    <xs:complexType name="aaaLogout" mixed="true">
        <xs:attribute name="outStatus">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="success"/>
                    <xs:enumeration value="failure"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

```
<aaaLogout
    inCookie="<real_cookie>"
    outStatus/>
```

応答

```
<aaaLogout cookie=""
    commCookie=""
    srcExtSys="0.0.0.0"
    destExtSys="0.0.0.0"
    srcSvc=""
    destSvc=""
    response="yes"
    outStatus="success">
</aaaLogout>
```

aaaRefresh

セッションはユーザ アクティビティによってアクティブに保つことができます（デフォルトのセッション時間枠内）。デフォルトでは、アクティビティがない時点から 7200 秒カウントダウンされます。7200 秒が経過すると、VNMC はスリープ モードを開始し、再度サインインを要求します。これにより、カウント ダウンが再起動されます。セッションは同じセッション ID を使用し続けます。

（注） このメソッドを使用すると、以前の Cookie の有効期限が切れ、新しい Cookie が発行されます。

要求構文

```
<xs:element name="aaaRefresh" type="aaaRefresh"
substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaRefresh" mixed="true">
      <xs:attribute name="inName">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inPassword">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:minLength value="0"/>
               <xs:maxLength value="512"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inCookie" type="xs:string"/>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="aaaRefresh" type="aaaRefresh"
substitutionGroup="externalMethod"/>
   <xs:complexType name="aaaRefresh" mixed="true">
      <xs:attribute name="outCookie" type="xs:string"/>
      <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
      <xs:attribute name="outPriv">
      <xs:simpleType>
         <xs:restriction base="xs:string">
            <xs:pattern
value="((policy|aaa|read-only|admin|tenant|operations|res-
config|fault),){0,7}(policy|aaa|
read-only|admin|tenant|operations|res-config|fault){0,1}"/>
         </xs:restriction>
      </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="outDomains" type="xs:string"/>
      <xs:attribute name="outChannel">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration value="fullssl"/>
               <xs:enumeration value="noencssl"/>
               <xs:enumeration value="plain"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="outEvtChannel">
```

```
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:enumeration value="fullssl"/>
                  <xs:enumeration value="noencssl"/>
                  <xs:enumeration value="plain"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="cookie" type="xs:string"/>
         <xs:attribute name="response" type="YesOrNo"/>
         <xs:attribute name="errorCode" type="xs:unsignedInt"/>
         <xs:attribute name="errorDescr" type="xs:string"/>
         <xs:attribute name="invocationResult" type="xs:string"/>
      </xs:complexType>
```

例

要求

```
<aaaRefresh
    cookie="<real_cookie>"
    inName="admin"
    inPassword="Nbv12345"
    inCookie="<real_cookie>"/>
```

応答

```
<aaaRefresh
    cookie="<real_cookie>"
    commCookie="" srcExtSys="0.0.0.0"
    destExtSys="0.0.0.0"
    srcSvc=""
    destSvc=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin"
    outDomains=""
    outChannel="fullssl"
    outEvtChannel="fullssl">
</aaaRefresh>
```

configCloneMo

特定の Mos（たとえば、Org）はその階層および操作を実行しているユーザのアクセス権に基づいて複製できます。Mo が複製されている場合、その子がすべて cloningMo のデフォルトのアクセス権で複製されます。この操作の構文は次のとおりです。

要求構文

```
<xs:element name="configCloneMo" type="configCloneMo"
substitutionGroup="externalMethod"/>
       <xs:complexType name="configCloneMo" mixed="true">
          <xs:attribute name="inToDn" type="referenceObject"/>
             <xs:attribute name="cookie" type="xs:string"/>
             <xs:attribute name="response" type="YesOrNo"/>
             <xs:attribute name="dn" type="referenceObject"/>
       </xs:complexType>
```

応答構文

```
<xs:element name="configCloneMo">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="outConfig">
          <xs:complexType>
            <xs:sequence>
            </xs:sequence>
```

```
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute type="xs:string" name="dn"/>
      <xs:attribute type="xs:string" name="cookie"/>
      <xs:attribute type="xs:string" name="commCookie"/>
      <xs:attribute type="xs:string" name="srcExtSys"/>
      <xs:attribute type="xs:string" name="destExtSys"/>
      <xs:attribute type="xs:string" name="srcSvc"/>
      <xs:attribute type="xs:string" name="destSvc"/>
      <xs:attribute type="xs:string" name="response"/>
    </xs:complexType>
  </xs:element>
```

例

要求

```
<configCloneMo cookie="<real_cookie>"
       dn="org-root/org-Tenant-1"
       inToDn="org-root/org-Tenant-3">
</configCloneMo>
```

応答

```
<configCloneMo
       dn="org-root/org-Tenant-1"
       cookie="<real_cookie> "
       commCookie="5/12/0/14e4"
       srcExtSys=""
       destExtSys=""
       srcSvc="sam_extXMLApi"
       destSvc="resource-mgr_dme"
       response="yes">
       <outConfig>
          <orgTenant
                    childAction="deleteNonPresent"
                    descr=""
                    dn="org-root/org-Tenant-3"
                    fltAggr="0"
                    level="1"
                    name="Tenant-3"
                    status="created">
          </orgTenant>
       </outConfig>
</configCloneMo>
```

configConfFiltered

次に、設定されたポリシーに従ってデータおよびアクティビティを制限する例を示します。

要求構文

```
<xs:element name="configConfFiltered" type="configConfFiltered"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configConfFiltered" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
         <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
```

```
                    </xs:restriction>
                </xs:simpleType>
            </xs:union>
        </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="cookie" type="xs:string"/>
    <xs:attribute name="response" type="YesOrNo"/>
    <xs:attribute name="classId" type="namingClassId"/>
</xs:complexType>
```

応答構文

```
<xs:element name="configConfFiltered" type="configConfFiltered"
substitutionGroup="externalMethod"/>
    <xs:complexType name="configConfFiltered" mixed="true">
        <xs:all>
            <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
        <xs:attribute name="classId" type="namingClassId"/>
    </xs:complexType>
```

例

要求

```
<configConfFiltered
    cookie="<real_cookie>"
    inHierarchical="false"
    classId="orgTenant">
    <inFilter>
        <eq class="orgTenant"
            property="name"
            value="Tenant1" />
    </inFilter>
    <inConfig>
        <orgDatacenter
            dn="org-HR"
            descr="HR (Human Resources- new Descr)"/>
    </inConfig>
</configConfFiltered>
```

応答

```
<configConfFiltered
    cookie="<real_cookie>"
    commCookie="5/15/0/617"
    srcExtSys="10.193.33.206"
    destExtSys="10.193.33.206"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes"
    classId="orgTenant">
    <outConfigs>
        <orgDatacenter
            descr="HR (Human Resources- new Descr)"
            dn="org-root/org-tenant1/org-HR"
            fltAggr="0"
            level="2"
            name="HR"
            status="modified"/>
    </outConfigs>
```

```
</configConfFiltered>
```

configConfMo

次に、単一のサブツリー（DN など）で、指定された管理対象オブジェクトを設定する例を示します。

要求構文

```
<xs:element name="configConfMo" type="configConfMo"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configConfMo" mixed="true">
      <xs:all>
         <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configConfMo" type="configConfMo"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configConfMo" mixed="true">
      <xs:all>
         <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

例

要求

```
<configConfMo
   dn=""
   cookie="<real_cookie>"
   inHierarchical="false">
   <inConfig>
      <aaaLdapEp
         attribute="CiscoAvPair"
         basedn="dc=pasadena,dc=company123,dc=com"
         descr=""
         dn="sys/ldap-ext"
         filter="sAMAccountName=$userid"
         retries="1"
         status="modified"
```

```
            timeout="30"/>
    </inConfig>
</configConfMo>
```

応答

```
<configConfMo
    dn=""
    cookie="<real_cookie>"
    commCookie="11/15/0/28"
    srcExtSys="10.193.33.101"
    destExtSys="10.193.33.101"
    destSvc="mgmt-controller_dme"
    response="yes">
    <outConfig>
        <aaaLdapEp
            attribute="CiscoAvPair"
            basedn="dc=pasadena,dc=company123,dc=com"
            childAction="deleteNonPresent"
            descr=""
            dn="sys/ldap-ext"
            filter="sAMAccountName=$userid"
            fsmDescr=""
            fsmPrev="updateEpSuccess"
            fsmProgr="100"
            fsmStageDescr=""
            fsmStamp="2010-11-22T23:41:01.826"
            fsmStatus="nop"
            fsmTry="0"
            intId="10027"
            name=""
            retries="1"
            status="modified"
            timeout="30"/>
    </outConfig>
</configConfMo>
```

configConfMoGroup

次に、設定されたポリシーに基づいて、管理対象オブジェクトのグループを設定する例を示します。

要求構文

```
<xs:element name="configConfMoGroup" type="configConfMoGroup"
substitutionGroup="externalMethod"/>
    <xs:complexType name="configConfMoGroup" mixed="true">
        <xs:all>
            <xs:element name="inDns" type="dnSet" minOccurs="0"/>
            <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="inHierarchical">
            <xs:simpleType>
                <xs:union memberTypes="xs:boolean">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:enumeration value="no"/>
                            <xs:enumeration value="yes"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:union>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
    </xs:complexType>
```

応答構文

```
<xs:element name="configConfMoGroup" type="configConfMoGroup"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configConfMoGroup" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

（注） orgDataCenter（org-root/org-tenant1 および org-root/org-tenant2 下）の descr プロパティが変更されます。descr プロパティは暗黙的でないため、変更できます。暗黙的な場合、変更は適用されず、新しい orgDataCenter が作成されます。

orgDataCenter（org-root/org-tenant1 および org-root/org-tenant2 下）の descr プロパティが変更されます。descr プロパティは暗黙的でないため、変更できます。暗黙的な場合、変更は適用されず、新しい orgDataCenter が作成されます。

要求

```
<configConfMoGroup
   cookie="<real_cookie>"
   inHierarchical="false">
   <inDns>
      <dn value="org-root/org-tenant1" />
      <dn value="org-root/org-tenant2" />
   </inDns>
   <inConfig>
      <orgDatacenter
         dn="org-HR"
         descr="HR (Human Resources)"/>
   </inConfig>
</configConfMoGroup>
```

応答

```
<configConfMoGroup
   cookie="<real_cookie>"
   commCookie="5/15/0/600"
   srcExtSys="10.193.33.206"
   destExtSys="10.193.33.206"
   srcSvc="sam_extXMLApi"
   destSvc="resource-mgr_dme"
   response="yes">
   <outConfigs>
      <orgDatacenter
         descr="HR (Human Resources)"
         dn="org-root/org-Cola/org-HR"
         fltAggr="0"
         level="2"
         name="HR"
         status="modified"/>
      <orgDatacenter
         descr="HR (Human Resources)"
         dn="org-root/org-tenant1/org-HR"
         fltAggr="0"
         level="2"
         name="HR"
         status="modified"/>
```

```
   </outConfigs>
</configConfMoGroup>
```

configConfMos

次に、複数のサブツリーで DN を使用して管理対象オブジェクトを設定する例を示します。

要求構文

```
<xs:element name="configConfMos" type="configConfMos"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configConfMos" mixed="true">
      <xs:all>
         <xs:element name="inConfigs" type="configMap" minOccurs="0">
            <xs:unique name="unique_map_key_2">
               <xs:selector xpath="pair"/>
               <xs:field xpath="@key"/>
            </xs:unique>
         </xs:element>
      </xs:all>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configConfMos" type="configConfMos"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configConfMos" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configMap" minOccurs="0">
            <xs:unique name="unique_map_key_4">
               <xs:selector xpath="pair"/>
               <xs:field xpath="@key"/>
            </xs:unique>
         </xs:element>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<configConfMos
   cookie="<real_cookie>">
```

```
   <inConfigs>
      <pair key="org-root/logprof-default">
         <policyLogProfile dn="org-root/logprof-default"
            name="default"
            level="debug1"
            size="10000000"
            backupCount="4"/>
      </pair>
<!-- Update Controller Device Profile -->
      <pair key="org-root/controller-profile-default">
         <policyControllerDeviceProfile
            dn="org-root/controller-profile-default"
            adminState="enabled">
            <commDnsProvider hostip="171.70.168.183" order="1"/>
            <commDnsProvider hostip="171.68.226.120" order="2"/>
            <commDnsProvider hostip="64.102.6.247" order="3"/>
         </policyControllerDeviceProfile>
      </pair>
   </inConfigs>
</configConfMos>
```

応答

```
<configConfMos
   cookie="<real_cookie>"
   commCookie="7/15/0/1a74"
   srcExtSys="10.193.34.70"
   destExtSys="10.193.34.70"
   srcSvc="sam_extXMLApi"
   destSvc="policy-mgr_dme"
   response="yes">
   <outConfigs>
      <pair key="org-root/logprof-default">
         <policyLogProfile
            backupCount="4"
            descr="the log level for every process"
            dn="org-root/logprof-default"
            intId="10065"
            level="debug1"
            name="default"
            size="10000000"/>
      </pair>
      <pair key="org-root/controller-profile-default">
         <policyControllerDeviceProfile
            adminState="enabled"
            coreFilePolicy=""
            descr="default profile for management server virtual machine"
            dn="org-root/controller-profile-default"
            dnsPolicy=""
            faultPolicy="default"
            httpPolicy="default"
            httpsPolicy="default"
            intId="10057"
            logProfilePolicy="default"
            name="default"
            snmpPolicy=""
            syslogPolicy=""
            telnetPolicy=""
            timezone=""/>
      </pair>
   </outConfigs>
</configConfMos>
```

configFindDnsByClassId

次に、識別名を検索し、クラス ID でソートされた識別名を返す例を示します。

要求構文

```
<xs:element name="configFindDnsByClassId" type="configFindDnsByClassId"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configFindDnsByClassId" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="classId" type="namingClassId"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configFindDnsByClassId" type="configFindDnsByClassId"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configFindDnsByClassId" mixed="true">
      <xs:all>
         <xs:element name="outDns" type="dnSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
      <xs:attribute name="classId" type="namingClassId"/>
   </xs:complexType>
```

例

要求

```
<configFindDnsByClassId
   classId="eventRecord"
   cookie="<real_cookie>" />
```

応答

```
<configFindDnsByClassId
   cookie="<real_cookie>"
   commCookie="2/12/0/1810"
   srcExtSys="172.20.101.150"
   destExtSys="172.20.101.150"
   srcSvc="sam_extXMLApi"
   destSvc="service-reg_dme"
   response="yes"
   classId="eventRecord">
   <outDns>
      <dn value="event-log/10210"/>
      <dn value="event-log/10250"/>
      <dn value="event-log/10211"/>
      <dn value="event-log/10221"/>
      <dn value="event-log/10251"/>
      <dn value="event-log/10141"/>
      <dn value="event-log/10151"/>
   </outDns>
</configFindDnsByClassId>
```

configResolveChildren

次に、管理対象情報ツリーで特定の DN 下にある管理対象オブジェクトの子を取得する例を示します。返される子の数を減らすためにフィルタを使用できます。

要求構文

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveChildren" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inDn" type="referenceObject"/>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="classId" type="namingClassId"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveChildren" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
      <xs:attribute name="classId" type="namingClassId"/>
   </xs:complexType>
```

例

要求

```
<configResolveChildren
   cookie="<real_cookie>"
   classId="aaaUser"
   inDn="sys/user-ext"
   inHierarchical="false">
   <inFilter>
   </inFilter>
</configResolveChildren>
```

応答

```
<configResolveChildren
    cookie="<real_cookie>"
    commCookie="11/15/0/2a59"
    srcExtSys="10.193.33.120"
    destExtSys="10.193.33.120"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes"
    classId="aaaUser">
    <outConfigs>
        <aaaUser descr="" dn="sys/user-ext/user-doe"
            email="" expiration="never" expires="no" firstName="John" intId="12999"
            lastName="Doe" name="doe" phone="" priv="admin,read-only"
            pwdSet="yes"/>
        <aaaUser descr="" dn="sys/user-ext/user-jacks" email="" expiration="never"
            expires="no" firstName="Play" intId="12734" lastName="Jacks"
            name="jacks"
            phone="" priv="fault,operations,policy,read-only,res-config,tenant"
            pwdSet="yes"/>
        <aaaUser descr="" dn="sys/user-ext/user-admin" email="" expiration="never"
            expires="no" firstName="" intId="10052" lastName="" name="admin"
            phone=""
            priv="admin,read-only" pwdSet="yes"/>
        <aaaUser descr="" dn="sys/user-ext/user-over" email="" expiration="never"
            expires="no" firstName="Roll" intId="12711" lastName="Over" name="over"
            phone="" priv="fault,operations,policy,read-only,res-config,tenant"
            pwdSet="yes"/>
        <aaaUser descr="" dn="sys/user-ext/user-fun" email="" expiration="never"
            expires="no" firstName="Have" intId="12708" lastName="Fun" name="fun"
            phone=""
            priv="read-only" pwdSet="yes"/>
        <aaaUser descr="testuser" dn="sys/user-ext/user-aaa" email=""
        expiration="never"
            expires="no" firstName="a" intId="10620" lastName="aa" name="aaa"
            phone=""
            priv="aaa,read-only" pwdSet="no"/>
    </outConfigs>
</configResolveChildren>
```

configResolveClass

次に、該当するクラスの要求された管理対象オブジェクトを返す例を示します。inHierarchical が true の場合、戻されたオブジェクトにも子が含まれます。

要求構文

```
<xs:element name="configResolveClass" type="configResolveClass"
substitutionGroup="externalMethod"/>
    <xs:complexType name="configResolveClass" mixed="true">
        <xs:all>
            <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="inHierarchical">
            <xs:simpleType>
                <xs:union memberTypes="xs:boolean">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:enumeration value="no"/>
                            <xs:enumeration value="yes"/>
                        </xs:restriction>
```

```
                </xs:simpleType>
             </xs:union>
          </xs:simpleType>
       </xs:attribute>
       <xs:attribute name="cookie" type="xs:string"/>
       <xs:attribute name="response" type="YesOrNo"/>
       <xs:attribute name="classId" type="namingClassId"/>
    </xs:complexType>
```

応答構文

```
<xs:element name="configResolveClass" type="configResolveClass"
substitutionGroup="externalMethod"/>
    <xs:complexType name="configResolveClass" mixed="true">
       <xs:all>
          <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
       </xs:all>
       <xs:attribute name="cookie" type="xs:string"/>
       <xs:attribute name="response" type="YesOrNo"/>
       <xs:attribute name="errorCode" type="xs:unsignedInt"/>
       <xs:attribute name="errorDescr" type="xs:string"/>
       <xs:attribute name="invocationResult" type="xs:string"/>
       <xs:attribute name="classId" type="namingClassId"/>
    </xs:complexType>
```

例

要求

```
<configResolveClass
    cookie="<real_cookie>"
    classId="pkiEp"
    inHierarchical="false">
    <inFilter>
    </inFilter>
</configResolveClass>
```

応答

```
<configResolveClass
    cookie="<real_cookie>"
    commCookie="11/15/0/2a5b"
    srcExtSys="10.193.33.120"
    destExtSys="10.193.33.120"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes"
    classId="pkiEp">
    <outConfigs>
       <pkiEp descr=""
          dn="sys/pki-ext"
          intId="10037"
          name=""/>
    </outConfigs>
</configResolveClass>
```

configResolveClasses

次に、複数のクラスの要求された管理対象オブジェクトを返す例を示します。inHierarchical が true の場合、戻されたオブジェクトにも子が含まれます。

要求構文

```
<xs:element name="configResolveClasses" type="configResolveClasses"
substitutionGroup="externalMethod"/>
    <xs:complexType name="configResolveClasses" mixed="true">
```

```
      <xs:all>
         <xs:element name="inIds" type="classIdSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configResolveClasses" type="configResolveClasses"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveClasses" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<configResolveClasses
   cookie="<real_cookie>"
   inHierarchical="false">
   <inIds>
      <Id value="eventRecord"/>
      <Id value="faultInst"/>
   </inIds>
</configResolveClasses>
```

応答

```
<configResolveClasses
   cookie="<real_cookie>"
   commCookie="2/12/0/181a"
   srcExtSys="172.20.101.150"
   destExtSys="172.20.101.150"
   srcSvc="sam_extXMLApi"
   destSvc="service-reg_dme"
   response="yes">
   <outConfigs>
      <eventRecord
         affected="observe/observed-1001-1"
         cause="transition"
         changeSet=""
         code="E4194388"
         created="2012-07-25T00:39:35.528"
         descr="[FSM:BEGIN]: Resolve Mgmt Controller
```

```
Fsm(FSM:sam:dme:ObserveObservedResolveControllerFsm)"
          dn="event-log/10133"
          id="10133"
          ind="state-transition"
          severity="info"
          trig="special"
          txId="3"
          user="internal"/>
       <eventRecord
          affected="observe/observed-1001-1"
          cause="transition"
          changeSet=""
          code="E4194388"
          created="2012-07-25T00:39:35.528"
          descr="[FSM:STAGE:END]:
(FSM-STAGE:sam:dme:ObserveObservedResolveControllerFsm:begin)"
          dn="event-log/10134"
          id="10134"
          ind="state-transition"
          severity="info"
          trig="special"
          txId="3"
          user="internal"/>
   </outConfigs>
</configResolveClasses>
```

configResolveDn

次に、指定された DN について単一の管理対象オブジェクトを取得する例を示します。

要求構文

```
<xs:element name="configResolveDn" type="configResolveDn"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveDn" mixed="true">
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configResolveDn" type="configResolveDn"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveDn" mixed="true">
      <xs:all>
         <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
```

```
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

例

要求

```
<configResolveDn
   cookie="<real_cookie>"
   dn="vmmEp/vm-mgr-vcenter1" />
```

応答

```
<configResolveDn dn="vmmEp/vm-mgr-vcenter1"
   cookie="<real_cookie>"
   commCookie="9/15/0/1c0d"
   srcExtSys="10.193.34.70"
   destExtSys="10.193.34.70"
   srcSvc="sam_extXMLApi"
   destSvc="vm-mgr_dme"
   response="yes">
   <outConfig>
      <vmManager
         adminState="enable"
         descr=""
         dn="vmmEp/vm-mgr-vcenter1"
         fltAggr="0"
         fsmDescr="AG registration with
vCenter(FSM:sam:dme:VmManagerRegisterWithVCenter)"
         fsmPrev="RegisterWithVCenterRegistering"
         fsmProgr="13"
         fsmRmtInvErrCode="none"
         fsmRmtInvErrDescr=""
         fsmRmtInvRslt=""
         fsmStageDescr="AG registration with
vCenter(FSM-STAGE:sam:dme:VmManagerRegisterWithVCenter:Registering)"
         fsmStamp="2010-11-11T21:37:15.696"
         fsmStatus="RegisterWithVCenterRegistering"
         fsmTry="1"
         hostName="vpod-31.host123.com"
         intId="21959"
         name="vcenter1"
         operState="unknown"
         stateQual=""
         type="vmware"
         version=""/>
   </outConfig>
</configResolveDn>
```

configResolveDns

次に、DN のリストについて管理対象オブジェクトを取得する例を示します。

要求構文

```
<xs:element name="configResolveDns" type="configResolveDns"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveDns" mixed="true">
      <xs:all>
         <xs:element name="inDns" type="dnSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
```

```
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="no"/>
                        <xs:enumeration value="yes"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:union>
        </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="cookie" type="xs:string"/>
    <xs:attribute name="response" type="YesOrNo"/>
</xs:complexType>
```

応答構文

```
<xs:element name="configResolveDns" type="configResolveDns"
substitutionGroup="externalMethod"/>
    <xs:complexType name="configResolveDns" mixed="true">
        <xs:all>
            <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            <xs:element name="outUnresolved" type="dnSet" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

```
<configResolveDns
    cookie="<real_cookie>"
    inHierarchical="false">
    <inDns>
        <dn value="sys" />
    </inDns>
    </configResolveDns>
```

応答

```
<configResolveDns
    cookie="<real_cookie>"
    commCookie="5/12/0/1009"
    srcExtSys="172.25.103.136"
    destExtSys="172.25.103.136"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outConfigs>
        <topSystem
            address="172.25.103.136"
            currentTime="2012-08-01T00:47:44.663"
            dn="sys"
            mode="stand-alone"
            name="localhost"
            systemOrg=""
            systemUpTime="04:04:05:00"/>
    </outConfigs>
    <outUnresolved>
    </outUnresolved>
</configResolveDns>
```

configResolveParent

次に、指定された DN について管理対象オブジェクトの親を取得する例を示します。

要求構文

```
<xs:element name="configResolveParent" type="configResolveParent"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveParent" mixed="true">
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configResolveParent" type="configResolveParent"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configResolveParent" mixed="true">
      <xs:all>
         <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

例

要求

```
<configResolveParent
   cookie="<real_cookie>"
   inHierarchical="false"
   dn="org-root/org-tenant1/org-HR">
</configResolveParent>
```

応答

```
<configResolveParent
   dn="org-root/org-tenant/org-HR"
   cookie="<real_cookie>"
   commCookie="2/12/0/1837"
   srcExtSys="172.20.101.150"
   destExtSys="172.20.101.150"
   srcSvc="sam_extXMLApi"
   destSvc="service-reg_dme"
   response="yes">
      <outConfig>
      <orgTenant
      descr="tenant123"
```

```
      dn="org-root/org-tenant"
      fltAggr="0"
      level="1"
      name="tenant123"/>
   </outConfig>
</configResolveParent>
```

configScope

次に、管理対象オブジェクトとその設定に関する詳細を返す例を示します。

要求構文

```
<xs:element name="configScope" type="configScope"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configScope" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inClass" type="namingClassId"/>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inRecursive">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="configScope" type="configScope"
substitutionGroup="externalMethod"/>
   <xs:complexType name="configScope" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
```

```
        <xs:attribute name="invocationResult" type="xs:string"/>
        <xs:attribute name="dn" type="referenceObject"/>
    </xs:complexType>
```

例

要求

```
<configScope
    dn="org-root"
    cookie="<real_cookie>"
    inClass="orgOrgRes"
    inHierarchical="false"
    inRecursive="false">
    <inFilter>
    </inFilter>
</configScope>
```

応答

```
<configScope dn="org-root"
    cookie="<real_cookie>"
    commCookie="2/15/0/2a53"
    srcExtSys="10.193.33.120"
    destExtSys="10.193.33.120"
    srcSvc="sam_extXMLApi"
    destSvc="service-reg_dme"
    response="yes">
    <outConfigs>
        <orgOrgCaps
            dn="org-root/org-caps"
            org="512"
            tenant="64"/>
        <orgOrgCounts
            dn="org-root/org-counter"
            org="36"
            tenant="7"/>
    </outConfigs>
</configScope>
```

eventSendHeartbeat

次に、現在のセッションがまだアクティブであることを示すイベントを送信する例を示します。

要求構文

```
<xs:element name="eventSendHeartbeat" type="eventSendHeartbeat"
substitutionGroup="externalMethod"/>
    <xs:complexType name="eventSendHeartbeat" mixed="true">
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
    </xs:complexType>
```

応答構文

```
<xs:element name="eventSendHeartbeat" type="eventSendHeartbeat"
substitutionGroup="externalMethod"/>
    <xs:complexType name="eventSendHeartbeat" mixed="true">
        <xs:attribute name="outSystemTime" type="dateTime"/>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

クライアント アプリケーションが eventSubscribeApps または eventSubscribe を使用してイベントをサブスクライブする場合、VNMC は eventSendHeartbeat を定期的（デフォルトでは 120 秒）に送信します。

応答

```
<eventSendHeartbeat cookie="0/0/0/2a76"
    commCookie=""
    srcExtSys="0.0.0.0"
    destExtSys="0.0.0.0"
    srcSvc=""
    destSvc=""
    response="yes"
    outSystemTime="2010-11-12T20:38:19.630">
</eventSendHeartbeat>
```

eventSubscribe

次に、アクティビティに対するサブスクライブ要求を送信する例を示します。

要求構文

```
<xs:element name="eventSubscribe" type="eventSubscribe"
substitutionGroup="externalMethod"/>
    <xs:complexType name="eventSubscribe" mixed="true">
        <xs:all>
            <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
    </xs:complexType>
```

応答構文

```
<xs:element name="eventSubscribe" type="eventSubscribe"
substitutionGroup="externalMethod"/>
    <xs:complexType name="eventSubscribe" mixed="true">
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

```
<eventSubscribe
    cookie="<real_cookie>">
    <inFilter>
    </inFilter>
</eventSubscribe>
```

応答

VNMC は、応答または通知を送信しません。

eventSubscribeApps

次に、指定されたアプリケーションのアクティビティに対するサブスクライブ要求の例を示します。クライアント アプリケーションは、さまざまなアプリケーションからイベントを受け取るために VNMC システムをサブスクライブできます。eventApplication では、ip はアプリケーション（DME）が実行されている VM の IP アドレスです。クライアント アプリケーションは VSG からのイベントも受け取るようサブスクライブできます。この場合、ip は VSG の IP アドレスである必要があり、タイプは管理対象エンドポイントです。

要求構文

```
<xs:element name="eventSubscribeApps" type="eventSubscribeApps"
substitutionGroup="externalMethod"/>
   <xs:complexType name="eventSubscribeApps" mixed="true">
      <xs:all>
         <xs:element name="inAppList" type="configSet" minOccurs="0"/>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="eventSubscribeApps" type="eventSubscribeApps"
substitutionGroup="externalMethod"/>
   <xs:complexType name="eventSubscribeApps" mixed="true">
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<eventSubscribeApps
   cookie="<real_cookie>"
   commCookie=""
   srcExtSys="0.0.0.0"
   destExtSys="0.0.0.0"
   <inAppList>
      <eventApplication
         ip="10.193.33.101"
         type="service-reg"/>
         <eventApplication
         ip="10.193.33.101"
         type="policy-mgr"/>
      <eventApplication
         ip="10.193.33.101"
         type="mgmt-controller"/>
   </inAppList>
   <inFilter>
   </inFilter>
</eventSubscribeApps>
```

応答

要求が成功すると、VNMC は応答または通知を送信しません。

faultAckFault

次に、障害が記録されたときに確認応答を送信する例を示します。

要求構文

```
<xs:element name="faultAckFault" type="faultAckFault"
substitutionGroup="externalMethod"/>
   <xs:complexType name="faultAckFault" mixed="true">
      <xs:attribute name="inId" type="xs:unsignedLong"/>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="faultAckFault" type="faultAckFault"
substitutionGroup="externalMethod"/>
   <xs:complexType name="faultAckFault" mixed="true">
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<faultAckFault
   inHierarchical="false"
   cookie="<real_cookie>"
   inId="10120" />
```

応答

```
<faultAckFault
   cookie="<real_cookie>"
   commCookie="5/15/0/6c"
   srcExtSys="10.193.33.214"
   destExtSys="10.193.33.214"
   srcSvc="sam_extXMLApi"
   destSvc="resource-mgr_dme"
   response="yes">
</faultAckFault>
```

faultAckFaults

次に、複数の障害が記録されたときに確認応答を送信する例を示します。

要求構文

```
<xs:element name="faultAckFaults" type="faultAckFaults"
substitutionGroup="externalMethod"/>
   <xs:complexType name="faultAckFaults" mixed="true">
      <xs:all>
         <xs:element name="inIds" type="idSet" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="faultAckFaults" type="faultAckFaults"
substitutionGroup="externalMethod"/>
   <xs:complexType name="faultAckFaults" mixed="true">
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<faultAckFaults
   cookie="<real_cookie>">
   <inIds>
```

```
      <id value="10656"/>
      <id value="10660"/>
   </inIds>
</faultAckFaults>
```

応答

```
<faultAckFaults
   cookie="<real_cookie>"
   commCookie="11/15/0/505"
   srcExtSys="10.193.34.70"
   destExtSys="10.193.34.70"
   srcSvc="sam_extXMLApi"
   destSvc="mgmt-controller_dme"
   response="yes">
</faultAckFaults>
```

faultResolveFault

次に、障害が解決されたときに応答を送信する例を示します。

要求構文

```
<xs:element name="faultResolveFault" type="faultResolveFault"
substitutionGroup="externalMethod"/>
   <xs:complexType name="faultResolveFault" mixed="true">
      <xs:attribute name="inId" type="xs:unsignedLong"/>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="faultResolveFault" type="faultResolveFault"
substitutionGroup="externalMethod"/>
   <xs:complexType name="faultResolveFault" mixed="true">
      <xs:all>
         <xs:element name="outFault" type="configConfig" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
      <xs:attribute name="invocationResult" type="xs:string"/>
   </xs:complexType>
```

例

要求

```
<faultResolveFault
   inHierarchical="false"
   cookie="<real_cookie>"
   inId="10120" />
```

応答

```
<faultResolveFault
   cookie="<real_cookie>"
   commCookie="5/15/0/6a"
   srcExtSys="10.193.33.214"
   destExtSys="10.193.33.214"
   srcSvc="sam_extXMLApi"
   destSvc="resource-mgr_dme"
   response="yes">
   <outFault>
      <faultInst
         ack="yes"
```

```
            cause="empty-pool"
            changeSet=""
            code="F0135"
            created="2010-11-19T11:02:41.568"
            descr="Virtual Security Gateway pool default is empty"
            dn="org-root/fwpool-default/fault-F0135"
            highestSeverity="minor"
            id="10120"
            lastTransition="2010-11-19T11:02:41.568"
            lc=""
            occur="1"
            origSeverity="minor"
            prevSeverity="minor"
            rule="fw-pool-empty"
            severity="minor"
            tags=""
            type="equipment"/>
    </outFault>
</faultResolveFault>
```

loggingSyncOcns

次に、DME からイベント ID を取得する例を示します。

要求構文

```
<xs:element name="loggingSyncOcns" type="loggingSyncOcns"
substitutionGroup="externalMethod"/>
    <xs:complexType name="loggingSyncOcns" mixed="true">
        <xs:attribute name="inFromOrZero" type="xs:unsignedLong"/>
        <xs:attribute name="inToOrZero" type="xs:unsignedLong"/>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
    </xs:complexType>
```

応答構文

```
<xs:element name="loggingSyncOcns" type="loggingSyncOcns"
substitutionGroup="externalMethod"/>
    <xs:complexType name="loggingSyncOcns" mixed="true">
        <xs:all>
            <xs:element name="outStimuli" type="MethodSet" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

```
<loggingSyncOcns
    cookie="<real_cookie>"
    inFromOrZero="0"
    inToOrZero="4567000"/>
```

応答

イベント ID のリスト。

orgResolveElements

この例では、指定された DN 内で、クエリー フィルタを満たす管理対象オブジェクトが取得され、組織（オプションでは子組織）で開始された管理対象オブジェクトが検索されます。この DN を持つ組織がない場合は、空のマップが返されます。この DN を持つ組織がある場合は、指定されたクラスとフィルタで管理対象オブ

ジェクトが検索されます。inHierarchical が true の場合、戻されたオブジェクトにも子が含まれます。inHierarchical が false の場合は、一致するオブジェクトだけが返されます。inSingleLevel が true の場合は、開始設定レベルのオブジェクトのみ返されます。inSingleLevel が false の場合、子組織のオブジェクトも返されます。

要求構文

```
<xs:element name="orgResolveElements" type="orgResolveElements"
substitutionGroup="externalMethod"/>
   <xs:complexType name="orgResolveElements" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inClass" type="namingClassId"/>
      <xs:attribute name="inSingleLevel">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="orgResolveElements" type="orgResolveElements"
substitutionGroup="externalMethod"/>
   <xs:complexType name="orgResolveElements" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configMap" minOccurs="0">
            <xs:unique name="unique_map_key_5">
               <xs:selector xpath="pair"/>
               <xs:field xpath="@key"/>
            </xs:unique>
         </xs:element>
      </xs:all>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="errorCode" type="xs:unsignedInt"/>
      <xs:attribute name="errorDescr" type="xs:string"/>
```

```
        <xs:attribute name="invocationResult" type="xs:string"/>
        <xs:attribute name="dn" type="referenceObject"/>
    </xs:complexType>
```

例

要求

```
<orgResolveElements
    dn="org-root/org-Cola"
    cookie="<real_cookie>"
    commCookie="7/15/0/19"
    inClass="policyPolicySet"
    inSingleLevel="no"
    inHierarchical="no">
    <inFilter>
    </inFilter>
</orgResolveElements>
```

応答

```
<orgResolveElements
    dn="org-root/org-Cola"
    cookie="<real_cookie>"
    commCookie="7/15/0/19"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfigs>
        <pair key="pset-default">
            <policyPolicySet
                descr="The default Policy Set"
                dn="org-root/pset-default"
                intId="10082"
                name="default"/>
        </pair>
        <pair key="pset-myPolicySet3">
            <policyPolicySet
                descr=""
                dn="org-root/org-Cola/pset-myPolicySet3"
                intId="12289"
                name="myPolicySet3"/>
        </pair>
        <pair key="pset-policySetSanity">
            <policyPolicySet
                descr=""
                dn="org-root/org-Cola/pset-policySetSanity"
                intId="24627"
                name="policySetSanity"/>
        </pair>
        <pair key="pset-pci_compliance_f">
            <policyPolicySet
                descr=""
                dn="org-root/pset-pci_compliance_f"
                intId="24539"
                name="pci_compliance_f"/>
        </pair>
        <pair key="pset-pci_compliance_h">
            <policyPolicySet
                descr=""
                dn="org-root/pset-pci_compliance_h"
                intId="24541"
```

```
            name="pci_compliance_h"/>
      </pair>
   </outConfigs>
</orgResolveElements>
```

orgResolveInScope

次に、システムが、該当する DN を持つ組織と親組織（任意）をルートまで再帰的に検索する例を示します。組織が検出されない場合は、空のマップが返されます。組織がある場合は、指定されたクラスとフィルタですべてのプールが検索されます。

（注） inSingleLevel が false の場合は、親組織が、ルート ディレクトリまで検索されます。

要求構文

```
<xs:element name="orgResolveInScope" type="orgResolveInScope"
substitutionGroup="externalMethod"/>
   <xs:complexType name="orgResolveInScope" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inClass" type="namingClassId"/>
      <xs:attribute name="inSingleLevel">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="orgResolveInScope" type="orgResolveInScope"
substitutionGroup="externalMethod"/>
   <xs:complexType name="orgResolveInScope" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configMap" minOccurs="0">
            <xs:unique name="unique_map_key_6">
               <xs:selector xpath="pair"/>
```

```
                <xs:field xpath="@key"/>
            </xs:unique>
        </xs:element>
    </xs:all>
    <xs:attribute name="cookie" type="xs:string"/>
    <xs:attribute name="response" type="YesOrNo"/>
    <xs:attribute name="errorCode" type="xs:unsignedInt"/>
    <xs:attribute name="errorDescr" type="xs:string"/>
    <xs:attribute name="invocationResult" type="xs:string"/>
    <xs:attribute name="dn" type="referenceObject"/>
</xs:complexType>
```

例

要求

```
<orgResolveInScope
    cookie="<real_cookie>"
    dn="org-root/org-Cola"
    inClass="policyVirtualNetworkServiceProfile"
    inHierarchical="true"
    InSingleLevel="false" >
    <inFilter>
        <eq class="policyVirtualNetworkServiceProfile"
            property="name"
            value="spsanity" />
    </inFilter>
</orgResolveInScope>
```

応答

```
<orgResolveInScope
    dn="org-root/org-Cola"
    cookie="<real_cookie>"
    commCookie="7/15/0/1c35"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="vnsp-spsanity">
            <policyVirtualNetworkServiceProfile
                childAction="deleteNonPresent"
                descr=""
                dn="org-root/org-Cola/vnsp-spsanity"
                intId="82018"
                name="spsanity"
                policySetNameRef="policySetSanity"
                vnspId="41">
            <policyVnspAVPair
                childAction="deleteNonPresent"
                descr=""
                id="1"
                intId="82019"
                name=""
                rn="vnsp-avp-1">
                <policyAttributeValue
                    childAction="deleteNonPresent"
                    id="1"
                    rn="attr-val1"
                    value="DEV"/>
                <policyAttributeDesignator
                    attrName="dept"
                    childAction="deleteNonPresent"
                    rn="attr-ref"/>
```

```
            </policyVnspAVPair>
         </policyVirtualNetworkServiceProfile>
      </pair>
   </outConfigs>
</orgResolveInScope>
```

poolResolveInScope

次に、システムが、該当する DN を持つプールと親プール（任意）をルートまで再帰的に検索する例を示します。プールがない場合は、空のマップが返されます。プールがある場合は、指定されたクラスとフィルタですべてのプールが検索されます。

（注） inSingleLevel が false の場合は、親プールがルート ディレクトリまで検索されます。

要求構文

```
<xs:element name="poolResolveInScope" type="poolResolveInScope"
substitutionGroup="externalMethod"/>
   <xs:complexType name="poolResolveInScope" mixed="true">
      <xs:all>
         <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
      </xs:all>
      <xs:attribute name="inClass" type="namingClassId"/>
      <xs:attribute name="inSingleLevel">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="inHierarchical">
         <xs:simpleType>
            <xs:union memberTypes="xs:boolean">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <xs:enumeration value="no"/>
                     <xs:enumeration value="yes"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:union>
         </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="cookie" type="xs:string"/>
      <xs:attribute name="response" type="YesOrNo"/>
      <xs:attribute name="dn" type="referenceObject"/>
   </xs:complexType>
```

応答構文

```
<xs:element name="poolResolveInScope" type="poolResolveInScope"
substitutionGroup="externalMethod"/>
   <xs:complexType name="poolResolveInScope" mixed="true">
      <xs:all>
         <xs:element name="outConfigs" type="configMap" minOccurs="0">
            <xs:unique name="unique_map_key_9">
```

```
                <xs:selector xpath="pair"/>
                <xs:field xpath="@key"/>
            </xs:unique>
        </xs:element>
    </xs:all>
    <xs:attribute name="cookie" type="xs:string"/>
    <xs:attribute name="response" type="YesOrNo"/>
    <xs:attribute name="errorCode" type="xs:unsignedInt"/>
    <xs:attribute name="errorDescr" type="xs:string"/>
    <xs:attribute name="invocationResult" type="xs:string"/>
    <xs:attribute name="dn" type="referenceObject"/>
</xs:complexType>
```

例

要求

```
<poolResolveInScope
    dn="org-root/org-tenant1"
    cookie="<real_cookie>"
    />
```

応答

```
<poolResolveInScope
    dn="org-root/org-cisco"
    cookie="<real_cookie>"
    commCookie="5/12/0/19"
    srcExtSys="172.25.103.136"
    destExtSys="172.25.103.136"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outConfigs>
        <pair key="fwpool-default">
            <fwPool
                assigned="0"
                descr="Default Pool of firewall resources"
                dn="org-root/fwpool-default"
                fltAggr="65536"
                id="1"
                intId="10066"
                name="default"
                size="0"/>
        </pair>
    </outConfigs>
</poolResolveInScope>
```

付録：UML ダイアグラム

ここでは、次の内容について説明します。

- VPN モデル
- 一般的なルールベースのポリシー モデル

VPN モデル

図 4　VPN モデルの詳細なデバイス ビュー

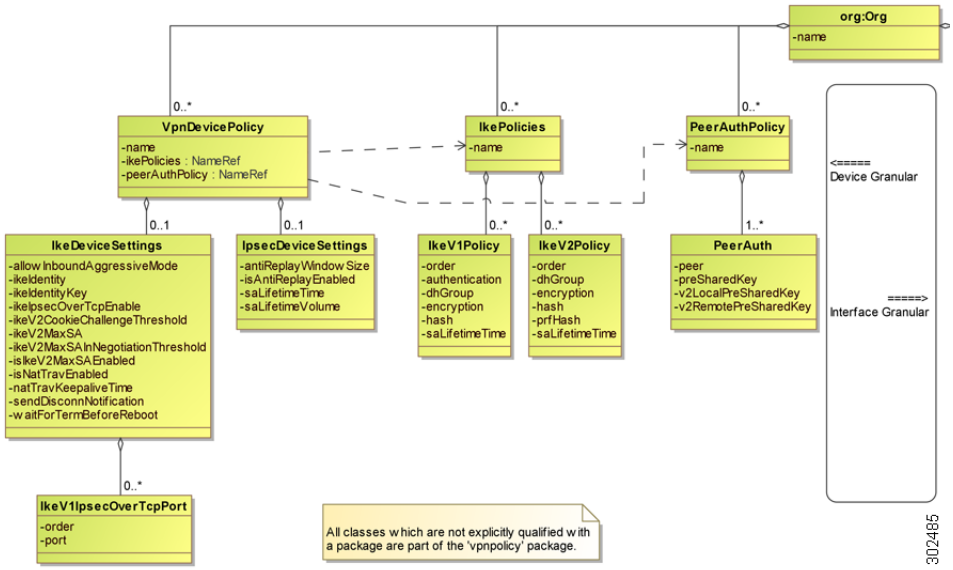
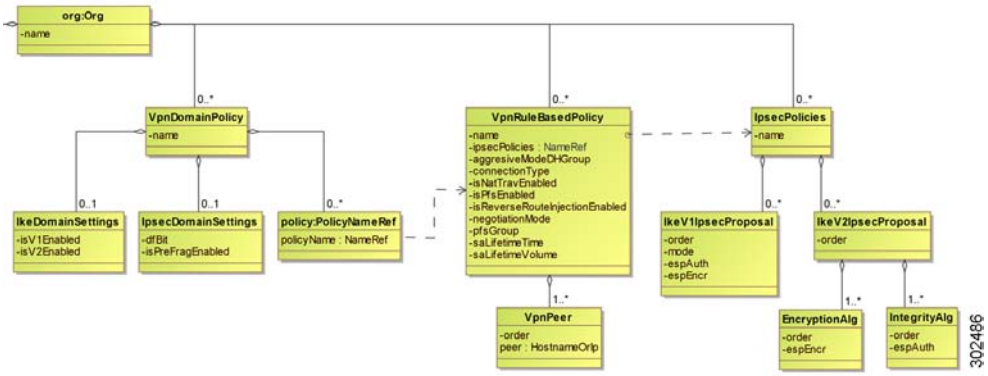
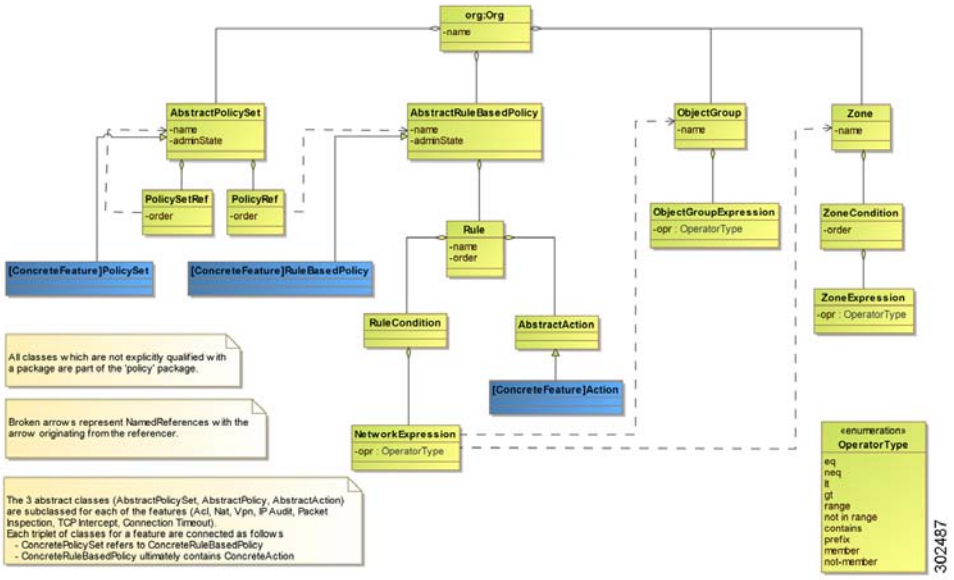


図 5　VPN モデルの詳細なインターフェイス ビュー



一般的なルールベースのポリシー モデル

図 6　一般的なルールベースのポリシー モデル



索引

マニュアルの入手方法およびテクニカル サポート

マニュアルの入手方法、テクニカル サポート、その他の有用な情報について、次の URL の『*What's New in Cisco Product Documentation*』を参照してください。
http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

『*What's New in Cisco Product Documentation*』は、シスコの新規および改訂版の技術マニュアルの一覧も示し、RSS フィードとして購読できます。また、リーダー アプリケーションを使用してコンテンツをデスクトップに配信することもできます。RSS フィードは無料のサービスです。

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先:シスコ コンタクトセンター
0120-092-255(フリーコール、携帯・PHS含む)
電話受付時間 : 平日 10:00〜12:00、13:00〜17:00
http://www.cisco.com/jp/go/contactcenter/